The handle http://hdl.handle.net/1887/37052 holds various files of this Leiden University dissertation.

Author: Vliet, Rudy van
Title: DNA expressions : a formal notation for DNA
Issue Date: 2015-12-10
Part I

DNA Expressions in General
Chapter 3

Formal DNA Molecules

Before we define the expressions in our DNA language, we want to be more precise about their meaning – the semantics of the DNA expressions. For this purpose, we formalize the double-word notation for DNA molecules that may contain nicks and gaps.

In this chapter, we introduce and analyse the resulting formal DNA molecules. We first consider \( \mathcal{N} \)-words, non-empty strings over the alphabet of the four different nucleotides. We subsequently define the formal DNA molecules and we identify components of these molecules. We finally discuss some properties, relations and functions of formal DNA molecules.

3.1 \( \mathcal{N} \)-words

Let \( \mathcal{N} = \{ A, C, G, T \} \) be the alphabet of nucleotides, and let the elements of \( \mathcal{N} \) be called \( \mathcal{N} \)-letters. We use the symbol \( a \) (possibly with a subscript) to denote single \( \mathcal{N} \)-letters. A non-empty string over \( \mathcal{N} \) is called an \( \mathcal{N} \)-word. Obviously, the set \( \mathcal{N}^+ \) of \( \mathcal{N} \)-words is closed under concatenation. We reserve the symbol \( \alpha \) (possibly with a subscript) to denote \( \mathcal{N} \)-words.

In general, when \( \alpha \) is an \( \mathcal{N} \)-word (e.g. \( \alpha = ACATG \)), it corresponds to two single-stranded DNA molecules: \( 5'\alpha-3' \) and \( 3'\alpha-5' \). Only if \( \alpha \) happens to be a palindrome (in the linguistic sense), e.g., if \( \alpha = ACTCA \), these two DNA molecules are the same.

The symbol \( c \) denotes the complement function. It is an endomorphism on \( \mathcal{N}^* \), specified by

\[
\begin{align*}
c(A) &= T, & c(C) &= G, & c(G) &= C, & c(T) &= A.
\end{align*}
\]

Thus, for an \( \mathcal{N} \)-word \( \alpha \), \( c(\alpha) \) results by replacing each letter of \( \alpha \) by its Watson-Crick complement. For example, \( c(ACATG) = TGTAC \).

Note that \( c(\alpha) \) itself is not a DNA molecule, with an orientation. It is just an \( \mathcal{N} \)-word, corresponding to two single-stranded DNA molecules with opposite orientations. Of course, for a given \( \mathcal{N} \)-word \( \alpha \), the two strands \( 5'\alpha-3' \) and \( 3'c(\alpha)-5' \) (for example) are each other’s Watson-Crick complement in the molecular sense.

3.2 Definition of formal DNA molecules

In the double-word notation of a perfect, double-stranded DNA molecule, every symbol in the upper word corresponds to a symbol in the lower word. Two such corresponding
symbols denote a base pair—two complementary nucleotides that are connected through a hydrogen bond.

In our formalization of the double-word notation, a base pair is represented by a composite symbol \( x = \left( \frac{x^+}{x^-} \right) \). Here \( x^+ \) stands for the nucleotide in the upper word and \( x^- \) stands for the nucleotide in the lower word.

Since we also want to denote DNA molecules with nicks and gaps, we need to extend this notation. We first consider the gaps. When we have a gap in either of the strands, we still use composite symbols \( \left( \frac{x^+}{x^-} \right) \), but the missing nucleotides are denoted by \( - \). We thus have \( x^+, x^- \in \mathcal{N} \cup \{-\} \). For convenience, we will speak of a base pair also if one of two complementary nucleotides is missing. If both nucleotides are present, we may call the base pair complete.

Of course, the value of \( x^+ \) restricts the value of \( x^- \), and vice versa. Because of the Watson-Crick complementarity and the fact that a missing nucleotide cannot face another missing nucleotide, only twelve of the twenty-five possible composite symbols \( \left( \frac{x^+}{x^-} \right) \) are really allowed: \( \left( \frac{A}{T} \right), \left( \frac{C}{G} \right), \left( \frac{T}{A} \right), \left( \frac{A}{C} \right), \left( \frac{G}{C} \right), \left( \frac{T}{C} \right), \left( \frac{C}{G} \right), \left( \frac{A}{T} \right), \left( \frac{G}{A} \right), \left( \frac{G}{T} \right), \left( \frac{T}{G} \right) \). The set containing these twelve composite symbols is denoted by \( \mathcal{A} \).

For future use, we partition \( \mathcal{A} \) into three subsets: \( \mathcal{A}_\equiv = \left\{ \left( \frac{A}{T} \right), \left( \frac{C}{G} \right), \left( \frac{T}{A} \right), \left( \frac{A}{C} \right), \left( \frac{G}{C} \right), \left( \frac{T}{C} \right) \right\} \) and \( \mathcal{A}_\equiv = \left\{ \left( \frac{G}{A} \right), \left( \frac{G}{T} \right), \left( \frac{T}{G} \right) \right\} \). The elements of \( \mathcal{A} \) are called \( \mathcal{A} \)-letters, the elements of \( \mathcal{A}_\equiv \) are \( \mathcal{A} \)-double letters, the elements of \( \mathcal{A}_+ \) are upper \( \mathcal{A} \)-letters, and the elements of \( \mathcal{A}_- \) are lower \( \mathcal{A} \)-letters. Letters can be used to form strings. A non-empty string over \( \mathcal{A} \) is called an \( \mathcal{A} \)-word. Analogously, we have a \( \mathcal{A} \)-word (with letters from \( \mathcal{A}_\equiv \)), an upper \( \mathcal{A} \)-word (with letters from \( \mathcal{A}_+ \)) and a lower \( \mathcal{A} \)-word (with letters from \( \mathcal{A}_- \)).

We also need symbols to denote nicks in a double word. There are three possibilities for the connection structure of two adjacent base pairs in a double word: there can be a nick in the upper word, there can be a nick in the lower word, or there can be no nick at all between the base pairs. Note that there cannot be both a nick in the upper word and a nick in the lower word between two adjacent base pairs. In such a situation, there would be no connection whatsoever between the base pairs, so they would be parts of different DNA molecules.

The case that there is no nick at all is our default; it is not denoted explicitly. A nick in the upper word is denoted by \( \nabla \) and a nick in the lower word by \( \triangle \). We call \( \nabla \) and \( \triangle \) the nick letters—\( \nabla \) is the upper nick letter, and \( \triangle \) is the lower nick letter.

Now, a complete description of a DNA molecule possibly containing nicks and gaps can be given by a non-empty string \( X \) over \( \mathcal{A}_{\nabla \triangle} = \mathcal{A} \cup \{\nabla, \triangle\} \).

Example 3.1 The DNA molecules depicted in Figure 2.10(a), Figure 2.12(a) and Figure 2.12(b) are denoted by

\[
X_1 = \left( \frac{A}{T} \right) \left( \frac{C}{G} \right) \left( \frac{T}{A} \right) \left( \frac{A}{C} \right),
\]

\[
X_2 = \left( \frac{A}{T} \right) \nabla \left( \frac{C}{G} \right) \left( \frac{T}{A} \right) \triangle \left( \frac{T}{C} \right) \left( \frac{C}{G} \right), \text{ and }
\]

\[
X_3 = \left( \frac{A}{T} \right) \left( \frac{C}{G} \right) \left( \frac{T}{A} \right) \left( \frac{C}{G} \right) \left( \frac{T}{G} \right),
\]

respectively. \( X_1 \) has length 5, \( X_2 \) has length 7, and \( X_3 \) has length 6. 

Not every string over \( \mathcal{A}_{\nabla \triangle} \) represents a DNA molecule. The requirements that strings over \( \mathcal{A}_{\nabla \triangle} \) need to satisfy follow from three observations on DNA molecules:
3.2 Definition of formal DNA molecules

1. To enable at least one phosphodiester bond between adjacent base pairs, a gap in one strand cannot be adjacent to a gap in the other strand (see Figure 3.1(a)).

2. A nick may occur only between two base pairs. In particular, it cannot occur at the left end or the right end of a DNA molecule (see Figure 3.1(b)).

3. Since a nick is a missing phosphodiester bond between two adjacent nucleotides in the same strand, we really need to have nucleotides on both sides of the nick. Moreover, the complementary nucleotides in the other strand must be present and they must be connected by a phosphodiester bond. Hence, a nick may occur only between two complete base pairs (see Figure 3.1(c)).

Now, we are ready to define our formalization of DNA molecules that may contain nicks and gaps:

**Definition 3.2** A formal DNA molecule is a string \( X = x_1x_2 \ldots x_r \) with \( r \geq 1 \) and for \( i = 1, \ldots, r \), \( x_i \in A^\sigma, \) satisfying

1. if \( x_i \in A_+ \), then \( x_{i+1} \notin A_- \) (\( i = 1, 2, \ldots, r - 1 \)),

2. if \( x_i \in A_- \), then \( x_{i+1} \notin A_+ \) (\( i = 1, 2, \ldots, r - 1 \)),

3. \( x_1, x_r \in A \),

4. if \( x_i \in \{\top, \bot\} \), then \( x_{i-1}, x_{i+1} \in A_\pm \) (\( i = 2, 3, \ldots, r - 1 \)).

The language of all formal DNA molecules is denoted by \( F \). Since \( X \in F \) is called a molecule (albeit ‘formal’), we will refer to the sequence of (possibly missing) nucleotides \( x_i^+ \) and upper nick letters in \( X \) as the upper strand of \( X \). The lower strand of \( X \) is defined analogously.

Note, however, that it does not make sense to talk about the upper strand and the lower strand of a (physical) DNA molecule, because a rotation of the molecule by an angle of 180° would change the upper strand into a lower strand and vice versa.

If a formal DNA molecule does not contain upper nick letters (or lower nick letters), then we say that its upper strand (lower strand, respectively) is nick free. If a formal DNA molecule does not contain nick letters at all, then the molecule itself is called nick free.

When we build up a formal DNA molecule from left to right, the choice of a certain letter completely determines the possibilities for the next letter. For example: a nick letter must be succeeded by a double \( A \)-letter; an upper \( A \)-letter may be succeeded by either an
other upper $\mathcal{A}$-letter or a double $\mathcal{A}$-letter, or it may terminate the formal DNA molecule (see Definition 3.2). With this in mind, it is easy to construct a right-linear grammar that generates the language $\mathcal{F}$. We thus have:

**Lemma 3.3** The language $\mathcal{F}$ of formal DNA molecules is regular.

### 3.3 Components of a formal DNA molecule

Let $X = x_1 \ldots x_r$ be a formal DNA molecule, with $x_i \in \mathcal{A}_{\vartriangle}$ for $i = 1, \ldots, r$. A *formal DNA submolecule* of $X$ is a substring $X^s$ of $X$ such that $X^s$ is a formal DNA molecule. It is easy to see that

**Lemma 3.4** A substring $X^s$ of a formal DNA molecule $X$ is a formal DNA molecule if and only if $|X^s| \geq 1$ and $L(X^s), R(X^s) \in \mathcal{A}$.

Hence, $X^s$ should not be empty and neither its first symbol nor its last symbol should be a nick letter.

If a formal DNA submolecule $X^s$ of $X$ is an upper $\mathcal{A}$-word, a lower $\mathcal{A}$-word or a double $\mathcal{A}$-word, and $|X^s| \geq 2$, then it is possible to simplify the notation for $X^s$ and $X$. Let $\alpha = a_1 \ldots a_l$ be an $\mathcal{N}$-word with $a_i \in \mathcal{N}$ ($i = 1, \ldots, l$), and let $X^s$ be a formal DNA submolecule of $X$ with $X^s = x_{i_0} \ldots x_{i_0+l-1}$ for some $i_0$ with $1 \leq i_0 \leq r - l + 1$ (so $|X^s| = l$).

If $X^s = (a_1^\alpha) \cdots (a_l^\alpha)$, then we may write

$$X^s = (a^\alpha) \text{ and } X = x_1 \ldots x_{i_0-1}(a^\alpha)x_{i_0+l} \ldots x_r.$$  

Similarly, if $X^s = (\Bar{a}_1^\alpha) \cdots (\Bar{a}_l^\alpha)$, then we may write

$$X^s = (\Bar{a}^\alpha) \text{ and } X = x_1 \ldots x_{i_0-1}(\Bar{a}^\alpha)x_{i_0+l} \ldots x_r.$$  

Finally, if $X^s = (\under{a}_1^{c(a_1)}) \cdots (\under{a}_l^{c(a_l)})$, then we may write

$$X^s = (\under{a}^{c(a)}) \text{ and } X = x_1 \ldots x_{i_0-1}(\under{a}^{c(a)})x_{i_0+l} \ldots x_r.$$  

By simplifying the notation, we may seem to extend the alphabet of $\mathcal{F}$ with infinitely many symbols $(a^\alpha), (\Bar{a}^\alpha)$ and $(\under{a}^{c(a)})$. We want to emphasize, however, that we actually only simplify the *presentation* of the formal DNA molecules. The formal DNA molecules themselves do not change; they are still strings over the finite alphabet $\mathcal{A}_{\vartriangle}$. In particular, the length of a formal DNA molecule $X = x_1 \ldots x_r$ with $x_i \in \mathcal{A}_{\vartriangle}$ for $i = 1, \ldots, r$ remains $r$, even if $X$ is written in a simplified notation.

**Definition 3.5** Let $X$ be a formal DNA molecule. Then the decomposition of $X$ is the sequence $x'_1, \ldots, x'_k$ of $k \geq 1$ non-empty strings over $\mathcal{A}_{\vartriangle}$ such that

- $X = x'_1 \ldots x'_k$,
- for $i = 1, \ldots, k$, $x'_i$ is either an upper $\mathcal{A}$-word, or a lower $\mathcal{A}$-word, or a double $\mathcal{A}$-word, or a nick letter, and
- for $i = 1, \ldots, k - 1$, if $x'_i$ is an upper $\mathcal{A}$-word, then $x'_{i+1}$ is not an upper $\mathcal{A}$-word, and similarly for lower $\mathcal{A}$-words and double $\mathcal{A}$-words.
### 3.3 Components of a formal DNA molecule

Components can be divided into double components and non-double components, non-double components can in turn be divided into single-stranded components and nick letters, etcetera.

Hence, the decomposition of $X$ cannot be simplified any further by concatenating consecutive elements of the same type. For the ease of notation, we will in general omit the commas and write $x'_1 \ldots x'_k$ instead of $x'_1, \ldots, x'_k$.

**Example 3.6** The decompositions of the formal DNA molecules from Example 3.1 (denoting the molecules shown in Figure 2.10, Figure 2.12(a) and Figure 2.12(b)) are

$$
X_1 = \left( \begin{array}{c}
\text{ACATG} \\
\text{TGTAC}
\end{array} \right),
$$

$$
X_2 = \left( \begin{array}{c}
\text{A} \\
\text{T}
\end{array} \right) \triangle \left( \begin{array}{c}
\text{CA} \\
\text{GT}
\end{array} \right) \triangle \left( \begin{array}{c}
\text{TG} \\
\text{AC}
\end{array} \right)
$$

and

$$
X_3 = \left( \begin{array}{c}
\text{A} \\
\text{T}
\end{array} \right) \left( \begin{array}{c}
\text{C} \\
\text{TA}
\end{array} \right) \left( \begin{array}{c}
\text{T} \\
\text{CG}
\end{array} \right),
$$

respectively.

If $x'_1 \ldots x'_k$ for some $k \geq 1$ is the decomposition of a formal DNA molecule $X$, then the substrings $x'_i$ are called the *components* of $X$. For $i = 1, \ldots, k$, if $x'_i$ is an upper $\mathcal{A}$-word (lower $\mathcal{A}$-word or double $\mathcal{A}$-word), then $x'_i$ is called an *upper component* (*lower component* or *double component*, respectively) of $X$. If $x'_i$ is not a double component, then we may also call it a *non-double component* of $X$. Upper components and lower components of $X$ are also called *single-stranded components* of $X$. The (rooted) tree in Figure 3.2 shows the relations between the different types of components.

Often, we will use pictures as the one in Figure 3.3 to depict a formal DNA molecule. For example, the $\mathcal{N}$-word $\alpha_3$ in this picture represents the lower component $\left( \begin{array}{c}
\alpha_3 \\
\text{CG}
\end{array} \right)$, the $\mathcal{N}$-word $\alpha_5$ represents the upper component $\left( \begin{array}{c}
\alpha_5 \\
\text{AC}
\end{array} \right)$, the $\mathcal{N}$-word $\alpha_6$ represents the double $\mathcal{A}$-word $\left( \begin{array}{c}
\alpha_6 \\
\text{CT}
\end{array} \right)$ (which is not a component!), the first occurrence of the symbol $\triangle$ represents...
Let $X$ be a formal DNA molecule and let $x'_1 \ldots x'_k$ for some $k \geq 1$ be the decomposition of $X$. Then

- for $i = 1, \ldots, k-1$, if $x'_i$ is a non-double component, then $x'_{i+1}$ is a double component;
- for $i = 1, \ldots, k-1$, if $x'_i$ is a double component, then $x'_{i+1}$ is a non-double component.

Indeed, the formal DNA molecule depicted in Figure 3.3 consists of a double component $\left(\alpha_1\right)_{c(\alpha_1)}$, a lower nick letter, a double component $\left(\alpha_2\right)_{c(\alpha_2)}$, a lower component $\left(\alpha_3\right)_c$, a double component $\left(\alpha_4\right)_{c(\alpha_4)}$, an upper component $\left(\alpha_5\right)_c$, a double component $\left(\alpha_6\right)_{c(\alpha_6)}$, an upper component $\left(\alpha_7\right)_c$, a double component $\left(\alpha_8\right)_{c(\alpha_8)}$, a lower nick letter, a double component $\left(\alpha_9\right)_{c(\alpha_9)}$, an upper component $\left(\alpha_{10}\right)_c$, a double component $\left(\alpha_{11}\right)_{c(\alpha_{11})}$, an upper nick letter, a double component $\left(\alpha_{12}\right)_{c(\alpha_{12})}$, an upper nick letter, a double component $\left(\alpha_{13}\right)_{c(\alpha_{13})}$ and a lower component $\left(\alpha_{14}\alpha_{15}\right)_c$.

**Proof:** If for some $i$ with $1 \leq i \leq k-1$, $x'_i$ is a double component, then by the definition of the decomposition, the next component $x'_{i+1}$ is a non-double component. Because nick letters can only occur between two double components and because an upper component cannot occur next to a lower component (see Definition 3.2), the reverse is also true: if for some $i$ with $1 \leq i \leq k-1$, $x'_i$ is an upper component, a lower component or a nick letter, then the next component $x'_{i+1}$ is a double component.

Two special cases of this result will also turn out to be useful:

**Corollary 3.8** Let $X$ be a nick free formal DNA molecule and let $x'_1 \ldots x'_k$ for some $k \geq 1$ be the decomposition of $X$.

1. For $i = 1, \ldots, k$, $x'_i$ is either an upper component, or a lower component, or a double component.
2. For $i = 1, \ldots, k-1$,
   - if $x'_i$ is a single-stranded component, then $x'_{i+1}$ is a double component, and
   - if $x'_i$ is a double component, then $x'_{i+1}$ is a single-stranded component.

When we observe that by definition the first and the last component of a formal DNA molecule cannot be nick letters, we also find
3.4 Properties, relations and functions of formal DNA molecules

Corollary 3.9 Let $X$ be a formal DNA molecule which does not contain any single-stranded component, and let $x'_1 \ldots x'_k$ for some $k \geq 1$ be the decomposition of $X$.

1. For $i = 1, \ldots, k$, $x'_i$ is either a double component, or an upper nick letter, or a lower nick letter.

2. $k = 2m - 1$ for some $m \geq 1$ (hence, $k$ is odd) and

$$X = \left( \frac{\alpha_1}{c(\alpha_1)} \right) y_1 \left( \frac{\alpha_2}{c(\alpha_2)} \right) y_2 \ldots y_{m-1} \left( \frac{\alpha_m}{c(\alpha_m)} \right)$$

for $N$-words $\alpha_1, \ldots, \alpha_m$ and nick letters $y_1, \ldots, y_{m-1}$.

3.4 Properties, relations and functions of formal DNA molecules

In this section, we introduce some properties of formal DNA molecules, relations between formal DNA molecules and functions on formal DNA molecules. We need these to be able to define the syntax and semantics of DNA expressions in Chapter 4.

Properties

Let $X = x_1 \ldots x_r$ be a formal DNA molecule, with $x_i \in A_{\supset\land}$ for $i = 1, \ldots, r$. Then the upper strand of $X$ is said to cover the lower strand to the right if $R(X) = x_r \not\in A_-$, hence, if $x_r^+ \neq -$; note that, since $x_r$ is not allowed to be a nick letter (Condition 2 of Definition 3.2), $x_r^+$ is well defined. Intuitively, in this case, the upper strand extends at least as far to the right as the lower strand.

If $R(X) = x_r \in A_-$, hence $x_r^- = -$ (the upper strand extends even beyond the lower strand to the right), then the upper strand strictly covers the lower strand to the right. In an analogous way we can define ‘(strict) covering to the left’.

Of course, the definition of ‘(strict) covering’ can also be formulated for the lower strand. For example, in the formal DNA molecule $X_3$ from Example 3.1, the lower strand strictly covers the upper strand to the right. Here, the strands extend equally far to the left, and so we say that the upper strand covers the lower strand to the left and vice versa.

Relations

We say that a formal DNA molecule $X_1$ prefits a formal DNA molecule $X_2$ by upper strands, denoted by $X_1 \sqsubseteq X_2$, if the upper strand of $X_1$ covers the lower strand to the right and the upper strand of $X_2$ covers the lower strand to the left, hence, if $R(X_1) \notin A_-$ and $L(X_2) \notin A_-$. Intuitively, when we write $X_1$ and $X_2$ after each other in such a case, the respective upper strands ‘make contact’.

Analogously, we define $X_1$ to prefit $X_2$ by lower strands if $R(X_1) \notin A_+$ and $L(X_2) \notin A_+$, and write then $X_1 \sqsubseteq X_2$. If either $X_1 \sqsubseteq X_2$ or $X_1 \sqsubseteq X_2$, then we may also say that $X_1$ prefits $X_2$, and write $X_1 \sqsubseteq X_2$.

If the order of the formal DNA molecules is clear, then we may also say that $X_1$ and $X_2$ fit together (by upper/lower strands).

In fact, we used the notion of prefitting already in the definition of a (single) formal DNA molecule $X$. When we demanded that an element of $A_+$ cannot be succeeded or
Figure 3.4: Schematic representation of two formal DNA molecules $X_1$ and $X_2$ such that the concatenation $X_1X_2$ is not a formal DNA molecule.

preceded by an element of $\mathcal{A}_-$ (Condition 1 in Definition 3.2), we actually demanded that the formal DNA molecule $(x_i^{+1})$ (of length 1) should prefit the formal DNA molecule $(x_i^{-1})$ (for each $i$ such that neither $x_i$ nor $x_{i+1}$ is a nick letter).

Unlike the set of all $\mathcal{N}$-words $\mathcal{N}^+$, the set of formal DNA molecules $\mathcal{F}$ is not closed under concatenation. Let, for example, $X_1$ and $X_2$ be formal DNA molecules, such that the upper strand of $X_1$ strictly covers the lower strand to the right and the lower strand of $X_2$ strictly covers the upper strand to the left. Then the concatenation $X_1X_2$ is not a formal DNA molecule, because Condition 1 of Definition 3.2 is violated for $i = |X_1|$.

This is illustrated in Figure 3.4. Thus in particular, even if $X_1 = (A)(G)(A)(T)(G)$ and $X_2 = (\bar{A})(\bar{C})(C)(G)(A)(T)(T)$ (so that the respective sticky ends of the DNA molecules form a perfect match), then $X_1X_2$ is not a formal DNA molecule. As a matter of fact, the following property holds:

**Lemma 3.10** The concatenation of two formal DNA molecules $X_1$ and $X_2$ is again a formal DNA molecule, if and only if $X_1 \sqsubseteq X_2$.

**Functions**

We define four endomorphisms on the set $\mathcal{A}_{\nabla,\Delta}^+$. $\nu^+$, $\nu^-$, $\nu$ and $\kappa$. Let $x \in \mathcal{A}_{\nabla,\Delta}^-$. Then the functions are defined by

$$
\nu^+(x) = \begin{cases} 
  x & \text{if } x \in \mathcal{A} \cup \{\nabla\} \\
  \lambda & \text{if } x = \nabla 
\end{cases} \quad (3.1)
$$

$$
\nu^-(x) = \begin{cases} 
  x & \text{if } x \in \mathcal{A} \cup \{\nabla\} \\
  \lambda & \text{if } x = \nabla 
\end{cases} \quad (3.2)
$$

$$
\nu(x) = \begin{cases} 
  x & \text{if } x \in \mathcal{A} \\
  \lambda & \text{if } x \in \{\nabla, \Delta\} 
\end{cases} \quad (3.3)
$$

$$
\kappa(x) = \begin{cases} 
  x & \text{if } x \in \mathcal{A}_+ \cup \{\nabla, \Delta\} \\
  \left(\frac{a}{c(a)}\right) & \text{if } x = \left(\frac{a}{c(a)}\right) \text{ for } a \in \mathcal{N} \\
  \left(\frac{a}{c(a)}\right) & \text{if } x = \left(\frac{a}{c(a)}\right) \text{ for } a \in \mathcal{N} 
\end{cases} \quad (3.4)
$$

Thus, $\nu^+$ removes all upper nick letters from its argument, $\nu^-$ removes all lower nick letters from its argument, $\nu$ removes both the upper nick letters and the lower nick letters from its argument, and $\kappa$ replaces every symbol from $\mathcal{A}_+$ and $\mathcal{A}_-$ in its argument by the corresponding symbol from $\mathcal{A}_+$.

From the point of view of the molecules represented, $\nu^+$ replaces all nicks in the upper strand of its argument by phosphodiester bonds, and $\nu^-$ does the same for nicks in the
lower strand of its argument. The function \( \nu \) replaces all nicks in both the upper strand and the lower strand by phosphodiester bonds. Finally, \( \kappa \) provides a complementary nucleotide for every nucleotide in its argument which is not complemented yet. The function does not introduce nicks, i.e., the nucleotides added get connected to their respective neighbours. On the other hand, the nicks present in the argument are not removed by \( \kappa \).

It is easy to see (by inspecting the effect of the functions on the symbols from \( A_{\land \lor} \)), that the composition of functions from the set \( \{ \nu^+, \nu^-, \nu, \kappa \} \) is commutative, i.e.,

\[
\begin{align*}
  h_2(h_1(X)) &= h_1(h_2(X)) \quad \text{for all } h_1, h_2 \in \{ \nu^+, \nu^-, \nu, \kappa \} \text{ and } X \in A_{\land \lor}^*. 
\end{align*}
\]

(3.5) For example, \( \kappa(\nu^+(X)) = \nu^+(\kappa(X)) \) for each \( X \in A_{\land \lor}^* \).

Further, the functions are idempotent. That is, applying the same function more than one time, does not change the result:

\[
\begin{align*}
  h(h(X)) &= h(X) \quad \text{for each } h \in \{ \nu^+, \nu^-, \nu, \kappa \} \text{ and } X \in A_{\land \lor}^*. 
\end{align*}
\]

(3.6) For example, \( \nu(\nu(X)) = \nu(X) \) for each \( X \in A_{\land \lor}^* \).

Finally, one can verify that \( \nu^- (\nu^+(X)) = \nu(X) \) for each \( X \in A_{\land \lor}^* \).

Hence, \( \nu \) is equal to the composition of \( \nu^+ \) and \( \nu^- \) (and, by commutativity, \( \nu \) is equal to the composition of \( \nu^- \) and \( \nu^+ \)).

Because \( \mathcal{F} \), the set of formal DNA molecules, is a subset of \( A_{\land \lor}^* \), \( \nu^+, \nu^-, \nu \) and \( \kappa \) can be applied to \( \mathcal{F} \). It is easy to verify that for each \( X \in \mathcal{F} \) and \( h \in \{ \nu^+, \nu^-, \nu, \kappa \} \), also \( h(X) \in \mathcal{F} \). For example, because of Condition 3 of Definition 3.2, every nick letter in \( X \) is both preceded and succeeded by an element of \( A_{\pm} \). When such a nick letter is removed from \( X \), by either \( \nu^+, \nu^- \) or \( \nu \), these elements of \( A_{\pm} \) become adjacent and this does not violate any condition of Definition 3.2.

Since we are really interested in \( \mathcal{F} \), we will consider the restriction of the functions \( \nu^+, \nu^-, \nu \) and \( \kappa \) to this subdomain. In order not to burden our notation too much, we will still use the notation \( \nu^+, \nu^-, \nu \) and \( \kappa \), respectively for these restricted functions, instead of \( \nu^+|_{\mathcal{F}}, \nu^-|_{\mathcal{F}}, \nu \) and \( \kappa \) – this should, however, not lead to confusion.

For the composition of functions from \( \{ \nu^+, \nu^-, \nu, \kappa \} \) with the functions \( L \) and \( R \) we have the following results (they follow directly from the definitions of \( L, R, \nu^+, \nu^-, \nu \) and \( \kappa \) and the definition of a formal DNA molecule):

**Lemma 3.11** For each \( X \in \mathcal{F} \),

\[
\begin{align*}
  L(\nu^+(X)) &= L(\nu^-(X)) = L(\nu(X)) = L(X), \\
  R(\nu^+(X)) &= R(\nu^-(X)) = R(\nu(X)) = R(X), \\
  L(\kappa(X)), R(\kappa(X)) &\in A_{\pm}.
\end{align*}
\]