A Solution to the Problem of the Maximal Number of Symbols for Biomolecular Computer

Jacek Waldmajer
Institute of Computer Science, University of Opole, Oleska 48, 45-052 Opole, Poland
E-mail: jwaldmajer@uni.opole.pl

Sebastian Sakowski
Faculty of Mathematics and Computer Science, University of Lodz, Banacha 22, 90-238 Lodz, Poland
E-mail: sebastian.sakowski@wmii.uni.lodz.pl

Keywords: biomolecular computer, biomolecular systems, DNA computing

Received: March 15, 2019

The authors present a solution to the problem of generating the maximum possible number of symbols for a biomolecular computer using restriction enzyme $BbvI$ and ligase as the hardware, and transition molecules built of double-stranded DNA as the software. The presented solution offers an answer to the open question, in the algorithm form, of the maximal number of symbols for a biomolecular computer that makes use of the restriction enzyme $BbvI$.

Povzetek: Razvit je nov način izračuna največjega števila simbolov za biomulekularni računalnik.

1 Introduction

The beginnings of research into possibilities of applying biomolecules to control biological systems, and also to construct computers, are to be found in theoretical works of the 1960s (Feynman 1961). Then, in the 1980s, Charles Bennett (1982, Bennett and Landauer 1985) pointed to potential possibilities of application of biomolecules to construct energy-efficient nanodevices. However, the world had to wait to see the first practical experiments realizing simple calculations with the use of biochemical reactions until the mid-1990s, when Leonard Adleman (1994) solved the problem of Hamilton's path in graph, using exclusively a biomolecule for this purpose. Successive research revealed the possibility of spontaneous formation of multidimensional structures built from biomolecules, which were made with the use of the conception of self-assembly (Whitesides et al. 1991, Seeman 2001, Gopinath et al. 2016). The multidimensional DNA structures made it possible to realize fractals, e.g., ones of Sierpiński triangle type (Rothemund 2004), which revealed a great potential in calculations based on self-assembly. In 2006, Paul Rothemund (2006) made use of self-assembling DNA molecules to obtain different multidimensional biomolecular structures. Properly prepared DNA molecules also made it possible to carry out a theoretical simulation of Turing machine (Rothemund 1995). Prior to this, in 2001 (Benenson et al. 2001) a practically acting non-deterministic finite automaton based on such DNA molecules, restriction enzyme FokI and DNA ligase was presented. In successive research, it was proved experimentally that such an automaton can work without the use of ligase enzyme (Benenson et al. 2003, Chen et al. 2007) and its complexities were extended in practical experiments, ones understood as the number of states using numerous restriction enzymes (Sakowski et al. 2017). It is worth adding that it was with success that laboratory experiments were carried out, in which this biomolecular system was applied to medical diagnosis and treatment (Benenson et al. 2004) and also to simple logical inference (Ran 2009). In another work which dealt with possibilities of applying DNA molecules, a challenge was taken up to not only increase the number of states of such an automaton (Unold et al. 2004), but also that of symbols possible for an automaton built from DNA (Soreni et al. 2005). Moreover, presented the notion of biomolecular automaton, informally characterized in the papers of Rothemund (1995), Benenson et al. (2001), Soreni et al (2005), was presented in a formal way (as a mathematical model called a tailor automaton in a new theory of tailor automata) in the paper Waldmajer et al. (2019).

In the above-mentioned work, Soreni and co-workers (Soreni et al. 2005) put forward a 3-state 3-symbol biomolecular automaton which used the restriction enzyme BbvI as well as considered the problem of determining the maximal number of symbols for the constructed biomolecular automaton. On the basis of the conducted assessment they pointed out that it is possible to construct 40 symbols, each of which is composed of 6 pairs of nucleotides. However, in their work, they pointed to merely 37 such symbols, including one which was erroneously determined. Consequently, they opened the following issue (p. 3937): *It is still an open question whether the maximal number of 6-bp sequences that produce distinct 4-bp sticky ends in both*

strands is 40. It is with reference to this open question that the authors of the present work undertook and managed to solve the problem mentioned by Soreni et al. in their work (2005) through: (1) indicating 40 symbols (see Tab. 4) which make the solution to the open problem, (2) proposing the idea of working of an algorithm that enables to generate 40 symbols for a biomolecular automaton using the restriction enzyme BbvI, and (3) formulating two general problems in the sphere of generating symbols for biomolecular automata which use one restriction enzyme (among which a biomolecular automaton using the restriction enzyme BbvI is a particular case) and more than one restriction enzyme.

The second section presents the idea of constructing and working of a 3-state 3-symbol biomolecular automaton using the restriction enzyme BbvI as presented by Soreni and co-workers in their work (Soreni et al. 2005). In the third section the conception of working of an algorithm generating the maximal number of symbols for a biomolecular automaton using the restriction enzyme BbvI was presented together with a discussion of various undesired situations which may occur in the course of working of a biomolecular automaton that makes use of one restriction enzyme (in particular for the restriction enzyme BbvI). In the last section, there were formulated two general problems of generating the maximal number of symbols for a certain class of biomolecular automata using one or more than one restriction enzymes.

2 Biomolecular finite automaton and the idea of its actions

In this section, we make a presentation of the 3-state 3-symbol biomolecular finite DNA automaton (see Fig. 1), which was presented by Soreni and co-workers (Soreni et al. 2005). The automaton uses the restriction enzyme BbvI, ligase enzyme and DNA double-stranded fragments (input molecule, set of transition molecules and set of detection molecules). The double-stranded DNA fragments include the adenine, cytosine, guanine, and thymine bases marked as A, C, G and T, respectively.

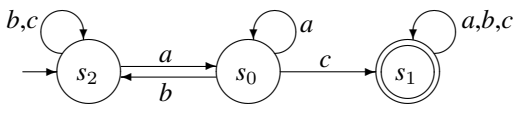

Figure 1: Graph representing a 3-state 3-symbol deterministic finite automaton M_1.

The task of the BbvI restriction enzyme is to cut the double-stranded DNA after recognizing a specific sequence (see Fig. 2A) in the double-stranded DNA.

The BbvI restriction enzyme will cut the double-stranded DNA after the 8th nucleotide in the DNA strand in the 5′-3′ direction and after the 12th nucleotide in the DNA strand in the 3′-5′ direction from the recognized specific sequence (see Fig. 2B).

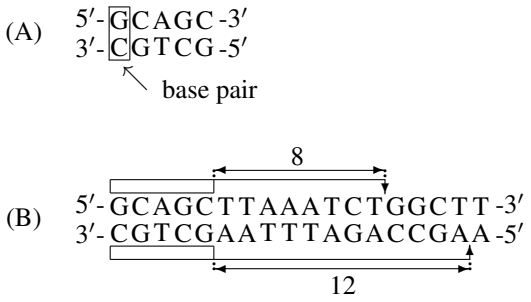

Figure 2: (A) Specific sequence recognized by the BbvI restriction enzyme. (B) The action of restriction endonuclease: BbvI.

The task of the ligase enzyme is to ligate the two double-stranded DNAs having complementary sticky ends (see Fig. 4A and 4B), where a sticky end is a single-stranded DNA at the end of a double-stranded DNA. In the given sense, the sticky end 'TTTA' of a single-stranded DNA (see Fig. 4A) is complementary to a sticky end 'AAAT' of the other double-stranded DNA (see Fig. 4B). The result of their ligation is one double-stranded DNA (see Fig. 4C).

Both the restriction enzyme BbvI and the ligase enzyme play the key role in the action of a biomolecular automaton, determining, respectively: the operation of cutting of a fragment of the double-stranded DNA and the operation of ligating of two fragments of double-stranded DNAs.

The input molecule (see Fig. 3) is a double-stranded DNA fragment in which it is possible to distinguish the following three basic parts: the input word x consisting of the symbols a, b and c (x = acb), the terminal symbol and the base sequence. At the both ends of the input molecule there occur additional base pairs and their occurrence is determined by the properties related to the action of the restriction enzyme.

To construct an input word of the 3-state 3-symbol deterministic finite automaton, the following three symbols: a, b and c (see Fig. 5) were used. These symbols were coded by means of six base pairs. Besides the aforementioned symbols, the additional terminal symbol t was introduced. This symbol is coded by means of the same number of base pairs as the symbols a, b and c. This symbol was used to acquire an output molecule which is used to determine whether the automaton has finished acting in the required state and has accepted the input word x.

The base sequence consists of a certain number of base pairs, contains a specific sequence recognizable by the BbvI restriction enzyme, and makes it possible to define the start state by determining the cut place of the input molecule by the BbvI restriction enzyme (cf. Fig. 3 and Fig. 2B). Let us note that the term "base sequence" did not appear in work Soreni et al. (2005). Introducing this term is meant to clearly determine the manner of setting the start state of a biomolecular automaton. According to the idea contained in the work of Soreni and co-workers

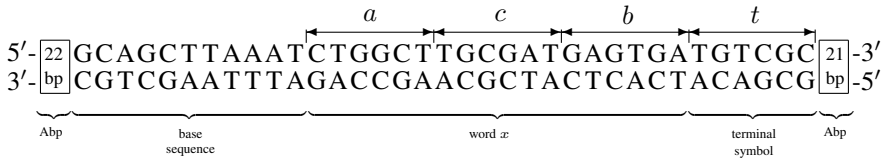

Figure 3: Input molecule containing the input word $x = acb$; Abp – Additional base pairs.

Table 1: Connection of the states s_0, s_1 and s_2 of a biomolecular automaton with the permanent cut places of the symbols a, b, c and t of the biomolecular automaton.

state	*symbol a*	*symbol b*	*symbol c*	*symbol t*
s_0	5′-|CTGGCT-3′ 3′-GACC|GA-5′	5′-|GAGTGA-3′ 3′-CTCA|CT-5′	5′-|TGCGAT-3′ 3′-ACGC|TA-5′	5′-|TGTCGC-3′ 3′-ACAG|CG-5′
s_1	5′-C|TGGCT-3′ 3′-GACCG|A-5′	5′-G|AGTGA-3′ 3′-CTCAC|T-5′	5′-T|GCGAT-3′ 3′-ACGCT|A-5′	5′-T|GTCGC-3′ 3′-ACAGC|G-5′
s_2	5′-CT|GGCT-3′ 3′-GACCGA|-5′	5′-GA|GTGA-3′ 3′-CTCACT|-5′	5′-TG|CGAT-3′ 3′-ACGCTA|-5′	5′-TG|TCGC-3′ 3′-ACAGCG|-5′

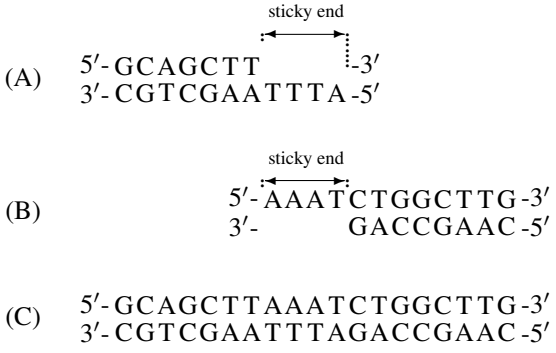

Figure 4: (A) and (B) Double-stranded DNAs with the complementary sticky ends. (C) The result of a ligation between the double-stranded DNAs with the complementary sticky ends included in (A) and (B).

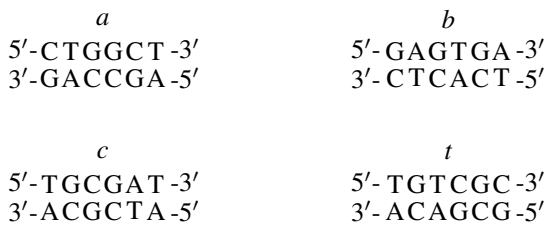

Figure 5: Symbols a, b, c and the terminal symbol t.

(Soreni et al. 2005), the reading of a symbol in a certain state of the automaton is identified with the cutting of the double-stranded DNA by the BbvI restriction enzyme in the area of a symbol, in a determined (permanent) place of the DNA strand, in the 5′-3′ direction and in a determined (permanent) place of the DNA strand in the 3′-5′ direction. Tab. 1 presents a connection between the states and two permanent cut places of the symbols.

In accordance with the input molecule presented on Fig. 3, the first cutting input molecule with the use of the restriction enzyme BbvI will follow in the area of the symbol a, which corresponds to the state s_2. In this way, in the state s_2, the symbol a was read and a fragment of DNA was obtained as presented on Fig. 6. In this sense, the state s_2 is a initial state. Adding to the base sequence of one or two pairs of nucleotides can set the start state to be the following: s_1 or s_0, respectively.

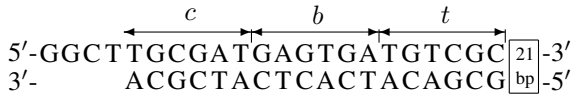

Figure 6: Double-stranded DNA fragment obtained after BbvI acting on the input molecule.

The set of transition molecules is used to implement a set of transitions in the 3-state 3-symbol deterministic finite automaton. We obtain transition from one state to the other (the same or another state), upon reading a symbol, through ligating with the use of ligase enzyme, of a DNA fragment obtained on Fig. 6 with one of the transition molecules. Each transition molecule contains a specific sequence recognizable by the BbvI restriction enzyme and the additional base pairs. Exemplary transition molecules are presented in Fig. 7: the transition molecule presented on Fig. 7A enables transition from the state s_0 to that of s_1 after reading the symbol c; the transition molecule presented on Fig. 7B enables transition from the state s_1 to that of s_1 after reading the symbol b; the transition molecule presented on Fig. 7C enables transition from the state s_2 to that of s_0 after reading the symbol a.

For each state, one detection molecule is constructed and thus a set of detection molecules is specified (see Fig. 8). It should be noted that the detection molecules have different numbers of additional base pairs, which makes it possible to determine laboratorily the state in which the automaton finished its action.

The beginning of the work: the BbvI, ligase enzyme, many copies of the transition molecules and many copies

(A) 5′-[35 bp]GCAGCT -3′
3′-CGTCGAACGC-5′

(B) 5′-[39 bp]GCAGCTT -3′
3′-CGTCGAATCAC-5′

(C) 5′-[35 bp]GCAGCTATT -3′
3′-CGTCGATAACCGA-5′

Figure 7: Selected transition molecules used in the transition function: (A) T_1: $(s_0, c) \rightarrow s_1$, (B) T_2: $(s_1, b) \rightarrow s_1$, (C) T_3: $(s_2, a) \rightarrow s_0$.

(A) 5′-[54 bp]ACAG-5′ -3′ (B) 5′-[200 bp]CAGC-5′ -3′

(C) 5′-[300 bp]AGCG-5′ -3′

Figure 8: (A) Detection molecule D_1 for the state s_0. (B) Detection molecule D_2 for the state s_1. (C) Detection molecule D_3 for state s_2.

of the detection molecules are placed in a laboratory tube; the final addition is many copies of the input molecule. After these elements have been mixed in the test tube, the biomolecular automaton starts its action. In successive steps there follows reading of the symbol a in the state s_2 (see Fig. 9a), making use of the transition molecule shown in Fig. 7C to transition from the state s_2 to that of s_0 after reading the symbol a (see Fig. 9b), reading of the symbol c in the state s_0 (see Fig. 9c), using the transition molecule presented in Fig. 7A to transition from the state s_0 to the state s_1 after reading the symbol c (see Fig. 9d) reading the symbol b in the state s_1 (see Fig. 9e), using the transition molecule presented in Fig. 7B and reading the terminal symbol t in the state s_1 (see. Fig. 9f-g). In the last step there follows ligation of a fragment of double-stranded DNA presented in Fig. 9g with one of the detection molecules (see Fig. 8B). As a result of ligation of these DNA fragments an output molecule is formed (see Fig. 9h), which – from the laboratory point of view – serves to determine the end state of a biomolecular finite automaton.

3 Algorithm for the problem of the maximal number of symbols

3.1 The formal aparatus used in the description of the algorithm

Let the set $\Delta = \{\text{A, C, G, T}\}$ and the function σ, which is bijection of the set Δ on Δ, which is defined in the following way: σ(A) = T, σ(T) = A, σ(C) = G and σ(G) = C be given. The set Δ is called a *set of nucleotides*, the elements of the set Δ are called *nucleotides*, and the function σ is called *complementarity of nucleotides*.

We call any finite sequence of nucleotides of the set Δ as a *word*. The word x which is the sequence $X_1, X_2, \ldots, X_j$ of nucleotides of the set Δ ($X_i \in \Delta$, $0 < i \leq j \in N$) is written as follows $x = X_1X_2 \ldots X_j$. The number of the elements of the sequence x is called the *length of the word* x (denoted symbolically: $|x|$), while the i-th nucleotide of the word x (the i-th element of the word x) as $x(i)$. The set of all the words formed from the nucleotides of the set Δ, whose length is greater than zero, is denoted as Δ^+.

Let $\Delta^+ \ni x = X_1X_2 \ldots X_j$ ($X_i \in \Delta$, $0 < i \leq j \in N$) and $\Delta^+ \ni y = Y_1Y_2 \ldots Y_j$ ($Y_i \in \Delta$, $0 < i \leq j \in N$). We call the word $X_j \ldots X_2X_1$ an *opposite word* (we denote symbolically: x^{-1}) to the word x. We call the word $xy = X_1X_2 \ldots X_iY_1Y_2 \ldots Y_j$ a *concatenation* xy of two words x and y such that $\Delta^+ \ni x = X_1\ X_2 \ldots X_i$ ($X_i \in \Delta$, $0 < i \in N$) and $\Delta^+ \ni y = Y_1Y_2 \ldots Y_j$ ($Y_j \in \Delta$, $0 < j \in N$). We say that the word x is *included in the word y, beginning with the k-th* ($1 \leq k \in N$) *position* (we denote symbolically: $x \subseteq_k y$), if $k + |x| \leq |y| + 1$ and $\exists u, v \in \Delta^*(y = uxv \wedge |u| = k - 1)$. The word x is a *sub-word of y* (we denote symbolically: $x \subseteq y$) when the word x is included in the word y, beginning with a certain position k, i.e., $x \subseteq y \Leftrightarrow \exists k(x \subseteq_k y)$. The word x is a prefix of the word y, when $x \subseteq_1 y$. The word x is a suffix of the word y, when x^{-1} is the prefix of the word y^{-1}.

The introduced notion of complementarity of nucleotides and the introduced denotations make it possible to define the function which will be called *complementariness of words*. The mapping Ξ: $\Delta^+ \rightarrow \Delta^+$ defined in the following way: $\Xi(x) = y$, where $|y| = |x|$ and $y(i) = \sigma(x(i))$ for each $i \in \{1, \ldots, |y|\}$ and, for $x \in \Delta^+$ is called *complementarity of words*.

Let $\Delta^+ \ni x = X_1X_2X_3X_4$ ($X_i \in \Delta$, $0 < i \leq j \in N$) and $\Delta^+ \ni y = Y_1Y_2Y_3Y_4$ ($Y_i \in \Delta$, $0 < i \leq l \in N$). The words x and y are *synthesable over the length* 3, when there exists the word $u \in \Delta^+$ of the length 3 being the suffix of the word x and the prefix of the word y. The concatenation of the synthesable words x and y over the length 3 is the word $z = [x, y]_3$, where $z = X_1X_2X_3X_4Y_4$.

3.2 Description of the algorithm

The idea of the algorithm of generating the maximal number of symbols for a biomolecular automaton using the restriction enzyme BbvI will be characterized through four stages, which are distinguished in the algorithm: the initial stage, the stage of deployment and verification, the stage of generation and the final stage. At each of the indicated stages we make use only of strands of symbols in the direction 5′-3′, since having strands of symbols in the direction 5′-3′, we can – by means of the principle of complementarity of nucleotides – obtain strands of symbols in the direction 3′-5′.

Let the set $\mathcal{A}_0$ of all 4-element sequences of nucleotides be given: $\mathcal{A}_0 = \{x : x \in \Delta^+ \wedge |x| = 4\}$=$\{\text{AAAA, AAAC, AAAG, \ldots, TTTG, TTTT}\}$. At the initial stage, we remove words (4-element sequences of nucleotides): AATT, ACGT, AGCT, ATAT, CCGG, CATG, CTAG, CGCG, GGCC, GATC, GTAC, GCGC, TTAA,

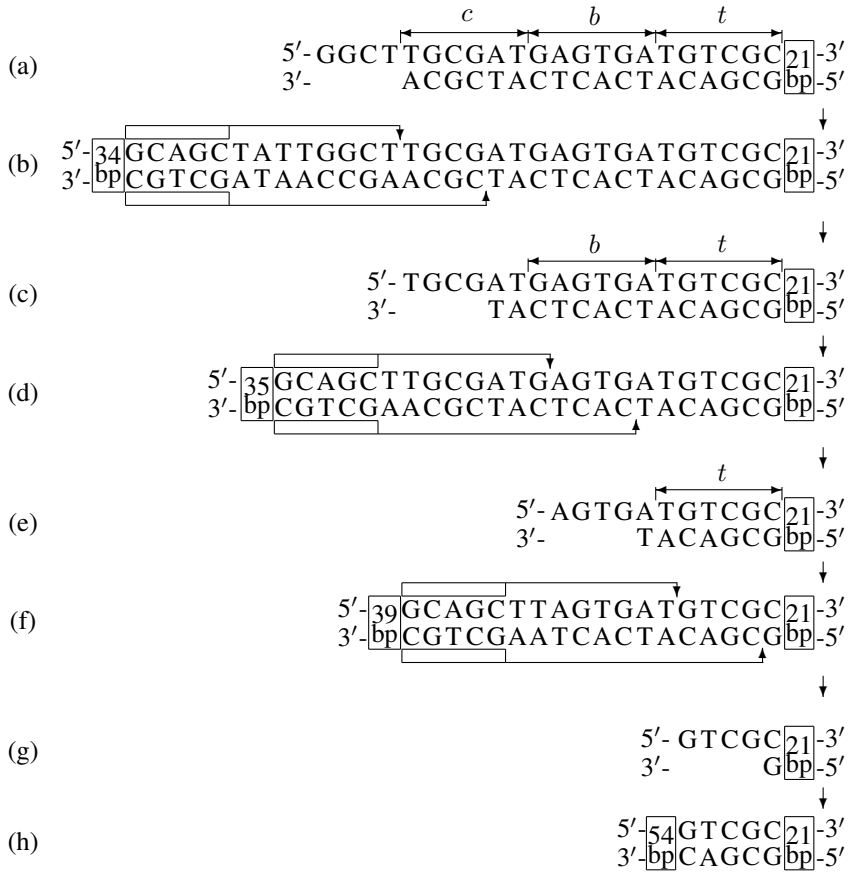

Figure 9: Control serving the reading of symbols of the word acb from the input molecule and obtaining an output molecule in the biomolecular automaton using the enzyme BbvI.

TCGA, TGCA, TATA from the set $\mathcal{A}_0$ of all 4-element sequences of nucleotides.

The appearance of the indicated sixteen words (4-element sequences of nucleotides) causes a biomolecular automaton to malfunction due to the possibility of ligation of a transition molecule with itself – each of the transition molecules exists in multi copies.

Let the transition molecule be given, in which we use the sticky end: CATG (see Fig. 10A). Let us note that this molecule occurs in many copies. Thus, as a result of action of the biomolecular automaton and ligation of one copy of the transition molecule T_{NS_1} (cf. Fig. 10A) with another copy of the same transition molecule there forms the double-stranded fragment of DNA presented in Fig. 10B. In consequence, this causes the number of copies of the molecule T_{NS_1}, to be limited, which can be made use of in further computations carried out by the biomolecular automaton.

So as to prevent the possibility of ligation of copies of the same transition molecule, it is necessary to remove from the set $\mathcal{A}_0$ the words which satisfy the following condition:

$$(*)\ \ x^{-1} = \Xi(x), \text{ where } x \in \mathcal{A}_0.$$

In this way, we reject sixteen words, given earlier, from the set $\mathcal{A}_0$ and as in consequence we obtain the set:

$$\mathcal{A}_1 = \{x : x \in \mathcal{A}_0 \wedge x^{-1} \neq \Xi(x)\} =$$

Figure 10: (A) Transition molecule T_{NS_1} using the sticky end: CATG. (B) Double-stranded fragment of DNA formed as a result of ligation of two copies of the transition molecule presented in (A).

$$\{x : x \in \Delta^+ \wedge |x| = 4 \wedge x^{-1} \neq \Xi(x)\},$$

where the number of the elements of the set $\mathcal{A}_1$ amounts to 240. Availing ourselves of the elements of the set $\mathcal{A}_1$, we form the maximal set $\mathcal{A}_2$ of pairs of the elements in the following manner:

$$\mathcal{A}_2 = \{(x, y) : x, y \in \mathcal{A}_1 \wedge x^{-1} = \Xi(y)\},$$

where the number of the elements of the set $\mathcal{A}_2$ amounts to 240. Then, using the set $\mathcal{A}_2$, we form the set $\mathcal{A}_3$ of pairs in the following way: the set $\mathcal{A}_3$ is the set $\mathcal{A}_2$, from which we removing certain pairs according to the principle of (P). The principle of (P): if the pairs (x, y) and (y, x) belong to the set $\mathcal{A}_2$, then we will remove from the set $\mathcal{A}_2$ a pair whose first element of the pair, comparing the both first el-

Table 2: Part I: 120 pairs (x,y) of four-element sequences of nucleotides.

No	x	y	No	x	y	No	x	y
1	AAAA	TTTT	21	ACCC	GGGT	41	AGTC	GACT
2	AAAC	GTTT	22	ACCG	CGGT	42	AGTG	CACT
3	AAAG	CTTT	23	ACCT	AGGT	43	ATAA	TTAT
4	AAAT	ATTT	24	ACGA	TCGT	44	ATAC	GTAT
5	AACA	TGTT	25	ACGC	GCGT	45	ATAG	CTAT
6	AACC	GGTT	26	ACGG	CCGT	46	ATCA	TGAT
7	AACG	CGTT	27	ACTA	TAGT	47	ATCC	GGAT
8	AACT	AGTT	28	ACTC	GAGT	48	ATCG	CGAT
9	AAGA	TCTT	29	ACTG	CAGT	49	ATGA	TCAT
10	AAGC	GCTT	30	AGAA	TTCT	50	ATGC	GCAT
11	AAGG	CCTT	31	AGAC	GTCT	51	ATGG	CCAT
12	AAGT	ACTT	32	AGAG	CTCT	52	ATTA	TAAT
13	AATA	TATT	33	AGAT	ATCT	53	ATTC	GAAT
14	AATC	GATT	34	AGCA	TGCT	54	ATTG	CAAT
15	AATG	CATT	35	AGCC	GGCT	55	CAAA	TTTG
16	ACAA	TTGT	36	AGCG	CGCT	56	CAAC	GTTG
17	ACAC	GTGT	37	AGGA	TCCT	57	CAAG	CTTG
18	ACAG	CTGT	38	AGGC	GCCT	58	CACA	TGTG
19	ACAT	ATGT	39	AGGG	CCCT	59	CACC	GGTG
20	ACCA	TGGT	40	AGTA	TACT	60	CACG	CGTG

ements of the pairs: (x, y) and (y, x), is lexicographically posterior (see Examp. 1). Tables Tab. 2 and Tab. 3 present 240 elements forming 120 pairs (x, y) of the set $\mathcal{A}_3$, where $x, y \in \mathcal{A}_1$.

Example 1: Let us note that the pairs (AAAA, TTTT), (TTTT, AAAA) $\in \mathcal{A}_2$. The first (element: AAAA) of the first pair (AAAA, TTTT) is lexicographically prior to the first element (element: TTTT) of the second pair (TTTT, AAAA). Thus, the pair (AAAA, TTTT) belongs to the set $\mathcal{A}_3$, and the pair (TTTT, AAAA) does not belong to $\mathcal{A}_3$.

Let us consider pair (x, y)=(AAAA, TTTT) from Tab. 2 (No 1) and two transition molecules in the biomolecular automaton with sticky ends: AAAA and TTTT (see Fig. 11A and Fig. 11B). As a result of ligation of these transition molecules is formed the double-stranded fragment of DNA presented in Fig. 11C. As a consequence, this causes the number of the copies of the molecules T_{NS_2} and T_{NS_3}, to be limited, which may be used in further calculations done by the biomolecular automaton. In connection with this, in the algorithm of generating the maximal number of symbols for a biomolecular automaton using the restriction enzyme BbvI only one element of each of the given 120 pairs of the set $\mathcal{A}_3$ should be used. Selecting individual elements of the successive pairs in this manner, we obtain the family $\mathcal{P}(A_1)$ of maximal sets $B \subset A_1$, such that for each $x, y \in B$ the condition holds,

$$(**)\ x^{-1} \neq \Xi(y).$$

The indicated condition (**) prevents the formation of transition molecules which could ligate with one another during the action of the biomolecular automaton.

In the next part of the algorithm, we will select elements of the family $\mathcal{P}(A_1)$ as sets meant to serve to check the possibilities of generating 40 symbols for the biomolecular automaton using the restriction enzyme BbvI. Thus, let the set C be a chosen element of the family $\mathcal{P}(A_1)$.

In the first part of the stage of deployment and verification, we select a single assignment of 120 words being the elements of the set C, to three sets G_1, G_2 and G_3 (40 words to each set) from among successive possible combinations of assigning the 120 words of the set C to 3 sets consisting of 40 each.

In the second part of the stage of deployment and verification we pre-check whether we are able to form 40 words of length 6 (whether we can create 40 strands in the direction 5′-3′). We examine this by comparing the elements of: first, the sets G_1, G_2 and then G_2, G_3 in the following way:

1. for the sets G_1 and G_2 we check whether the number of occurrences of each word x of length 3, being a suffix in the words of the set G_1, is identical with the

Table 3: Part II: 120 pairs (x,y) of four-element sequences of nucleotides.

No	x	y	No	x	y	No	x	y
61	CAGA	TCTG	81	CGGA	TCCG	101	GCAC	GTGC
62	CAGC	GCTG	82	CGGC	GCCG	102	GCCA	TGGC
63	CAGG	CCTG	83	CGTA	TACG	103	GCCC	GGGC
64	CATA	TATG	84	CGTC	GACG	104	GCGA	TCGC
65	CATC	GATG	85	CTAA	TTAG	105	GCTA	TAGC
66	CCAA	TTGG	86	CTAC	GTAG	106	GGAA	TTCC
67	CCAC	GTGG	87	CTCA	TGAG	107	GGAC	GTCC
68	CCAG	CTGG	88	CTCC	GGAG	108	GGCA	TGCC
69	CCCA	TGGG	89	CTGA	TCAG	109	GGGA	TCCC
70	CCCC	GGGG	90	CTGC	GCAG	110	GGTA	TACC
71	CCCG	CGGG	91	CTTA	TAAG	111	GTAA	TTAC
72	CCGA	TCGG	92	CTTC	GAAG	112	GTCA	TGAC
73	CCGC	GCGG	93	GAAA	TTTC	113	GTGA	TCAC
74	CCTA	TAGG	94	GAAC	GTTC	114	GTTA	TAAC
75	CCTC	GAGG	95	GACA	TGTC	115	TAAA	TTTA
76	CGAA	TTCG	96	GACC	GGTC	116	TACA	TGTA
77	CGAC	GTCG	97	GAGA	TCTC	117	TAGA	TCTA
78	CGAG	CTCG	98	GAGC	GCTC	118	TCAA	TTGA
79	CGCA	TGCG	99	GATA	TATC	119	TCCA	TGGA
80	CGCC	GGCG	100	GCAA	TTGC	120	TGAA	TTCA

(A) 5′-[35]bp GCAGCT -3′
3′-[35]bp CGTCGAAAAA -5′

(B) 5′-[35]bp GCAGCT -3′
3′-[35]bp CGTCGATTTT -5′

(C) 5′-[35]bp GCAGCTTTTTAGCTGC[35]bp -3′
3′-[35]bp CGTCGAAAAATCGACG[35]bp -5′

Figure 11: (A) Transition molecule T_{NS_2} using the sticky end: AAAA. (B) Transition molecule T_{NS_3} using the sticky end: TTTT. (C) Double-stranded fragment of DNA formed as a result of ligation of the two transition molecules presented in (A) and (B).

number of occurrences of the word x as a prefix in the words of the set G_2.

2. for the sets G_2 and G_3 we check whether the number of occurrences of each word x of length 3 being a suffix in the words of the set G_2 is identical with the number of occurrences of the word x as a prefix in the words of the set G_3.

At the stage of generating we introduce the auxiliary set $D = \emptyset$ and examine the possibility of forming 40 words of length 6 (40 strands of symbols in the direction 5′-3′) making use of the elements of the sets G_1, G_2 and G_3, as well as synthesizable concatenations of words of length 3. Each word of length 6 is obtained through a double use of synthesizable concatenations of two words of length 3:

1. we select one word from each of the three sets G_1, G_2 and G_3 in such a way as to make possible concatenation of synthesizable words $x \in G_1$, $y \in G_2$ of length 3 and also to enable concatenation of synthesizable words $y \in G_2$, $z \in G_3$ of length 3.

2. having selected the words $x \in G_1$, $y \in G_2$, $z \in G_3$ which satisfy the above-mentioned condition, we form a word u of length 6 (a strand of symbol in the direction 5′-3′): $u = [[x, y]_3, z]_3$ (symbol assembling, see Fig. 12),

3. the word u obtained upon satisfying the above-presented condition is added to the set D.

In the case where it is impossible to form 40 words (40 words u) from the elements of the sets G_1, G_2 and G_3 in the way given above, we return to checking another possibility of assigning the elements of the set C to the sets G_1, G_2 and G_3. In the case where all the possible assignments of the elements of the set C to the sets G_1, G_2 and G_3, we return to examining the next element of the family $\mathcal{P}(A_1)$. In the case where all the elements of the family

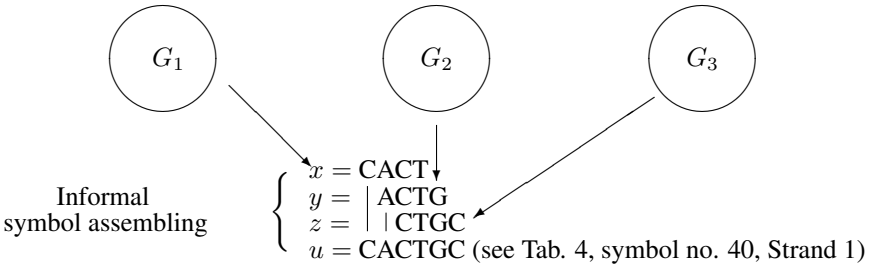

Figure 12: Idea of symbol assembling.

$\mathcal{P}(A_1)$ have been checked and it is impossible to obtain 40 symbols, the algorithm communicates: "Unable to obtain 40 symbols for the biomolecular automaton using the restriction enzyme BbvI".

If the set D has 40 words determined from the elements of the set G_1, G_2 and G_3, we check whether the words of this set (the strands of the symbols in the direction 5′-3′) avoid each of the four, described below, undesired situations, due to the appearance of the sequence recognized by the restriction enzyme BbvI.

The first undesired situation concerns an inclusion of a sequence recognized by the restriction enzyme BbvI inside any symbol. An example to illustrate the above undesired situation is presented in Fig. 13A. Let us note that the second, analogous, undesired situation can occur if a sequence recognized by the restriction enzyme BbvI is included inside any symbol "reversed by 180^o". An example of the latter is shown in Fig. 13B.

(A) 5′-**GCAGC**G-3′ / 3′-CGTCGC-5′ (B) 5′-AGCTGC-3′ / 3′-T**CGACG**-5′

Figure 13: Undesired situations: (A) a sequence recognized by the restriction enzyme BbvI is included in the symbol. (B) a sequence recognized by the restriction enzyme BbvI is contained in the symbol "reversed by 180^o".

The third undesired situation concerns an inclusion of a sequence recognized by the restriction enzyme BbvI in connection of two symbols. An instance illustrating the above-described situation is presented in Fig. 14A. Let us note that the fourth, analogous, undesired situation can occur if a sequence recognized by the restriction enzyme BbvI is included in the connection of two symbols "reversed by 180^o". An example to illustrate the above undesired situation is shown in Fig. 14B.

The appearance of any of the four undesired situations can lead to the occurrence of an undesired action of a biomolecular automaton. In connection with these situations, it is necessary to examine, respectively:

1. whether each word $x \in D$ satisfies the condition: $\sim (e_x \subseteq x)$,
2. whether each word $x \in D$ satisfies the condition: $\sim ((\Xi(e_x))^{-1} \subseteq x)$, where $\sim ((\Xi(e_x))^{-1} \subseteq x)$ means that a sequence recognized by the restriction enzyme BbvI cannot be included in the symbol "reversed by 180^o" and relativized solely to one considered strand in the direction 5′-3′,
3. whether the concatenation $z = xy$ of any words $x \in D$ and $y \in D$ satisfies the condition: $\sim (e_x \subseteq z)$,
4. whether the concatenation $z = xy$ of any words $x \in D$ and $y \in D$ satisfies the condition: $\sim ((\Xi(e_x))^{-1} \subseteq z)$, where $\sim ((\Xi(e_x))^{-1} \subseteq z)$ means that a sequence recognized by the restriction enzyme BbvI cannot be included in the ligation of two symbols "reversed by 180°" and relativized only to one considered strand in the direction 5′-3′.

(A) 5′-TAG**GCAGC**TTAT-3′ / 3′-ATCCGTCGAATA-5′

(B) 5′-TCCGCTGCGGGG-3′ / 3′-AGG**CGACG**CCCC-5′

Figure 14: Undesired situations: (A) a sequence recognized by the restriction enzyme BbvI is included in the ligation of two symbols. (B) a sequence recognized by the restriction enzyme BbvI is included in ligation of two symbols "reversed by 180^o".

In the case one of the four undesired situation is detected, we return to checking another possibility of assigning the elements of the set C to the sets G_1, G_2 and G_3. When all the possible assignments of the elements of the set C to the sets G_1, G_2 and G_3 have been checked, we return to examining another element of the family $\mathcal{P}(A_1)$. In the case all the elements of the family $\mathcal{P}(A_1)$ have been checked and it has been found that it is impossible to obtain 40 symbols that do not include undesired situations, the algorithm returns the message: "Unable to obtain 40 symbols for a biomolecular automaton using the restriction enzyme BbvI". If the set D has 40 words formed from the elements of the sets G_1, G_2, G_3 and there does not occur a single undesired situation, then we move on to the last stage.

At the last stage we determine elements of 40 complementary words for each word of the set D, making use of

Table 4: List of 40 symbols (Strand 1 with its complementary Strand 2) consisting of 6 bp obtained for a biomolecular automaton using restriction enzyme BbvI.

No	Strand 1	Strand 2	No	Strand 1	Strand 2
1	TCGCTA	AGCGAT	21	GCCCGC	CGGGCG
2	CGTTCG	GCAAGC	22	CGCCAG	GCGGTC
3	ATTGAT	TAACTA	23	ATGGGT	TACCCA
4	CGAGTA	GCTCAT	24	GGTAGG	CCATCC
5	CAGGGG	GTCCCC	25	AGGTTA	TCCAAT
6	TAGATA	ATCTAT	26	GCTGTG	CGACAC
7	ATAGTT	TATCAA	27	GTGTAT	CACATA
8	GTTTTG	CAAAAC	28	TATTTA	ATAAAT
9	TTGTTG	AACAAC	29	TTACGA	AATGCT
10	TTGGTG	AACCAC	30	CGACTT	GCTGAA
11	GTGCCG	CACGGC	31	CTTCCG	GAAGGC
12	CTAATG	GATTAC	32	CCGTCT	GGCAGA
13	ATGCGT	TACGCA	33	TCTTAT	AGAATA
14	CGTGAG	GCACTC	34	TATGTC	ATACAG
15	GAGCAA	CTCGTT	35	GTCATC	CAGTAG
16	CAAGCC	GTTCGG	36	ATCGGT	TAGCCA
17	GCCTTT	CGGAAA	37	GGTCCT	CCAGGA
18	TTTCTG	AAAGAC	38	CCTCTC	GGAGAG
19	CTGAAT	GACTTA	39	CTCCAC	GAGGTG
20	AATCCC	TTAGGG	40	CACTGC	GTGACG

the function Ξ of complementarity of words. In this way we acquire pairs of words which mean: a strand of the symbol in the direction $5'$-$3'$ and a strand of the symbol in the direction $3' - 5'$, respectively. On the basis of the presented conception of the algorithm, there were generated 40 words (strands in the direction $5'$-$3'$) denoted as Strand 1 in Tab. 4, as well as 40 words (strands in the direction $3' - 5'$) denoted as Strand 2 in Tab. 4. At the same time, this is giving an answer to the open question asked in 2005: It is possible to generate 40 symbols for a biomolecular automaton using the restriction enzyme BbvI, in which the symbols are coded by means of 6 pairs of nucleotides.

4 Conclusions

The considerations developed in this work aim, on the one hand, to give an answer to the open question posed in the work of Soreni and co-workers (Soreni et al. 2005), relating to the possibility of indicating 40 symbols for a biomolecular automaton which makes use of the restriction enzyme BbvI, in which symbols are coded by means of 6 pairs of nucleotides. On the other hand, they point to the possibility of characterizing the idea of acting of an algorithm which makes it possible to generate 40 symbols for a biomolecular automaton using the restriction enzyme BbvI. Let us note that the open question posed by Soreni and co-workers (Soreni et al. 2005) relating to the possibility of obtaining 40 symbols for a biomolecular automaton using the restriction enzyme BbvI can be generalized in three possible ways: (1) as symbols coded with the use of a different number of pairs of nucleotides, (2) as any other restriction enzyme (used in a biomolecular automaton) and also (3) as the possibility of using more than one restriction enzyme in a biomolecular automaton. Thus, to point to the possibilities of generalization of the question one can start pondering over: (1) the possibility of generating the maximal number of symbols (coded by n pairs of nucleotides) for a biomolecular automaton using one restriction enzyme, (2) the possibility of generating the maximal number of symbols (coded by n pairs of nucleotides) for a biomolecular automaton using more than one restriction enzyme, and also (3) the possibility of an algorithmic approach in each of the two indicated cases. In this way it is possible to raise two general problems which are relativized to the number of restriction enzymes used in a biomolecular automaton and require working out relevant algorithms. Problem 1: generate the maximal number of symbols (coded by n pairs of nucleotides) for a biomolecular automaton using one restriction enzyme. Problem 2: generate the maximal number of symbols (coded by n pairs of nucleotides) for a biomolecular automaton using more than one restriction enzyme. The above-posed problems require considering and defining the conditions which must be imposed on the relations between the restriction enzyme, symbols and other elements which are components of a biomolecular automaton. The output conditions which ought to be considered and taken account of in the above-mentioned relation are the conditions included in the works Krasiński et al. (2013) and Sakowski et al. (2017). Taking into account these conditions will make it possible to determine all the indispensable conditions which serve to elaborate on algorithms enabling to solve the both general problems mentioned above. The solution to the mentioned general problems make it possible to algorithms development for the generating symbols, which are important for laboratory implementation of biomolecular automata.

References

[1] Adleman, L. (1994). Molecular computation of solutions to combinatorial problems. Science, 226, 1021-1024.
https://doi.org/10.1126/science.7973651

[2] Benenson, Y., Paz-Elizur, T., Adar, R., Keinan, E., Livneh, Z., & Shapiro, E. (2001). Programmable and autonomous computing machine made of biomolecules. Nature, 414, 430-434.
https://doi.org/10.1038/35106533

[3] Benenson, Y., Adar, R., Paz-Elizur, T., Livneh, Z., & Shapiro, E. (2003). DNA molecule provides a computing machine with both data and fuel. PNAS, 100, 2191-2196.
https://doi.org/10.1073/pnas.0535624100

[4] Benenson, Y., Gil, B., Ben-Dor, U., Adar, R., Shapiro, E. (2004). An autonomous molecular computer for logical control of gene expression. Nature, 429, 423–429.
https://doi.org/10.1038/nature02551

[5] Bennett, Ch. (1982). The Thermodynamics of computation – a Review. International Journal of Theoretical Physics, 21(12), 905-940.
https://doi.org/10.1007/BF02084158

[6] Bennett, Ch., & Landauer, R. (1985). The fundamental physical limits of computation. Scientific American, 253, 48–56.
https://doi.org/10.1038/scientificamerican0785-48

[7] Chen, P., Jing, L., Jian, Z., Lin, H., Zhizhou, Z. (2007). Differential dependence on DNA ligase of type II restriction enzymes: a practical way toward ligase-free DNA automaton. Biochem. and Bioph. Research Communications, 353, 733-737.
https://doi.org/10.1016/j.bbrc.2006.12.082

[8] Feynman, R. P. (1961). There's plenty of room at the bottom, In D. Gilbert (Ed.) Miniaturization, Reinhol, 282–296.

[9] Gopinath, A., Miyazono, E., Faraon, A., Rothemund, P.W.K. (2016). Engineering and mapping nanocavity emission via precision placement of DNA origami. Nature, 535, 401-405.
https://doi.org/10.1038/nature18287

[10] Krasiński, T., Sakowski, S., Waldmajer, J., Popławski, T. (2013). Arithmetical analysis of biomolecular finite automaton. Fundamenta Informaticae, 128, 463-474.
https://doi.org/10.3233/FI-2013-953

[11] Ran, T., Douek, Y., Milo, L., & Shapiro, E. (2012). A programmable NOR-based device for transcription profile analysis. Scientific reports, 2, 641.
https://doi.org/10.1038/srep00641

[12] Rothemund P. W. K. (1995). DNA and restriction enzyme implementation of Turing machines. Discrete Mathematics and Theoretical Computer Science, 27, 75-120.
https://doi.org/10.1090/dimacs/027/06

[13] Rothemund, P. W., Papadakis, N., & Winfree, E. (2004). Algorithmic self-assembly of DNA Sierpinski triangles. PLoS biology, 2(12), 2041-2053.
https://doi.org/10.1371/journal.pbio.0020424

[14] Rothemund, P.W.K (2006). Folding DNA to Create Nanoscale Shapes and Patterns. Nature, 440, 297-302.
https://doi.org/10.1038/nature04586

[15] Seeman, N. (2001). DNA Nicks and Nodes and Nanotechnology. Nano Letters, 1, 22-26.
https://doi.org/10.1021/nl000182v

[16] Sakowski, S., Krasiński, T., Sarnik, J., Blasiak, J., Waldmajer, J., Poplawski, T. (2017). A detailed experimental study of a DNA computer with two endonucleases. Zeitschrift für Naturforschung C, 72(7-8), 303-313.
https://doi.org/10.1515/znc-2016-0137

[17] Sakowski, S., Krasinski, T., Waldmajer, J., Sarnik, J., Blasiak, J., & Poplawski, T. (2017). Biomolecular computers with multiple restriction enzymes. Genetics and molecular biology, 40(4), 860-870.
https://doi.org/10.1590/1678-4685-gmb-2016-0132

[18] Soreni, M., Yogev, S., Kossoy E., Shoham Y., Keinan E. (2005). Parallel biomolecular computation on surfaces with advanced finite automata. Journal of the American Chemical Society 127, 3935-3943.
https://doi.org/10.1021/ja047168v

[19] Unold, O., Troć, M., Dobosz, T., Trusiewicz, A. (2004). Extended molecular computing model. WSEAS Transactions on Biology and Biomedicine 1, 15-19.

[20] Waldmajer, J., Bonikowski, Z., Sakowski, S. (2019). Theory of tailor automata. Theoretical Computer Science 785, 60-82.
https://doi.org/10.1016/j.tcs.2019.02.002

[21] Whitesides, G. M., Mathias, J. P., & Seto, C. T. (1991). Molecular self-assembly and nanochemistry: a chemical strategy for the synthesis of nanostructures. Science, 254(5036), 1312-1319.
https://doi.org/10.1126/science.1962191