Natural Computing (2006) 5: 127–149
DOI 10.1007/s11047-005-4464-y

The construction of minimal DNA expressions

RUDY VAN VLIET[1,*] HENDRIK JAN HOOGEBOOM[1]
and GRZEGORZ ROZENBERG[1,2]
[1]*Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Niels Bohrweg 1, 2333 CA, Leiden, The Netherlands;* [2]*Department of Computer Science, University of Colorado at Boulder, Boulder, CO, 80309, USA (*Author for correspondence: E-mail: rvvliet@liacs.nl)*

Abstract. We describe a formal language/notation for DNA molecules that may contain nicks and gaps. The elements of the language, DNA expressions, denote formal DNA molecules. Different DNA expressions may denote the same formal DNA molecule. We analyse the shortest DNA expressions denoting a given formal DNA molecule. We determine lower bounds on their lengths and explain how we construct these minimal DNA expressions.

Key words: constructions, DNA molecules, formal language, minimal DNA expressions

1. Introduction

Since the discovery of the structure and function of DNA molecules, DNA has been (and still is) intensively studied by biologists and biochemists. Formal study of computational properties of DNA really began when Head (1987) defined formal languages consisting of strings that can be modified by operations based on the way that restriction enzymes process DNA molecules. The interest of the computer science community in the computational potential of DNA was boosted, when Adleman (1994) reported on an experiment in a biolab that solved a small instance of the directed Hamiltonian path problem using DNA, enzymes and standard biomolecular operations. Since then, research on DNA computing is flourishing, see, e.g., Chen and Reif (2004), Ferretti et al. (2005) and Păun et al. (1998).

However, not much attention is paid in the literature to possible notations for denoting DNA molecules – exceptions are Boneh et al. (1996) and Li (1999). In most cases, one simply uses the standard double-string notation (like $\begin{smallmatrix}\text{ACATG}\\ \text{TGTAC}\end{smallmatrix}$) to describe a double-stranded DNA molecule. A formal notation for RNA is considered in, among others, Rivas and Eddy (2000).

In this paper, we describe a concise and precise notation for DNA molecules, based on the letters A, C, G and T and three operators $\uparrow$, $\downarrow$ and $\updownarrow$ (to be pronounced as *uparrow*, *downarrow* and *updownarrow*, respectively). The resulting DNA expressions denote formal DNA molecules – a formalization of DNA molecules. We do not only account for perfect double-stranded DNA molecules, but also for single-stranded DNA molecules and for double-stranded DNA molecules containing nicks (missing phosphodiester bonds between adjacent nucleotides in the same strand) and gaps (missing nucleotides in one of the strands).

Our three operators bear some resemblance to the operators used in Boneh et al. (1996) and Li (1999), but their functionality is quite different. The operator $\uparrow$ acts as a kind of ligase for the upper strands: it creates upper strands and connects the upper strands of its arguments. The operator $\downarrow$ is the analogue for lower strands. Finally, $\updownarrow$ fills up the gap(s) in its argument. The effects of the operators do not perfectly correspond to the effects of existing techniques in real-life DNA synthesis. Yet, the operators are useful to describe certain types of DNA molecules.

In our formal language, different DNA expressions may denote the same formal DNA molecule. We examine which DNA expressions are minimal, i.e., have the shortest length among all DNA expressions denoting the same formal DNA molecule. First, we determine lower bounds on the length of these DNA expressions. Subsequently, we construct DNA expressions that actually achieve these lower bounds and thus are minimal.

We have published the definitions and the most important results in this paper earlier in Van Vliet et al. (2005). In that paper, we also counted the number of different minimal DNA expressions for a given molecule, and we gave a direct characterization of minimal DNA expressions. However, due to space limitations, we had to omit the proofs of the results.

In the present paper, we concentrate on a few central results. Because of the more restricted scope of this paper, we can elaborate more on the proofs. More formal details can be found in Van Vliet (2004).

2. $\mathcal{N}$-words and formal DNA molecules

We use the alphabet $\mathcal{N} = \{\mathrm{A, C, G, T}\}$ to denote the nucleotides that DNA consists of. The four elements are called *$\mathcal{N}$-letters*. A non-empty string over $\mathcal{N}$ is called an $\mathcal{N}$-word.

For an $\mathcal{N}$-word α, $c(\alpha)$ is the element-wise (non-reversed) Watson–Crick *complement* of α. Thus, more formally, c is a letter-to-letter morphism (also called a *coding*). For example, $c(\mathrm{ACATG}) = \mathrm{TGTAC}$.

The semantical basis of our formal language are *formal DNA molecules*. Formal DNA molecules are strings over the set $\mathcal{A}_{\triangledown\vartriangle} = \mathcal{A}_+ \cup \mathcal{A}_- \cup \mathcal{A}_\pm \cup \{^\triangledown, _\vartriangle\}$, where $\mathcal{A}_+ = \{\binom{\mathrm{A}}{-}, \binom{\mathrm{C}}{-}, \binom{\mathrm{G}}{-}, \binom{\mathrm{T}}{-}\}$, $\mathcal{A}_- = \{\binom{-}{\mathrm{A}}, \binom{-}{\mathrm{C}}, \binom{-}{\mathrm{G}}, \binom{-}{\mathrm{T}}\}$ and $\mathcal{A}_\pm = \{\binom{\mathrm{A}}{\mathrm{T}}, \binom{\mathrm{C}}{\mathrm{G}}, \binom{\mathrm{G}}{\mathrm{C}}, \binom{\mathrm{T}}{\mathrm{A}}\}$. The elements of $\mathcal{A}_+ \cup \mathcal{A}_- \cup \mathcal{A}_\pm$ are called $\mathcal{A}$-letters. The elements of $\mathcal{A}_+$ and $\mathcal{A}_-$ correspond to gaps in the lower strand and the upper strand, respectively. The symbols $^\triangledown$ and $_\vartriangle$ are called *nick letters*. The *upper nick letter* $^\triangledown$ represents a nick in the upper strand of the DNA molecule; the *lower nick letter* $_\vartriangle$ represents a nick in the lower strand.

Not all strings over $\mathcal{A}_{\triangledown\vartriangle}$ are formal DNA molecules. We impose three natural conditions on the strings, which, among others, prevent the DNA molecule represented from 'falling apart.'

DEFINITION 1. *A formal DNA molecule is a string $X = x_1 x_2 \cdots x_r$ with $r \geq 1$ and for $i = 1, \ldots, r$, $x_i \in \mathcal{A}_{\triangledown\vartriangle}$, satisfying*

- *if $x_i \in \mathcal{A}_+$, then $x_{i+1} \notin \mathcal{A}_-$ $\quad (i = 1, 2, \ldots, r-1)$,*
  if $x_i \in \mathcal{A}_-$, then $x_{i+1} \notin \mathcal{A}_+$ $\quad (i = 1, 2, \ldots, r-1)$,
- $x_1, x_r \notin \{^\triangledown, _\vartriangle\}$,
- *if $x_i \in \{^\triangledown, _\vartriangle\}$, then $x_{i-1}, x_{i+1} \in \mathcal{A}_\pm$ $\quad (i = 2, 3, \ldots, r-1)$.*

A formal DNA molecule without nick letters is called *nick free*. Examples of formal DNA molecules are

$$X_1 = \binom{\mathrm{A}}{\mathrm{T}}\binom{\mathrm{C}}{\mathrm{G}}\binom{\mathrm{A}}{\mathrm{T}}\binom{\mathrm{T}}{\mathrm{A}}\binom{\mathrm{G}}{\mathrm{C}}, \tag{1}$$

$$X_2 = \binom{\mathrm{A}}{\mathrm{T}}^\triangledown\binom{\mathrm{C}}{\mathrm{G}}\binom{\mathrm{A}}{\mathrm{T}}_\vartriangle\binom{\mathrm{T}}{\mathrm{A}}\binom{\mathrm{G}}{-}, \text{ and} \tag{2}$$

$$X_3 = \binom{-}{\mathrm{T}}\binom{\mathrm{C}}{\mathrm{G}}\binom{\mathrm{A}}{-}\binom{\mathrm{T}}{-}\binom{\mathrm{G}}{\mathrm{C}}. \tag{3}$$

Both X_1 and X_3 are nick free. We assume that if two nucleotides in the same strand are separated by a gap (as is the case for the G and the C in the lower strand of X_3), then they are not connected by a (long) phosphodiester bond.

Often, we simplify the notation of a formal DNA molecule, by 'concatenating' consecutive symbols of $\mathcal{A}_+$; we then write $\binom{a_i \cdots a_j}{-}$ instead of $\binom{a_i}{-} \cdots \binom{a_j}{-}$. Analogously, we may concatenate consecutive elements of $\mathcal{A}_-$ and consecutive elements of $\mathcal{A}_\pm$. When we simplify the *notation* of a formal DNA molecule, we do not modify the formal DNA molecule itself. In particular, it remains a string over $\mathcal{A}_{\bigtriangledown\bigtriangleup}$.

A non-empty sequence of elements of $\mathcal{A}_+$ is called an *upper $\mathcal{A}$-word*. Analogously, we have a *lower $\mathcal{A}$-word* (with elements of $\mathcal{A}_-$) and a *double $\mathcal{A}$-word* (with elements of $\mathcal{A}_\pm$). An occurrence of an upper $\mathcal{A}$-word in a formal DNA molecule X is called an *upper component* of X, if it is neither (directly) preceded, nor (directly) succeeded by an upper $\mathcal{A}$-word. Hence, the upper $\mathcal{A}$-word cannot be extended. Of course, a *lower component* and a *double component* are defined analogously. In addition to these three types of components, the nick letters occurring in X are components by themselves. An upper component or a lower component may also be called a *single-stranded component* of X.

For a formal DNA molecule X, the *decomposition* of X is defined as the sequence $x'_1, \ldots, x'_k$ with $k \geq 1$, such that $X = x'_1 \cdots x'_k$ and each x'_i is a component of X. For the ease of notation, we will in general write $x'_1 \cdots x'_k$ instead of $x'_1, \ldots, x'_k$.

For example, the decompositions of the formal DNA molecules X_1, X_2 and X_3 are

$$X_1 = \begin{pmatrix}\text{ACATG}\\ \text{TGTAG}\end{pmatrix} \quad (\text{with } k = 1),$$

$$X_2 = \begin{pmatrix}\text{A}\\ \text{T}\end{pmatrix}^{\triangledown}\begin{pmatrix}\text{CA}\\ \text{GT}\end{pmatrix}_{\vartriangle}\begin{pmatrix}\text{T}\\ \text{A}\end{pmatrix}\begin{pmatrix}\text{G}\\ -\end{pmatrix} \quad (\text{with } k = 6), \quad \text{and}$$

$$X_3 = \begin{pmatrix}-\\ \text{T}\end{pmatrix}\begin{pmatrix}\text{C}\\ \text{G}\end{pmatrix}\begin{pmatrix}\text{AT}\\ -\end{pmatrix}\begin{pmatrix}\text{G}\\ \text{C}\end{pmatrix} \quad (\text{with } k = 4).$$

By definition, an upper component of a formal DNA molecule X cannot be succeeded by a lower component and vice versa, and nick letters occurring in X must be preceded and succeeded by a double component. This implies that the decomposition of X is an alternating sequence of double components on the one hand and other types of components on the other hand.

For example, the decomposition of X_2 consists of a double component, an upper nick letter, a double component, a lower nick letter, a double component and an upper component, respectively.

We define three functions on formal DNA molecules. Let $X = x_1 \dots x_r$ for some $r \geq 1$ be a formal DNA molecule. Then $L(X) = x_1$ and $R(X) = x_r$. Hence, the functions L and R give the leftmost symbol and the rightmost symbol of a formal DNA molecule. Further, $|X|_{\mathcal{A}}$ counts the $\mathcal{A}$-letters occurring in X. For example, $L(X_2) = \binom{\mathrm{A}}{\mathrm{T}}$, $R(X_2) = \binom{\mathrm{G}}{-}$ and $|X_2|_{\mathcal{A}} = 5$.

3. DNA expressions

The elements of our language are called *DNA expressions*. The semantics of a DNA expression E is a formal DNA molecule, denoted by $\mathcal{S}(E)$. In this paper, we will describe the syntax and semantics of a DNA expression in words and by means of examples. For a formal definition, we refer to (Van Vliet, 2004).

DNA expressions are the result of applying the three operators $\uparrow$, $\downarrow$ and $\updownarrow$ to basic $\mathcal{N}$-words. In general, the operator $\uparrow$ can have any number $n \geq 1$ arguments $\varepsilon_1, \dots, \varepsilon_n$, which may be $\mathcal{N}$-words or DNA expressions. The result of applying $\uparrow$ to these arguments is the DNA expression $\langle \uparrow \varepsilon_1 \cdots \varepsilon_n \rangle$. It is called an $\uparrow$-*expression*. Analogously, we may have a $\downarrow$-*expression* $\langle \downarrow \varepsilon_1 \cdots \varepsilon_n \rangle$. The operator $\updownarrow$ can have only one argument ε_1, which may again be an $\mathcal{N}$-word or a DNA expression, yielding an $\updownarrow$-*expression* $\langle \updownarrow \varepsilon_1 \rangle$.

Hence, the set of all DNA expressions is a language over the alphabet $\mathcal{N} \cup \{\uparrow, \downarrow, \updownarrow, \langle, \rangle\}$. The length of a specific DNA expression E is defined as the number of its symbols and is denoted by $|E|$. The *outermost operator* of a DNA expression is (the occurrence of) the operator which has been performed last. For example, the outermost operator of an $\uparrow$-expression is $\uparrow$. All other occurrences of operators in a DNA expression, i.e., the occurrences in the argument(s) of the outermost operator, are called *inner occurrences*.

The effect of $\uparrow$ is the following: (1) for each argument that is an $\mathcal{N}$-word α, it produces an upper $\mathcal{A}$-word $\binom{\alpha}{-}$, (2) it removes all upper nick letters occurring in its arguments, and (3) it connects the upper strands of consecutive arguments.

Let us use $\mathcal{S}^+(\varepsilon_i)$ to denote the formal DNA molecule corresponding to an argument ε_i of $\uparrow$. Then by step (1),

$$\mathcal{S}^+(\varepsilon_i) = \begin{cases} \binom{\alpha_i}{-}, & \text{if } \varepsilon_i \text{ is an } \mathcal{N}\text{-word } \alpha_i,\\ \mathcal{S}(\varepsilon_i), & \text{if } \varepsilon_i \text{ is a DNA expression.}\end{cases}$$

Step (3) requires that for $i = 1, \ldots, n-1$, the upper strand of the molecule $X_i = \mathcal{S}^+(\varepsilon_i)$ extends at least as far to the right as the lower strand: $R(X_i)$ must not be an element of $\mathcal{A}_-$. Analogously, if $X_{i+1} = \mathcal{S}^+(\varepsilon_{i+1})$, then $L(X_{i+1})$ must not be an element of $\mathcal{A}_-$. Otherwise, there would be a gap in the upper strand 'between' X_i and X_{i+1}, and we would not be able to connect the upper strands. Such natural requirements are tedious to formalize, which is why we omit a full formal definition of DNA expressions.

Lower nick letters that occur in the arguments of $\uparrow$ are not removed. On the contrary, if both $R(X_i)$ and $L(X_{i+1})$ are elements of $\mathcal{A}_\pm$, then $\uparrow$ introduces a lower nick letter between X_i and X_{i+1}. Thus, the lower strands of consecutive arguments are not connected.

The simplest $\uparrow$-expression is of the form $\langle\uparrow \alpha_1\rangle$ for an $\mathcal{N}$-word α_1. Its semantics is the formal DNA molecule $\binom{\alpha_i}{-}$. Figure 1(a) shows the effect of $\uparrow$ for two less trivial examples. For the ease of understanding, we replaced the arguments of $\uparrow$ that are DNA expressions by pictorial representations of the corresponding DNA molecules. The result of the operator is depicted in the same way. For example, the first $\uparrow$-expression has three arguments: a DNA expression, an $\mathcal{N}$-word and another DNA expression, respectively.

Although the DNA molecules corresponding to $\genfrac{}{}{0pt}{}{\text{ACAT}}{\text{TGT}}$ and $\genfrac{}{}{0pt}{}{\text{G}}{\text{AC}}$ have matching sticky ends, $\langle\uparrow \genfrac{}{}{0pt}{}{\text{ACAT}}{\text{TGT}}\ \genfrac{}{}{0pt}{}{\text{G}}{\text{AC}}\rangle$ is not a DNA expression, because $L\big(\binom{-}{\text{A}}\binom{\text{G}}{\text{C}}\big) \in \mathcal{A}_-$. Hence, the operator $\uparrow$ does not account for annealing. Analogously, $\langle\uparrow \genfrac{}{}{0pt}{}{\text{AC}}{\text{TGT}}\ \genfrac{}{}{0pt}{}{\text{G}}{\text{AC}}\rangle$ is not a DNA expression.

The effect of the operator $\downarrow$ is analogous to that of $\uparrow$. Instead of upper strands, however, it involves lower strands. This is illustrated in Figure 1(b).

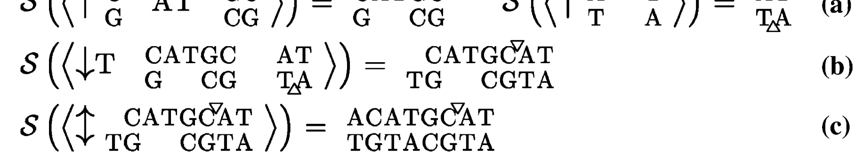


Figure 1. Examples of (a) the effect of the operator $\uparrow$; (b) the effect of the operator $\downarrow$; (c) the effect of the operator $\updownarrow$.

Finally, the operator $\updownarrow$ complements its argument: it provides a complementary nucleotide for every nucleotide that is not yet complemented. Each nucleotide added is connected to its direct neighbours. The operator does not introduce or remove nick letters. The argument of $\updownarrow$ may be any $\mathcal{N}$-word or any DNA expression. If the argument is an $\mathcal{N}$-word α_1, then it is interpreted as $\langle\uparrow \alpha_1\rangle$. Hence, $\mathcal{S}(\langle\updownarrow \alpha_1\rangle) = \binom{\alpha_1}{c(\alpha_1)}$.

Figure 1(c) illustrates the effect of $\updownarrow$. A complete DNA expression denoting the formal DNA molecule from this example is

$$E = \langle\updownarrow \langle\downarrow \mathrm{T}\langle\uparrow \langle\updownarrow \mathrm{C}\rangle\mathrm{AT}\langle\downarrow \langle\updownarrow \mathrm{G}\rangle\langle\updownarrow \mathrm{C}\rangle\rangle\rangle\langle\uparrow \langle\updownarrow \mathrm{A}\rangle\langle\updownarrow \mathrm{T}\rangle\rangle\rangle\rangle. \tag{4}$$

It is the result of the step-by-step construction from Figure 1.

We say that a formal DNA molecule X is *expressible*, if there exists a DNA expression E with $\mathcal{S}(E) = X$. Unfortunately, there exist formal DNA molecules that are not expressible. In fact, we have:

THEOREM 2. *A formal DNA molecule is expressible, if and only if it does not contain both upper nick letters and lower nick letters.*

Hence, a formal DNA molecule is expressible, if and only if either it does not contain any nick letters, or it contains only upper nick letters, or it contains only lower nick letters. Thus, the molecules X_1 and X_3 from (1) and (3) are expressible, because they do not contain any nick letters. Also, the molecules $X_4 = \binom{\mathrm{AC}}{-}\binom{\mathrm{AT}}{\mathrm{TA}}\triangledown\binom{\mathrm{G}}{\mathrm{C}}$ and $X_5 = \binom{-}{\mathrm{T}}\binom{\mathrm{CA}}{\mathrm{GT}}{}_{\triangle}\binom{\mathrm{T}}{\mathrm{A}}\binom{\mathrm{G}}{-}\binom{\mathrm{C}}{\mathrm{G}}{}_{\triangle}\binom{\mathrm{A}}{\mathrm{T}}$ are expressible, because X_4 contains only an upper nick letter and X_5 contains only lower nick letters. However, the molecule X_2 from (2) is not expressible, because it contains both an upper nick letter and a lower nick letter.

We conclude this section with some remarks on the choice of the operators. Obviously, there are many possible choices for operators that can be used to construct DNA expressions. Our choice of operators was based on two considerations: (1) the basic two components of a double-stranded DNA molecule are the two strands, and (2) the operators we consider should obey some notion of locality.

In the case of the operators $\uparrow$ and $\downarrow$, 'locality' means that they act on one of the strands – in particular, $\uparrow$ seals (repairs) the nicks only in the upper strand, while $\downarrow$ seals the nicks only in the lower strand.

Note that applying both $\uparrow$ and $\downarrow$ (in any order) to one argument will seal any existing nick.

In the case of $\updownarrow$, 'locality' means that the string of nucleotides filling in a gap gets also properly connected (bonded) to its neighbours, while the pre-existing nicks are not sealed.

4. The length of a DNA expression

Different DNA expressions may denote the same formal DNA molecule. Such DNA expressions are called *equivalent*. In fact, for each expressible formal DNA molecule X, there exist infinitely many DNA expressions denoting X. For example, it is easily verified that if E is an $\uparrow$-expression denoting X, then so is $\langle\uparrow E\rangle$. We can repeat this construction arbitrarily many times.

We will examine the minimal length of DNA expressions denoting a given formal DNA molecule. For all types of expressible formal DNA molecules, we will also describe the DNA expressions that achieve this length. Before we do so, we make an elementary observation:

LEMMA 3. *Let E be a DNA expression denoting a formal DNA molecule X, and let p be the number of operators occurring in E. Then $|E| = 3 \cdot p + |X|_{\mathcal{A}}$.*

Because each occurrence of an operator is accompanied by an opening bracket and a closing bracket, the term $3 \cdot p$ accounts for the operators and the brackets in E. Consequently, $|X|_{\mathcal{A}}$ counts the $\mathcal{N}$-letters occurring in E. Note that this number only depends on X, and not on the specific DNA expression E. Indeed, for the DNA expression E from (4), the number p of operators is 10, the number of $\mathcal{A}$-letters in $X = \mathcal{S}(E)$ is 8 and $|E| = 3 \cdot 10 + 8 = 38$.

5. Lower bounds for the length of a DNA expression

We first examine lower bounds for the length of a DNA expression E denoting a formal DNA molecule X. These lower bounds will be expressed in terms of some simple counting functions of X. We now introduce these counting functions.

It follows from the definition of a DNA expression that both *upper* components and *lower* nick letters are the result of an occurrence of the operator $\uparrow$. Therefore, such components are called $\uparrow$-*components.* Analogously, lower components and upper nick letters are called $\downarrow$-*components.*

Recall that the decomposition of a formal DNA molecule is an alternating sequence of double components on the one hand and other types of components on the other hand. If we disregard the double components, then we only have a sequence of other types of components, which are $\uparrow$-components and $\downarrow$-components. A (maximal) series of $\uparrow$-components is succeeded by a (maximal) series of $\downarrow$-components, which in turn is succeeded by a (maximal) series of $\uparrow$-components, and so on. We are interested in these maximal series, which we call *primitive* $\uparrow$*-blocks* and *primitive* $\downarrow$*-blocks*, respectively. A formal definition of these blocks also includes the double components occurring in the molecule:

DEFINTION 4. *Let X be a formal DNA molecule and let* $x'_1 \cdots x'_k$ *for some* $k \geq 1$ *be the decomposition of X. A primitive* $\uparrow$*-block of X is a nonempty substring* $X_1 = x'_{a_0} \cdots x'_{a_1}$ *of X, where* $1 \leq a_0 \leq a_1 \leq k$, such that

- X_1 *contains at least one* $\uparrow$*-component,*
- X_1 *does not contain any* $\downarrow$*-component,*
- *– either* $a_0 = 1$ *(hence* X_1 *is a prefix of X),*
  – or $a_0 \geq 2$ *and* x'_{a_0-1} *is a* $\downarrow$*-component,*
- *– either* $a_1 = k$ *(hence* X_1 *is a suffix of X),*
  – or $a_1 \leq k-1$ *and* x'_{a_1+1} *is a* $\downarrow$*-component.*

Note that a primitive $\uparrow$-block starts with the double component preceding the series of $\uparrow$-components (if such a double component exists) and ends with the double component succeeding the series of $\uparrow$-components (again, if such a double component exists). The definition of a primitive $\downarrow$-block is completely analogous.

In Figure 2, we have indicated the primitive $\uparrow$-blocks and the primitive $\downarrow$-blocks of a certain formal DNA molecule X containing upper nick letters. The α_i's occurring in this picture denote the $\mathcal{N}$-words determining the upper, lower and double components of X. For example, the last primitive $\downarrow$-block of X is $X'_3 = \binom{\alpha_{14}}{c(\alpha_{14})} \triangledown \binom{\alpha_{15}}{c(\alpha_{15})} \binom{-}{\alpha_{16}}$

We define three counting functions on formal DNA molecules:

DEFINITION 5. *Let X be a formal DNA molecule.*

- $B_\uparrow(X)$ *is the number of primitive* $\uparrow$*-blocks of X.*
- $B_\downarrow(X)$ *is the number of primitive* $\downarrow$*-blocks of X.*
- $n_\updownarrow(X)$ *is the number of double components of X.*

For the formal DNA molecule X from Figure 2, we have $B_{\uparrow}(X) = 3, B_{\downarrow}(X) = 4$ and $n_{\updownarrow}(X) = 10$.

Note that, unless a formal DNA molecule X is of the form $\binom{\alpha_1}{c(\alpha_1)}$ for an $\mathcal{N}$-word α_1, either $B_{\uparrow}(X) > 0$, or $B_{\downarrow}(X) > 0$ (or both). Because primitive $\uparrow$-blocks and primitive $\downarrow$-blocks alternate in a formal DNA molecule, we have

LEMMA 6. *For each formal DNA molecule X, $B_{\uparrow}(X) - 1 \leq B_{\downarrow}(X) \leq B_{\uparrow}(X) + 1$.*

Because the effects of the operators $\uparrow$ and $\downarrow$ are symmetric, we can also derive symmetrical results for $\uparrow$-expressions and $\downarrow$-expressions. In some later results, we will refer to this symmetry rather than fully stating a symmetrical claim.

When a formal DNA molecule X is denoted by a DNA expression E, the values of the counting functions for X can be related to those for the arguments ε_i of E.

LEMMA 7. *Let E be a DNA expression, and let $X = \mathcal{S}(E)$.*

(1) *If $E = \langle \uparrow \varepsilon_1 \cdots \varepsilon_n \rangle$ for some $n \geq 1$ and $\mathcal{N}$-words and DNA expressions $\varepsilon_1, \ldots, \varepsilon_n$, then let, for $i = 1, \ldots, n, X_i = \mathcal{S}^+(\varepsilon_i)$.*

$$B_{\downarrow}(X) \leq B_{\downarrow}(X_1) + \cdots + B_{\downarrow}(X_n), \tag{5}$$

$$n_{\updownarrow}(X) \leq n_{\updownarrow}(X_1) + \cdots + n_{\updownarrow}(X_n). \tag{6}$$

(2) *If $E = \langle \downarrow \varepsilon_1 \cdots \varepsilon_n \rangle$ for some $n \geq 1$ and $\mathcal{N}$-words and DNA expressions $\varepsilon_1, \ldots, \varepsilon_n$, then ... (symmetric to Claim 1).*

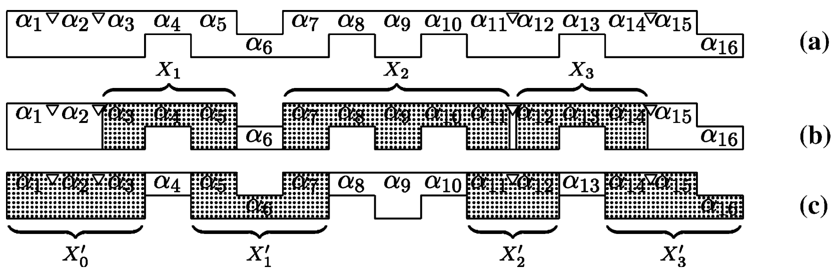

α1 ▿ α2 ▿ α3 α4 α5 α6 α7 α8 α9 α10 α11 ▿ α12 α13 α14 ▿ α15 α16
(a)
X1 X2 X3
(b)
(c)
X'0 X'1 X'2 X'3

Figure 2 Primitive $\uparrow$-blocks and primitive $\downarrow$-blocks. (a) An example formal DNA molecule X that contains (upper) nick letters. (b) The primitive $\uparrow$-blocks of X; note that the upper nick letters are not part of these blocks. (c) The primitive $\downarrow$-blocks of X.

(3) *If* $E = \langle \updownarrow E_1 \rangle$ *for a DNA expression* E_1, *then let* $X_1 = \mathcal{S}(E_1)$.

$$B_{\uparrow}(X) \leq B_{\uparrow}(X_1), \tag{7}$$

$$B_{\downarrow}(X) \leq B_{\downarrow}(X_1), \tag{8}$$

$$n_{\updownarrow}(X) \leq n_{\updownarrow}(X_1) + 1. \tag{9}$$

As an illustration of Claim 1, consider

$$\begin{aligned} E = \langle \uparrow\ & \langle \downarrow \langle \updownarrow \alpha_1 \rangle \langle \uparrow \langle \updownarrow \alpha_2 \rangle \alpha_3 \langle \updownarrow \alpha_4 \rangle \rangle \alpha_5 \langle \updownarrow \alpha_6 \rangle \rangle \\ & \langle \downarrow \langle \updownarrow \alpha_7 \rangle \alpha_8 \langle \updownarrow \alpha_9 \rangle \langle \uparrow \langle \updownarrow \alpha_{10} \rangle \alpha_{11} \rangle \rangle \ \langle \uparrow \alpha_{12} \langle \updownarrow \alpha_{13} \rangle \langle \updownarrow \alpha_{14} \rangle \rangle \rangle. \end{aligned} \tag{10}$$

We have depicted this DNA expression and the resulting formal DNA molecule in Figure 3. In this case, $n = 3$ and

$$\begin{aligned} B_{\downarrow}(X) &= 2 < 2 + 1 + 0, \\ n_{\updownarrow}(X) &= 7 < 4 + 3 + 2. \end{aligned}$$

Proof

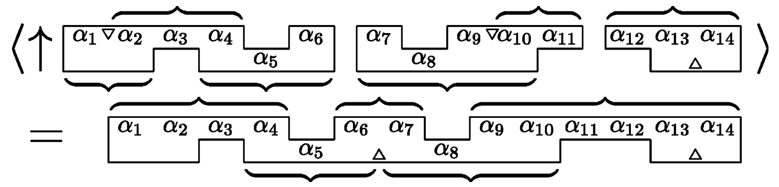
(1) Recall that the operator $\uparrow$ removes the upper nick letters occurring in $X_1, \ldots, X_n$ and connects the upper strands of these molecules. The latter action may introduce lower nick letters.

If all $\downarrow$-components in a primitive $\downarrow$-block of an X_i are upper nick letters, then we lose this block when we remove the upper nick letters. The introduction of a lower nick letter (which is an $\uparrow$-component) has no effect on the primitive $\downarrow$-blocks. This explains inequality (5).

By the removal of an upper nick letter, the two double components preceding and succeeding it merge into one. Hence, we lose one double component. A lower nick letter is introduced between X_i and X_{i+1}, if and only if both the last component of X_i and the first component of X_{i+1} are double components. This implies that the connection of the upper strands of $X_1, \ldots, X_n$ does not affect the number of double components. We conclude that inequality (6) holds.

(3) The operator $\updownarrow$ complements the single-stranded components of X_1.

If all $\uparrow$-components in a primitive $\uparrow$-block of X_1 are upper components, then we lose this block. Moreover, suppose that two primitive $\uparrow$-blocks are separated by a primitive $\downarrow$-block in which all $\downarrow$-components are lower components. When these lower components are complemented, the primitive $\downarrow$-block disappears and the two primitive



$\alpha_1\,\triangledown\,\alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \alpha_7\ \alpha_8\ \alpha_9\,\triangledown\,\alpha_{10}\ \alpha_{11}\ \alpha_{12}\ \alpha_{13}\ \alpha_{14}$
$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \alpha_7\ \alpha_8\ \alpha_9\ \alpha_{10}\ \alpha_{11}\ \alpha_{12}\ \alpha_{13}\ \alpha_{14}$

Figure 3. Pictorial representation of the example $\uparrow$-expression E with $S(E)=X$ for which $B_\downarrow(X)$ and $n_\updownarrow(X)$ are calculated (see equation (10)). The primitive $\uparrow$-blocks and the primitive $\downarrow$-blocks of the formal DNA molecules involved are also indicated.

$\uparrow$-blocks merge into one. Both effects contribute to inequality (7). Analogously, we have inequality (8).

If X_1 consists only of a single-stranded component, i.e., if $n_\updownarrow(X_1)=0$, then X consists only of a double component: $n_\updownarrow(X)=1=n_\updownarrow(X_1)+1$. In all other cases, when a single-stranded component is complemented, it merges with an existing, adjacent double component. This does not increase the number of double components. On the contrary, if a single-stranded component is both preceded and succeeded by a double component, then these double components merge into one and we lose a double component. Hence, inequality (9) is certainly valid. $\square$

We can now formulate lower bounds on the lengths of DNA expressions:

THEOREM 8. *Let E be a DNA expression, and let $X=\mathcal{S}(E)$.*

(1) *If E is an $\uparrow$-expression, then $|E|\geq 3+3\cdot B_\downarrow(X)+3\cdot n_\updownarrow(X)+|X|_{\mathcal{A}}$.*

(2) *If E is a $\downarrow$-expression, then $|E|\geq 3+3\cdot B_\uparrow(X)+3\cdot n_\updownarrow(X)+|X|_{\mathcal{A}}$.*

(3) *If E is an $\updownarrow$-expression, then* $|E|\geq 3\cdot B_\uparrow(X)+3\cdot n_\updownarrow(X)+|X|_{\mathcal{A}}$,
$|E|\geq 3\cdot B_\downarrow(X)+3\cdot n_\updownarrow(X)+|X|_{\mathcal{A}}$.

The term $3+3\cdot B_\downarrow(X)$ occurring in Claim 1, the term $3+3\cdot B_\uparrow(X)$ occurring in Claim 2, and the terms $3\cdot B_\uparrow(X)$ and $3\cdot B_\downarrow(X)$ occurring in Claim 3 correspond to occurrences of the two operators $\uparrow$ and $\downarrow$. The term $3\cdot n_\updownarrow(X)$ occurring in all claims corresponds to occurrences of the operator $\updownarrow$, which are needed to obtain the double components of X. For example, an $\uparrow$-expression denoting a formal DNA molecule X contains at least $(1+B_\downarrow(X))$ occurrences of $\uparrow$ and $\downarrow$ together, and at least $n_\updownarrow(X)$ occurrences of $\updownarrow$.

Proof. By induction on the number p of operators occurring in E.

If $p=1$, then E is $\langle\uparrow \alpha_1\rangle, \langle\downarrow \alpha_1\rangle$ or $\langle\updownarrow \alpha_1\rangle$ for an $\mathcal{N}$-word α_1. For these DNA expressions, the applicable claims hold by definition.

Let $p \geq 1$, and suppose that the claims hold for all DNA expressions containing at most p operators (induction hypothesis). Let E be a DNA expression that contains $p+1$ operators.

• If $E = \langle\uparrow \varepsilon_1 \cdots \varepsilon_n\rangle$ for some $n \geq 1$ and $\mathcal{N}$-words and DNA expressions $\varepsilon_1, \ldots, \varepsilon_n$, then let, for $i = 1, \ldots, n, X_i = \mathcal{S}^+(\varepsilon_i)$. Clearly, each argument ε_i contains at most p operators.
For an $\mathcal{N}$-word ε_i, X_i is an upper $\mathcal{A}$-word and by definition,

$$|\varepsilon_i| = |X_i|_{\mathcal{A}} = 3 \cdot B_{\downarrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}}.$$

For an $\uparrow$-expression ε_i, by the induction hypothesis,

$$\begin{aligned}|\varepsilon_i| &\geq 3 + 3 \cdot B_{\downarrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}} \\ &> 3 \cdot B_{\downarrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}}.\end{aligned}$$

For a $\downarrow$-expression ε_i, by the induction hypothesis and Lemma 6,

$$\begin{aligned}|\varepsilon_i| &\geq 3 + 3 \cdot B_{\uparrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}} \\ &\geq 3 \cdot B_{\downarrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}}.\end{aligned}$$

Finally, for an $\updownarrow$-expression ε_i, by the induction hypothesis

$$|\varepsilon_i| \geq 3 \cdot B_{\downarrow}(X_i) + 3 \cdot n_{\updownarrow}(X_i) + |X_i|_{\mathcal{A}}.$$

Now, by Lemma 7(1),

$$|E| = 3 + |\varepsilon_1| + \cdots + |\varepsilon_n| \geq 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

• If $E = \langle\updownarrow E_1\rangle$ for a DNA expression E_1, then let $X_1 = \mathcal{S}(E_1)$. E_1 contains p operators. As in the case for an $\uparrow$-expression E, by the induction hypothesis,

$$|E_1| \geq 3 \cdot B_{\downarrow}(X_1) + 3 \cdot n_{\updownarrow}(X_1) + |X_1|_{\mathcal{A}}.$$

We can now apply Lemma 7(3):

$$|E| = 3 + |E_1| \geq 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

□

6. Minimal DNA expressions for a nick free molecule

We are not just interested in lower bounds on the lengths of DNA expressions; we also want to *construct* the shortest DNA expressions denoting a given formal DNA molecule. A DNA expression E is called *minimal*, if for every equivalent DNA expression E', $|E'| \geq |E|$.

We first consider nick free formal DNA molecules. These consist only of upper components, lower components and double components. By Theorem 2, each nick free formal DNA molecule is expressible.

For a perfect double-stranded molecule, it is easy to construct a minimal DNA expression:

THEOREM 9. *If* $X = \binom{\alpha_1}{c(\alpha_1)}$ *for an* $\mathcal{N}$*-word* α_1*, then the only minimal DNA expression denoting* X is $E = \langle \updownarrow \alpha_1 \rangle$.

For nick free formal DNA molecules X with at least one single-stranded component, the construction of minimal DNA expressions is more involved. Because the operator $\updownarrow$ complements its argument, there does not exist any $\updownarrow$-expression denoting X. We now focus on the construction of minimal $\uparrow$-expressions. We will sometimes refer to the analogous definitions and constructions needed for minimal $\downarrow$-expressions.

Intuitively, upper components $\binom{\alpha_i}{-}$ of X result when $\uparrow$ has arguments that are $\mathcal{N}$-words α_i, and double components $\binom{\alpha_i}{c(\alpha_i)}$ of X can be produced efficiently by arguments of the form $\langle \updownarrow \alpha_i \rangle$. The lower components of X, however, require a special treatment. For them, we define a generalization of the primitive $\downarrow$-block. We use this to partition X into substrings containing lower components and substrings not containing lower components.

Note that each $\downarrow$-component of a nick free formal DNA molecule is in fact a lower component. Therefore, in the context of nick free formal DNA molecules, we will use the term primitive *lower* block instead of primitive $\downarrow$-block. This should not give rise to confusion.

DEFINITION 10. *Let X be a nick free formal DNA molecule. A lower block of X is a substring of X which both starts and ends with a primitive lower block. A lower block partitioning of X is a sequence* $Y_0, \overline{X}_1, Y_1, \ldots, \overline{X}_r, Y_r$ *for some* $r \geq 0$ *such that*

- $X = Y_0 \overline{X}_1 Y_1 \cdots \overline{X}_r Y_r$, and

- *for* $j = 1, \dots, r, \overline{X}_j$ *is a lower block of* X, and
- *for each primitive lower block* X_1 *of* X, *there is a* j *with* $1 \leq j \leq r$, *such that* X_1 *is contained in* $\overline{X}_j$.

Clearly, if a lower block starts and ends with the same primitive lower block, then it *is* a primitive lower block. This is the case for all three lower blocks in the lower block partitioning depicted in Figure 4(a). In general, however, a lower block may contain more than one primitive lower block. An example is lower block $\overline{X}_1$ in Figure 4(b).

A lower block partitioning of X is a partitioning of X based on (disjoint) lower blocks, which together contain all primitive lower blocks. As each lower component occurs in a primitive lower block of X, the lower blocks also contain all lower components of X. Hence, the substrings Y_j in a lower block partitioning only contain upper components and double components of X.

Usually, we will write $Y_0\overline{X}_1Y_1 \cdots \overline{X}_rY_r$ instead of $Y_0, \overline{X}_1, Y_1, \dots, \overline{X}_r, Y_r$ to describe a lower block partitioning. Of course, an *upper block* and an *upper block partitioning* of a nick free formal DNA molecule are defined analogously.

If the first primitive lower block of a nick free formal DNA molecule X is a prefix of X, then the substring Y_0 occurring in the definition of a lower block partitioning is empty. Analogously, the substring Y_r may be empty. In particular, if X contains at least one lower component, but does not contain any upper component, then it is easily verified that the only primitive lower block of X is X itself. Consequently, the only lower block partitioning of X is Y_0XY_1, where $Y_0 = Y_1 = \lambda$ (the empty string). This is depicted in Figure 5(a).

On the other hand, if X does not contain any lower component, then it certainly does not contain any primitive lower block. Hence,

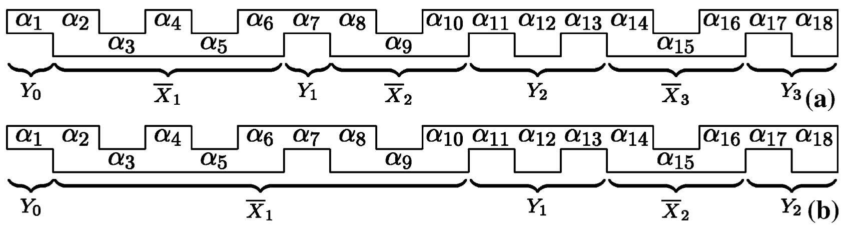

α_1 α_2 α_3 α_4 α_5 α_6 α_7 α_8 α_9 α_{10} α_{11} α_{12} α_{13} α_{14} α_{15} α_{16} α_{17} α_{18}
Y_0 $\overline{X}_1$ Y_1 $\overline{X}_2$ Y_2 $\overline{X}_3$ Y_3 (a)
α_1 α_2 α_3 α_4 α_5 α_6 α_7 α_8 α_9 α_{10} α_{11} α_{12} α_{13} α_{14} α_{15} α_{16} α_{17} α_{18}
Y_0 $\overline{X}_1$ Y_1 $\overline{X}_2$ Y_2 (b)

Figure 4. Two different lower block partitionings of a nick free formal DNA molecule X. (a) The partitioning based on the three primitive lower blocks $\overline{X}_1$, $\overline{X}_2$ and $\overline{X}_3$ of X. (b) A partitioning based on two lower blocks $\overline{X}_1$ and $\overline{X}_2$ of X.

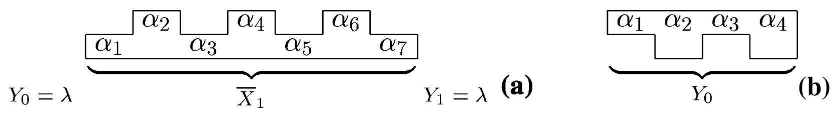


Figure 5. The (unique) lower block partitionings for two special types of nick free formal DNA molecules X. (a) X does not contain any upper component and contains at least one lower component. (b) X does not contain any lower component.

the only lower block partitioning of X is $Y_0 = X$ (i.e., $r = 0$). Figure 5(b) shows an example of this.

We consider the number of primitive upper blocks and primitive lower blocks of a lower block:

LEMMA 11. *Let X be a nick free formal DNA molecule and let $\overline{X}_1$ be a lower block of X. Then $B_\uparrow(\overline{X}_1) = B_\downarrow(\overline{X}_1) - 1$.*

We relate the values of $B_\downarrow$ and $n_\updownarrow$ for a nick free formal DNA molecule X to the values for the substrings in a lower block partitioning of X:

LEMMA 12. *Let X be a nick free formal DNA molecule and let $Y_0\ \overline{X}_1 Y_1 \cdots \overline{X}_r Y_r$ for some $r \geq 0$ be a lower block partitioning of X.*

(1) $B_\downarrow(X) = B_\downarrow(\overline{X}_1) + \cdots + B_\downarrow(\overline{X}_r)$, and for $j = 0, 1, \ldots, r, B_\downarrow(Y_j) = 0$.
(2) $n_\updownarrow(X) = n_\updownarrow(Y_0) + n_\updownarrow(\overline{X}_1) + n_\updownarrow(Y_1) + \cdots + n_\updownarrow(\overline{X}_r) + n_\updownarrow(Y_r)$.

It is instructive to compare the equalities in this result to the inequalities in Lemma 7(1).

We are now ready to construct minimal DNA expressions for nick free formal DNA molecules containing single-stranded components.

THEOREM 13. *Let X be a nick free formal DNA molecule which contains at least one single-stranded component, and let $x'_1 \cdots x'_k$ for some $k \geq 1$ be the decomposition of X.*

(1) *If $B_\uparrow(X) \geq B_\downarrow(X)$, then*
- *let $Y_0\overline{X}_1 Y_1 \ldots \overline{X}_r Y_r$ for some $r \geq 0$ be an arbitrary lower block partitioning of X;*
- *for $j = 1, \ldots, r$, let E_j be an arbitrary minimal DNA expression denoting $\overline{X}_j$;*
- *for $j = 0, 1, \ldots, r$, let $Y_j = x'_{a_j} \ldots x'_{b_j}$ for some $a_j \geq 1$ and $b_j \leq k$;*
- *for $j = 0, 1, \ldots, r$ and for $i = a_j, \ldots, b_j$, let*

$$\varepsilon_i = \begin{cases} \alpha_i & \text{if } x'_i = \binom{\alpha_i}{-} \text{ for an } \mathcal{N}\text{-word } \alpha_i \\ \langle \updownarrow \alpha_i \rangle & \text{if } x'_i = \binom{\alpha_i}{c(\alpha_i)} \text{ for an } \mathcal{N}\text{-word } \alpha_i. \end{cases}$$

Now,

$$E = \langle \uparrow \varepsilon_{a_0} \cdots \varepsilon_{b_0} E_1 \varepsilon_{a_1} \cdots \varepsilon_{b_1} \cdots E_r \varepsilon_{a_r} \cdots \varepsilon_{b_r} \rangle$$

is a minimal $\uparrow$-expression denoting X, and

$$|E| = 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

(2) *If $B_{\downarrow}(X) \geq B_{\uparrow}(X)$, then ... (symmetric to Claim 1).*

Indeed, the minimal DNA expressions E described in this result achieve the applicable lower bounds from Theorem 8. We will see later that the minimal DNA expressions E_j occurring in Claim 1 can be recursively constructed by Claim 2, and vice versa.

Note that if $B_{\uparrow}(X) = B_{\downarrow}(X) \geq 1$, then we can construct both minimal $\uparrow$-expressions and minimal $\downarrow$-expressions denoting X. If on the other hand, $B_{\uparrow}(X) > B_{\downarrow}(X)$, then by Theorem 8(2), each $\downarrow$-expression denoting X is longer than the $\uparrow$-expression described in Claim 1. In that case, each minimal DNA expression is an $\uparrow$-expression. Analogously, if $B_{\downarrow}(X) > B_{\uparrow}(X)$, then each minimal DNA expression denoting X is a $\downarrow$-expression.

All minimal $\uparrow$-expressions and $\downarrow$-expressions denoting a nick free formal DNA molecule can be obtained by the construction from Theorem 13. The result only depends on the lower block partitioning (for a minimal $\uparrow$-expression) or the upper block partitioning (for a minimal $\downarrow$-expression) that we choose. The construction (of one minimal DNA expression) takes a time linear in the length of X.

As an example, we again consider the nick free formal DNA molecule X from Figure 4, for which $B_{\uparrow}(X) = 4, B_{\downarrow}(X) = 3$ and $n_{\updownarrow}(X) = 9$. Because $B_{\uparrow}(X) > B_{\downarrow}(X)$, a minimal DNA expression denoting X must be an $\uparrow$-expression.

We arbitrarily choose the lower block partitioning from Figure 4(b), based on two lower blocks $\overline{X}_1$ and $\overline{X}_2$. We now recursively determine minimal DNA expressions E_1 and E_2 denoting $\overline{X}_1$ and $\overline{X}_2$, respectively. These minimal DNA expressions become arguments of the $\uparrow$-expression E we are constructing, together with

$\mathcal{N}$-words α_i and $\updownarrow$-expressions $\langle\updownarrow\alpha_i\rangle$ for the upper components and double components of X which are in Y_0, Y_1 and Y_2:

$$E = \langle\uparrow\ \alpha_1 E_1 \alpha_{11} \langle\updownarrow\ \alpha_{12}\rangle \alpha_{13} E_2 \alpha_{17} \langle\updownarrow\ \alpha_{18}\rangle\rangle.$$

Because $B_\uparrow(\overline{X}_1) = 1 < 2 = B_\downarrow(\overline{X}_1)$, E_1 must be a $\downarrow$-expression, which is constructed in an analogous way: the only upper block partitioning of $\overline{X}_1$ is $Y_{1,0}\overline{X}_{1,1}Y_{1,1}$, as shown in Figure 6. We determine a minimal DNA expression $E_{1,1}$ for $\overline{X}_{1,1}$, which is, in turn, an $\uparrow$-expression:

$$E_{1,1} = \langle\uparrow\ \langle\updownarrow\ \alpha_6\rangle \alpha_7 \langle\updownarrow\ \alpha_8\rangle\rangle.$$

We then get

$$E_1 = \langle\downarrow\ \langle\updownarrow\ \alpha_2\rangle \alpha_3 \langle\updownarrow\ \alpha_4\rangle \alpha_5 \langle\uparrow\ \langle\updownarrow\ \alpha_6\rangle \alpha_7 \langle\updownarrow\ \alpha_8\rangle\rangle \alpha_9 \langle\updownarrow\ \alpha_{10}\rangle\ \rangle.$$

The minimal DNA expression E_2 is relatively easy to construct:

$$E_2 = \langle\downarrow\ \langle\updownarrow\ \alpha_{14}\rangle \alpha_{15} \langle\updownarrow\ \alpha_{16}\rangle\rangle.$$

Consequently,

$$\begin{aligned} E = \langle\uparrow\ \alpha_1\ \ &\langle\downarrow\ \langle\updownarrow\ \alpha_2\rangle\ \alpha_3\ \langle\updownarrow\ \alpha_4\rangle\ \alpha_5\ \langle\uparrow\ \langle\updownarrow\ \alpha_6\rangle\ \alpha_7\ \langle\updownarrow\ \alpha_8\rangle\rangle\ \alpha_9\ \langle\updownarrow\ \alpha_{10}\rangle\rangle \\ &\alpha_{11}\langle\updownarrow\ \alpha_{12}\rangle\ \alpha_{13}\ \ \langle\downarrow\ \langle\updownarrow\ \alpha_{14}\rangle\ \alpha_{15}\ \langle\updownarrow\ \alpha_{16}\rangle\rangle\ \ \alpha_{17}\ \langle\updownarrow\ \alpha_{18}\rangle\ \rangle. \end{aligned}$$

Indeed,

$$|E| = 39 + |X|_{\mathcal{A}} = 3 + 3 \cdot B_\downarrow(X) + 3 \cdot n_\updownarrow(X) + |X|_{\mathcal{A}}.$$

Proof of Theorem 13: The two claims are proved simultaneously, by induction on the lower of $B_\uparrow(X)$ and $B_\downarrow(X)$. Recall that there does not exist any $\updownarrow$-expression denoting X.

All steps of the proof are analogous for both claims, and we focus on Claim 1. Indeed, for each nick free formal DNA molecule, there exists at least one lower block partitioning. It follows from the definition of a lower block partitioning that the indices a_j and b_j and the arguments ε_i are well defined.

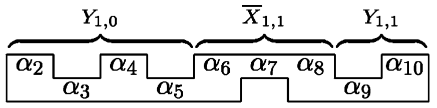


Figure 6. The (unique) upper block partitioning of the lower block $\overline{X}_1$ from Fig. 4(b).

• If $B_{\uparrow}(X) \geq B_{\downarrow}(X) = 0$, then the only lower block partitioning of X is $Y_0 = X$ (see Figure 5(b)). Hence, we do not need any minimal DNA expression E_j. It follows from the definition of the ε_i's that $E = \langle \uparrow \varepsilon_{a_0} \cdots \varepsilon_{b_0} \rangle$ is indeed a DNA expression denoting $Y_0 = X$, and that

$$|E| = 3 + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}} = 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

By Theorem 8(1) and (2), there does not exist a shorter $\uparrow$-expression or $\downarrow$-expression denoting X.

• Let $p \geq 0$, and suppose that for each nick free formal DNA molecule X with at least one single-stranded component and either $B_{\downarrow}(X) \leq p$ or $B_{\uparrow}(X) \leq p$, both claims are valid (induction hypothesis).

Now, let X be a nick free formal DNA molecule for which $B_{\uparrow}(X) \geq B_{\downarrow}(X) = p + 1$, and let $Y_0 \overline{X}_1 Y_1 \cdots \overline{X}_r Y_r$ be an arbitrary lower block partitioning of X.

By Lemma 11, for $j = 1, \ldots, r$, $B_{\uparrow}(\overline{X}_j) = B_{\downarrow}(\overline{X}_j) - 1 \leq B_{\downarrow}(X) - 1 = p$. Now, by the induction hypothesis, we can construct a minimal $\downarrow$-expression E'_j denoting $\overline{X}_j$, for which

$$\begin{aligned} |E'_j| &= 3 + 3 \cdot B_{\uparrow}(\overline{X}_j) + 3 \cdot n_{\updownarrow}(\overline{X}_j) + |\overline{X}_j|_{\mathcal{A}} \\ &= 3 \cdot B_{\downarrow}(\overline{X}_j) + 3 \cdot n_{\updownarrow}(\overline{X}_j) + |\overline{X}_j|_{\mathcal{A}}. \end{aligned}$$

Of course, an arbitrary minimal DNA expression E_j denoting $\overline{X}_j$ has the same length.

We can now prove that E is a DNA expression denoting X. Moreover, for $j = 0, \ldots, r$,

$$\sum_{i=a_j}^{b_j} |\varepsilon_i| = 3 \cdot n_{\updownarrow}(Y_j) + |Y_j|_{\mathcal{A}},$$

and by Lemma 12,

$$\begin{aligned} |E| &= 3 + \sum_{j=0}^{r} \sum_{i=a_j}^{b_j} |\varepsilon_i| + \sum_{j=1}^{r} |E_j| = 3 + \sum_{j=0}^{r} \big(3 \cdot n_{\updownarrow}(Y_j) + |Y_j|_{\mathcal{A}}\big) \\ &\quad + \sum_{j=1}^{r} \big(3 \cdot B_{\downarrow}(\overline{X}_j) + 3 \cdot n_{\updownarrow}(\overline{X}_j) + |\overline{X}_j|_{\mathcal{A}}\big) \\ &= 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}. \end{aligned}$$

By Theorem 8(1) and (2), there does not exist a shorter $\uparrow$-expression or $\downarrow$-expression denoting X. $\square$

7. Minimal DNA expressions for a molecule with nicks

To construct minimal DNA expressions for an expressible formal DNA molecule X containing nick letters, we first decompose X into nick free pieces and nick letters. We call the result the *nick free decomposition* of X. In Figure 7, we have depicted the nick free decomposition $Z_1 \triangle Z_2 \triangle Z_3 \triangle Z_4$ of a formal DNA molecule containing three lower nick letters and no upper nick letters.

A DNA expression E is called *operator-minimal*, if for every equivalent DNA expression E' with the same outermost operator, $|E'| \geq |E|$. For example, consider the formal DNA molecule Z_2, for which $B_\uparrow(Z_2) = 1$, $B_\downarrow(Z_2) = 2$ and $n_\updownarrow(Z_2) = 4$. The $\uparrow$-expression

$$E_2 = \langle \uparrow \langle \downarrow \langle \updownarrow \alpha_5 \rangle \alpha_6 \langle \updownarrow \alpha_7 \rangle \rangle \alpha_8 \langle \downarrow \langle \updownarrow \alpha_9 \rangle \alpha_{10} \langle \updownarrow \alpha_{11} \rangle \rangle \rangle,$$

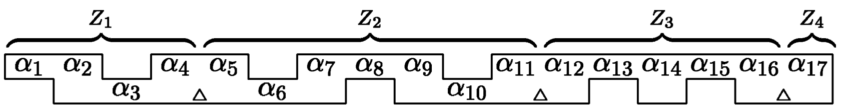
which denotes Z_2 and has length

$$|E_2| = 21 + |Z_2|_{\mathcal{A}} = 3 + 3 \cdot B_\downarrow(Z_2) + 3 \cdot n_\updownarrow(Z_2) + |Z_2|_{\mathcal{A}},$$

is operator-minimal, because by Theorem 8(1), there can be no shorter $\uparrow$-expression denoting Z_2. However, because $B_\downarrow(Z_2) > B_\uparrow(Z_2)$, E_2 is not minimal. As we observed after the statement of Theorem 13, each minimal DNA expression E'_2 denoting Z_2 is a $\downarrow$-expression, which has length

$$|E'_2| = 3 + 3 \cdot B_\uparrow(Z_2) + 3 \cdot n_\updownarrow(Z_2) + |Z_2|_{\mathcal{A}} = 18 + |Z_2|_{\mathcal{A}}.$$

We are in particular interested in operator-minimal $\uparrow$-expressions and $\downarrow$-expressions denoting nick free formal DNA molecules. These operator-minimal DNA expressions appear to be constructed in exactly the same way as the minimal $\uparrow$-expressions and $\downarrow$-expressions for nick free formal DNA molecules, which we have seen in the previous section. In particular, they are also based on lower block partitionings and upper block partitionings. The only difference is that operator-minimal $\uparrow$-expressions and $\downarrow$-expressions can be constructed for *every* nick free formal DNA molecule, and not just for formal DNA molecules X satisfying certain conditions on $B_\uparrow(X)$ and $B_\downarrow(X)$.



Z_1
Z_2
Z_3
Z_4
α_1 α_2 α_3 α_4 α_5 α_6 α_7 α_8 α_9 α_{10} α_{11} α_{12} α_{13} α_{14} α_{15} α_{16} α_{17}
$\triangle$ $\triangle$ $\triangle$

Figure 7. A formal DNA molecule X containing lower nick letters. The nick free decomposition of X is $Z_{1\triangle}\ Z_{2\triangle}\ Z_{3\triangle}\ Z_4$.

We now describe the minimal DNA expressions denoting expressible formal DNA molecules containing nick letters. We only give the formulation for molecules with lower nick letters, as the formulation for the case with upper nick letters is completely analogous. Note that by definition, there do not exist $\downarrow$-expressions that denote a formal DNA molecule containing lower nick letters.

THEOREM 14. *Let X be an expressible formal DNA molecule which contains at least one lower nick letter* $\triangle$, and let $Z_{1\triangle}Z_{2\triangle}\dots{}_{\triangle}Z_m$ *for some $m \geq 2$ be the nick free decomposition of X. For $h = 1, \dots, m$, let E_h be an operator-minimal $\uparrow$-expression denoting Z_h and let the string $\widehat{E}_h$ be the sequence of the arguments of E_h. Then $E = \left\langle \uparrow \widehat{E}_1 \dots \widehat{E}_m \right\rangle$ is a minimal DNA expression denoting X and*

$$|E| = 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

Each minimal DNA expression denoting X is constructed in this way.

We return to the formal DNA molecule X from Figure 7, for which $B_{\downarrow}(X) = 3$ and $n_{\updownarrow}(X) = 10$. We already established the nick free decomposition $Z_{1\triangle}Z_{2\triangle}Z_{3\triangle}Z_4$ of X and considered an operator-minimal $\uparrow$-expression E_2 denoting Z_2. It is not difficult to also construct operator-minimal $\uparrow$-expressions for Z_1, Z_3 and Z_4:

$$\begin{aligned}
E_1 &= \langle \uparrow \alpha_1 \langle \downarrow \langle \updownarrow \alpha_2 \rangle \alpha_3 \langle \updownarrow \alpha_4 \rangle\rangle\rangle, \\
E_3 &= \langle \uparrow \langle \updownarrow \alpha_{12} \rangle \alpha_{13} \langle \updownarrow \alpha_{14} \rangle \alpha_{15} \langle \updownarrow \alpha_{16} \rangle\rangle, \\
E_4 &= \langle \uparrow \langle \updownarrow \alpha_{17} \rangle\rangle.
\end{aligned}$$

The corresponding minimal DNA expression denoting the entire formal DNA molecule X is

$$\begin{aligned}
E = \langle \uparrow\ & \alpha_1\ \langle \downarrow \langle \updownarrow \alpha_2 \rangle\ \alpha_3\ \langle \updownarrow \alpha_4 \rangle\rangle \\
& \langle \downarrow \langle \updownarrow \alpha_5 \rangle\ \alpha_6\ \langle \updownarrow \alpha_7 \rangle\rangle\ \alpha_8\ \langle \downarrow \langle \updownarrow \alpha_9 \rangle\ \alpha_{10}\ \langle \updownarrow \alpha_{11} \rangle\rangle \\
& \langle \updownarrow \alpha_{12} \rangle\ \alpha_{13}\ \langle \updownarrow \alpha_{14} \rangle\ \alpha_{15}\ \langle \updownarrow \alpha_{16} \rangle \quad \langle \updownarrow \alpha_{17} \rangle\ \rangle.
\end{aligned}$$

Indeed,

$$|E| = 42 + |X|_{\mathcal{A}} = 3 + 3 \cdot B_{\downarrow}(X) + 3 \cdot n_{\updownarrow}(X) + |X|_{\mathcal{A}}.$$

Also this construction requires linear time.

8. Conclusions and directions for future research

We have introduced DNA expressions as a formal notation for DNA molecules that may contain nicks and gaps. However, there are (formal) DNA molecules with nicks that cannot be represented by our expressions. We have given lower bounds on the length of a DNA expression denoting a given formal DNA molecule. These lower bounds are expressed in terms of a few simple counting functions. For each expressible formal DNA molecule, we have described (constructions for) the minimal DNA expression(s) denoting it. For nick free molecules with single-stranded components, we have also demonstrated that the resulting DNA expressions are indeed minimal.

Because each expressible formal DNA molecule can be denoted by infinitely many DNA expressions, one may ask for a normal form: a well-defined set of properties such that for each expressible formal DNA molecule X, there is a unique DNA expression denoting X and satisfying those properties. And given a normal form, one may ask for an algorithm that, for each DNA expression, determines the equivalent DNA expression in normal form. We will report on research concerning normal forms in a forthcoming publication.

A possible motivation for considering formal notations for DNA molecules is to provide (a first step towards) a formal calculus for the processing of DNA molecules. Such a calculus would have to contain operators that correspond to various biochemical operations on DNA molecules, as well as expressions for sorts of DNA molecules resulting from applications of these operators. Having such a calculus would be a clear advantage for research in areas such as DNA computing and (parts of) genetic engineering.

From the mathematical point of view, the set of operators acting on expressions denoting DNA molecules does not have to correspond exactly to the biochemical operations. However, one should be able to express such operations by suitable compositions of mathematical operations.

As we have remarked in Section 3, the set of operators that we consider is one of many possible choices. Investigating other notations for DNA molecules based on different motivations (e.g., corresponding to specific biochemical procedures such as PCR) could certainly advance research on formal calculi for DNA processing.

References

Adleman LM (1994) Molecular computation of solutions to combinatorial problems. Science 266: 1021–1024

Boneh D, Dunworth C and Lipton RJ (1996) Breaking DES using a molecular computer. In: Lipton RJ and Baum EB (eds) DNA Based Computers, Proceedings of a DIMACS workshop, April 4, 1995, pp. 37–66. Princeton University, American Mathematical Society, Providence, RI

Chen J and Reif J (eds) (2004) DNA Computing, 9th International workshop on DNA based computers, DNA9, Madison, WI, USA, June 1–3, 2003, Revised Papers, LNCS 2943. Springer-Verlag, Berlin

Ferretti C, Mauri G and Zandron C (eds) (2005) DNA Computing, 10th International workshop on DNA computing, DNA10, Milan, Italy, June 7–10, 2004, Revised Selected Papers, LNCS 3384. Springer-Verlag, Berlin

Head T (1987) Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. Bulletin of Mathematical Biology 49(6): 737–759

Li Z (1999) Algebraic properties of DNA operations. In: Kari L, Rubin H and Wood DH (eds) Proceedings of the fourth international meeting on DNA based computers, University of Pennsylvania, Philadelphia, USA, June 15–19, 1998, BioSystems 52: 55–61

Păun Gh, Rozenberg G and Salomaa A (1998) DNA Computing – New Computing Paradigms. Springer-Verlag, Berlin

Rivas E and Eddy SR (2000) The language of RNA: a formal grammar that includes pseudoknots. Bioinformatics 16(4): 334–340

van Vliet R (2004) Combinatorial aspects of minimal DNA expressions (ext.). Technical Report 2004–03, Leiden Institute of Advanced Computer Science, Leiden University, see `www.liacs.nl/home/rvvliet/mindnaexpr.html`

van Vliet R, Hoogeboom HJ and Rozenberg G (2005) Combinatorial aspects of minimal DNA expressions. In: Ferretti C, Mauri G, Zandron C (eds) DNA Computing, 10th International workshop on DNA computing, DNA10, Milan, Italy, June 7-10, 2004, Revised Selected Papers, LNCS 3384. Springer-Verlag, Berlin, pp. 375–388