The Thirteenth International Symposium on Artificial Life and Robotics 2008(AROB 13th '08),
B-Con Plaza, Beppu, Oita, Japan, January 31-February 2, 2008

Exploring regular reversibility in Watson-Crick finite automata*

J.M. Sempere

Dept. de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia (Spain)
(Tel : + 34 96 387 73 53; Fax :+34 96 387 73 59)
(Email address: jsempere@dsic.upv.es)

Abstract: Watson-Crick finite automata were inspired by formal language theory, finite states machines and some ingredients from DNA computing such as working with molecules as double stranded complementary strings. Here, we define different kinds of regular reversibility in this model. Mainly, we will explore regular reversibility in the upper (lower) strand and in the double strand.

Keywords: Models of DNA computing, formal languages, Watson-Crick finite automata, regular reversibility.

I. INTRODUCTION

Watson-Crick finite automaton (WKFA) [1] is a good example of how DNA biological properties can be adapted to propose computation models in the framework of DNA computing. A recent survey on WKFA has been published in [2]. The WKFA model works with double strings inspired by double-stranded molecules with a complementary relation between symbols (here, inspired by classical complementary relation between nucleotides A-T and C-G). Different restrictions over the model have been proposed, mainly devoted to restrict the number of final states (i.e., *all final* and *stateless* WKFA) and the way of processing the upper and lower string (i.e., 1-*limited* and *simple* WKFA). Here we propose a different characterization of the model based on a classical concept of formal language theory such as regular reversibility.

Reversible languages were introduced by Angluin [3] as a subclass of regular languages. In her work, Angluin showed that they form an infinite hierarchy (namely, *k*-reversible languages are *k*+1- reversible ones) and she proposed an efficient method to identify any *k*-reversible language from samples of it.

Here, we will introduce regular reversibility in different ways. First, we will introduce a representation theorem for languages accepted by WKFA, which allows us to study WKFA through linear and even linear languages. Then, we will study two possibilities of defining reversibility: in the upper (lower) strand and in the double strand. Finally, we will give some guidelines for future works.

II. BASIC CONCEPTS AND NOTATION

In this section we will introduce basic concepts from formal language theory according to [4] and [5] and from DNA computing according to [6].

1. Formal Language Theory

An alphabet Σ is a finite nonempty set of elements named symbols. A string defined over Σ is a finite ordered sequence of symbols from Σ. The infinite set of all the strings defined over Σ will be denoted by Σ^*. Given a string $x \in \Sigma^*$ we will denote its length by $|x|$. The empty string will be denoted by λ and Σ^+ will denote $\Sigma^*-\{\lambda\}$. Given a string x we will denote by x^r the reversal string of x. A language L defined over Σ is a set of strings from Σ.

A grammar is a construct $G = (N,\Sigma,P,S)$ where N and Σ are the alphabets of auxiliary and terminal symbols with $N \cap \Sigma = \varnothing$, $S \in N$ is the *axiom* of the grammar and P is a finite set of productions in the form $\alpha \rightarrow \beta$. The language of the grammar is denoted by L(G) and it is the set of terminal strings that can be obtained from S by applying symbol substitutions according to P. Formally, $w_1 \Rightarrow_G w_2$ if $w_1 = u\alpha v$, $w_2 = u\beta v$ and $\alpha \rightarrow \beta \in P$. We will denote by $\Rightarrow^*_G$ the reflexive and transitive closure of $\Rightarrow_G$.

We will say that a grammar $G = (N,\Sigma,P,S)$ is *right linear* (regular) if every production in P is in the form $A \rightarrow uB$ or $A \rightarrow w$ with $A,B \in N$ and $u,w \in \Sigma^*$. The class of languages generated by right linear grammars coincides with the class of regular languages and will be denoted by REG. We will say that a grammar $G = (N,\Sigma,P,S)$ is *linear* if every production in P is in the form $A \rightarrow uBv$ or $A \rightarrow w$ with $A,B \in N$ and $u,v,w \in \Sigma^*$. The class of languages

* Work partially supported by the Spanish Ministry of Education and Science under research project TIN2007-60769.

generated by linear grammars will be denoted by LIN. We will say that a grammar G = (N,Σ,P,S) is *even linear* if every production in P is in the form A→uBv or A→w with A,B ∈ N, u,v,w ∈ Σ* and |u|=|v|. The class of languages generated by even linear grammars will be denoted by ELIN. A well known result from formal language theory is the inclusions REG ⊂ ELIN ⊂ LIN.

A finite automaton (FA) is defined by the tuple A=(Q,Σ,δ,I,F), where Q is a finite set of states, Σ is an input alphabet, I ⊆ Q is a set of initial states, F ⊆ Q is a set of final states and δ:Q×(Σ∪{λ})→P(Q) is a transition function where P(Q) denotes the power set of Q, that is the set of all possible subsets of Q. The automaton accepts an input string if there exist a sequence of transitions, according to δ, such that it begins in the initial state and, after analyzing the string, it ends in a final state. The language accepted by a finite automaton A is defined as the set of strings that it accepts and it is denoted by L(A). A particular case of finite automaton is the deterministic one where the transition function is defined as δ:Q×Σ→Q and the set I is composed by an unique state. Given any finite automaton A=(Q,Σ,δ,I,F), we will define the reverse automaton of A, and we will denote it by A^r, as the tuple $A^r=(Q,\Sigma,\delta^r,F,I)$ where $\delta^r(q,a)=\{p\in Q : q \in \delta(p,a)\}$.

A homomorphism *h* is defined as a mapping *h*: Σ→Γ* where Σ and Γ are alphabets. We can extend the definition of homomorphism over strings as *h*(λ)=λ and *h*(ax) = *h*(a) *h*(x) with a ∈Σ and *x* ∈ Σ*. Finally, the homomorphism over a language L ⊆ Σ* is defined as *h*(L) = { *h*(*x*) : *x* ∈ L }.

2. Regular reversible languages

Reversible languages were proposed by D. Angluin in [3]. There, she proposed an efficient method to identify these languages from samples of them. In addition, she studied different characterizations and relations between the *k*-reversible language classes. Here we will introduce some concepts and definitions proposed in her work.

We will say that a finite automaton A is *zero-reversible* if A and A^r are deterministic. Given a finite automaton A=(Q,Σ,δ,I,F) and a state q ∈ Q, we will say that the string *x* is a *k-follower* of q if and only if |*x*|=*k* and δ(q,*x*)≠∅. We will say that a finite automaton A=(Q,Σ,δ,I,F) is *deterministic with lookahead k* if and only if for any pair of distinct states q and p, if q,p ∈ I or q,p ∈ δ(s,a) then there is no string that is a k-follower of both q and p. We will say that a finite automaton A is *k-reversible* if and only if A is deterministic and A^r is deterministic with lookahead *k*. We will say that a language is *k*-reversible if there exist a minimum DFA with respect to the number of states which is *k*-reversible. The class of *k*-reversible languages will be denoted by *k*-REV. The following inclusion holds *k*-REV ⊂ (*k*+1)-REV. Finally the class of reversible languages, REV, will denote the class of languages that are *k*-reversible for any $k \geq 0$.

3. Watson-Crick finite automata

Given an alphabet Σ = {a_1, …, a_n}, we will use the symmetric (and injective) relation of complementarity ρ ⊆ Σ×Σ. For any string $x \in \Sigma^*$, we will denote by ρ(*x*) the string obtained by substituting the symbol *a* in *x* by the symbol *b* such that (*a,b*) ∈ ρ (remember that ρ is injective) with ρ(λ)=λ.

Given an alphabet Σ, a *sticker* over Σ will be the pair (*x,y*) such that $x = x_1 v x_2$, $y = y_1 w y_2$ with *x,y* ∈ Σ* and ρ(v)=w. The sticker (*x,y*) will be denoted by $\begin{pmatrix} x \\ y \end{pmatrix}$. A sticker $\begin{pmatrix} x \\ y \end{pmatrix}$ will be a complete and complementary *molecule* if |*x*|=|*y*| and ρ(*x*)=*y*. A complementary and complete molecule $\begin{pmatrix} x \\ y \end{pmatrix}$ will be denoted as $\begin{bmatrix} x \\ y \end{bmatrix}$. Obviously, any sticker $\begin{pmatrix} x \\ y \end{pmatrix}$ or molecule $\begin{bmatrix} x \\ y \end{bmatrix}$ can be represented by $x\#y^r$ where # ∉ Σ. Here, we will use $x\#y^r$ instead of *x*#*y* due to the grammar construction that we will propose in the following. Furthermore, inspired by DNA structure $x\#y^r$ represents the upper and lower nucleotide strings within the same direction 3'–5' (or 5'-3').

Formally, an *arbitrary* WK finite automaton is defined by the tuple M=(V,ρ,Q,s_0,F,δ), where Q and V are disjoint alphabets (states and symbols), ρ is a symmetric (and injective) relation of complementarity between symbols of V, s_0 is the initial state, F ⊆ Q is a set of final states and $\delta:Q\times\begin{pmatrix} V^* \\ V^* \end{pmatrix} \rightarrow P(Q)$.

The language of complete and complementary molecules accepted by M will be denoted by the set $L_m(M)$, while the upper strand language accepted by M will be denoted by $L_u(M)$ and defined as the set of strings *x* such that M enters into a final state after analyzing the molecule $\begin{bmatrix} x \\ y \end{bmatrix}$.

4. A Representation Theorem

Now, given any WKFA M, we will introduce a

representation theorem for the languages $L_m(M)$ and $L_u(M)$. First, remember that any double string $\begin{pmatrix} x \\ y \end{pmatrix}$ can be represented by the string $x\#y^r$. Then, the following result holds

Theorem 1. (*Sempere*,[7]) Let $M=(V,\rho,Q,s_0,F,\delta)$ be an arbitrary WK finite automaton. Then there exists a linear language L_1 and an even linear language L_2 such that $L_m(M)=L_1 \cap L_2$.

The construction for L_1 and L_2 proposed in the theorem is defined as follows. First, the grammar $G_1=(N,V\cup\{\#\},P,s_0)$ where $N = Q$, s_0 is the axiom of the grammar and P is defined as

1. If $q \in F$ then $q \rightarrow \# \in P$.
2. If $p \in \delta(q,\begin{pmatrix} x_1 \\ x_2 \end{pmatrix})$ then $q \rightarrow x_1\, p\, x_2^r \in P$.

The language L_2 is defined by the grammar $G_2=(\{S\}, V\cup\{\#\}, P, S)$ where P is defined as follows

1. $S \rightarrow \# \in P$.
2. For every pair of symbols $a,b \in V$, such that $(a,b) \in \rho$, $S \rightarrow a\, S\, b \in P$.

It can be easily proved that $L(G_2) = \{x_1\#x_2^r \in V^*\#V^* : |x_1|=|x_2|$ and $\rho(x_1)=x_2\}$. That is, L_2 can be established as the set of complete and complementary molecules $\begin{bmatrix} x \\ y \end{bmatrix}$ in the form $x\#y^r$.

From L_1 and L_2 it is clear that $L_1 \cap L_2$ is the set of complete and complementary molecules accepted by M in the form $x\#y^r$.

In order to characterize the upper strand language we will provide the following result

Corollary 1. (*Sempere*,[7]) Let $M=(V,\rho,Q,s_0,F,\delta)$ be an arbitrary WK finite automaton. Then $L_u(M)$ can be expressed as $g(h^{-1}(L_1 \cap L_2) \cap R)$ with L_1 being a linear language, L_2 an even linear language, R a regular language and *g* and *h* homomorphisms.

III. REGULAR REVERSIBILITY IN WATSON-CRICK FINITE AUTOMATA

In this section, we will introduce regular reversibility in the upper or lower strand, and in the double strand of the WKFA model. Given that the languages accepted by arbitrary WKFA can be represented by linear and even linear languages, we will introduce two reductions from these language classes to the class REG.

The first transformation, the so called σ operator, was first introduced in [8] and it was applied for the definition of local testable even linear languages in [9]. It is defined inductively as follows: $\sigma:\Sigma^* \rightarrow (\Sigma\times\Sigma)^*$ $(\Sigma \cup \{\lambda\})$ with

1. $\sigma(\lambda)=\lambda$.
2. $(\forall\, a \in \Sigma)\ \sigma(a) = a$.
3. $(\forall\, a,b \in \Sigma)\ (\forall\, x \in \Sigma^*)\ \sigma(axb) = [ab]\sigma(x)$.

The operation σ is applied over languages as $\sigma(L) = \{\sigma(x) : x \in L\}$. The inverse transformation σ^{-1} can be easily deduced from σ. It has been proved in [8] that for every even linear language L, $\sigma(L)$ is regular.

The second transformation is a grammatical construction that transforms every linear grammar into an even linear one. It is defined as follows.

Let $G_1=(N,\Sigma,P,S)$ be a linear grammar. Then $G_2=(N,\Sigma\cup\{@\},P',S)$ is an even linear grammar where the productions of P' are defined as follows

1. If $A \rightarrow w \in P$ then $A \rightarrow w \in P'$.
2. If $A \rightarrow uBv \in P$ with $|u|=|v|$, then $A \rightarrow uBv \in P'$.
3. If $A \rightarrow uBv \in P$ with $|u| < |v|$, then $A \rightarrow u\, @^{|v|-|u|} Bv \in P'$.
4. If $A \rightarrow uBv \in P$ with $|u| > |v|$, then $A \rightarrow u\, Bv\, @^{|u|-|v|} \in P'$.

The last grammar is an even linear one and it can be easily proved that $g(L(G_2)) = L(G_1)$ where *g* is a homomorphism such that $g(@)=\lambda$ and $g(a)=a$ for every $a \in \Sigma$.

1. Regular reversibility in the double strand

We will take the representation proposed in theorem 1. So, any molecule $\begin{bmatrix} x \\ y \end{bmatrix}$ can be represented by $x\#y^r$. Let us take G_1 as the linear grammar proposed in the theorem and let us take G_2 as the transformed even linear grammar corresponding to G_1. Obviously, for any string $x\#y^r$ of $L(G_1)$ we obtain a string u#v in $L(G_2)$ such that $g(u)\#g(v)=x\#y^r$, where *g* is the homomorphism defined before.

Now, we can work with G_2 and we apply the transformation σ over $L(G_2)$. Observe that $\sigma(L(G_2))$ is regular.

Example 1. Let $M = (V,\rho,Q,s_0,F,\delta)$ be the WKFA defined by the following transitions

$$\delta(q_0,\begin{pmatrix} a \\ \lambda \end{pmatrix}) = \{q_a\} \qquad \delta(q_a,\begin{pmatrix} a \\ \lambda \end{pmatrix}) = \{q_a\}$$

$\delta(q_a, \begin{pmatrix} b \\ a \end{pmatrix}) = \{q_b\}$ $\delta(q_b, \begin{pmatrix} b \\ a \end{pmatrix}) = \{q_b\}$

$\delta(q_b, \begin{pmatrix} c \\ b \end{pmatrix}) = \{q_c\}$ $\delta(q_c, \begin{pmatrix} c \\ b \end{pmatrix}) = \{q_c\}$

$\delta(q_c, \begin{pmatrix} \lambda \\ c \end{pmatrix}) = \{q_f\}$ $\delta(q_f, \begin{pmatrix} \lambda \\ c \end{pmatrix}) = \{q_f\}$

Let us take q_f as the final state, q_0 as the initial state and the complementarity relation $\rho = \{(a,a), (b,b),(c,c)\}$. Then, every complete and complementary molecule accepted by M takes the form $\begin{bmatrix} a^n b^n c^n \\ a^n b^n c^n \end{bmatrix}$ with $n \geq 1$.

Now, the representation linear grammar G_M, according to M is defined by the following productions (take q_0 as the axiom)

$q_0 \rightarrow a\ q_a$ $q_a \rightarrow a\ q_a \mid b\ q_b\ a$

$q_b \rightarrow b\ q_b\ a \mid c\ q_c\ b$ $q_c \rightarrow c\ q_c\ b \mid q_f\ c$

$q_f \rightarrow q_f\ c \mid \#$

The corresponding even linear grammar is the following

$q_0 \rightarrow a\ q_a\ @$ $q_a \rightarrow a\ q_a\ @ \mid b\ q_b\ a$

$q_b \rightarrow b\ q_b\ a \mid c\ q_c\ b$ $q_c \rightarrow c\ q_c\ b \mid @\ q_f\ c$

$q_f \rightarrow @\ q_f\ c \mid \#$

Finally, we can provide the following right linear grammar to obtain the transformation σ over the last grammar

$q_0 \rightarrow [a@]\ q_a$ $q_a \rightarrow [a@]\ q_a \mid [ba]\ q_b$

$q_b \rightarrow [ba]\ q_b \mid [cb]\ q_c$ $q_c \rightarrow [cb]\ q_c \mid [@c]\ q_f$

$q_f \rightarrow [@c]\ q_f \mid \#$

Observe that the last grammar generates the language defined as

$L = \{[a@]^n\ [ba]^m\ [cb]^p\ [@c]^q\ \# : n,m,p,q \geq 1\}$.

Then, if we take the morphism g with $g(@) = \lambda$ and $g(d) = d$ for every $d \in \{ a,b,c,\# \}$ we can obtain $g(\sigma^{-1}(L)) = \{ a^n b^m c^p \# c^q b^p a^m : n,m,p,q \geq 1 \}$ which, together with the complementary relation ρ, corresponds to the language accepted by M.

So, the definition of regular reversibility will be applied over the regular language obtained by the result $\sigma(L(G_M))$ for any WKFA M. Observe that every transformed language in *k*-REV has a corresponding regular reversible language defined by the transitions of the WKFA.

2. Regular reversibility in the upper (lower) strand

Now, we will deal only with the upper (lower) strand. Observe that, the definition of the WKFA transitions can be transformed into FA transitions by taking the upper or lower strand (i.e., the transition $p \in \delta(q, \begin{pmatrix} x \\ y \end{pmatrix})$ implies that $p_u \in \delta_u(q,x)$ and $p_l \in \delta_l(q,y)$). So, for every WKFA we can obtain two different finite automata which control the transitions in the upper and lower strands. Here, we will work with *simple* WKFA [1]. We will say that a WKFA is *simple* if for every transition $\delta(q, \begin{pmatrix} x \\ y \end{pmatrix})$, $x=\lambda$ or $y=\lambda$. It has been proved that simple WKFA are normal forms for arbitrary WKFA. That is, for every arbitrary WKFA there exists an equivalent simple WKFA. Furthermore, we can work with the so called 1-*limited* WKFA which are simple WKFA where every transition is performed by analyzing only one symbol every time. Now, we will obtain finite automata from arbitrary 1-*limited* WKFA through the following construction. Let $M=(V,\rho,Q,s,F,\delta)$ be an arbitrary 1-*limited* WKFA. Then, we can define the finite automaton $A_u = (Q, V, \delta_u, s, F)$, where δ_u is defined as follows

1. $p \in \delta_u(q,a)$ if and only if $p \in \delta(q, \begin{pmatrix} a \\ \lambda \end{pmatrix})$.

2. $p \in \delta_u(q,\lambda)$ if and only if $p \in \delta(q, \begin{pmatrix} \lambda \\ a \end{pmatrix})$.

We can define the finite automaton $A_l = (Q, V, \delta_l, s, F)$ where δ_l is defined as follows

1. $p \in \delta_l(q,a)$ if and only if $p \in \delta(q, \begin{pmatrix} \lambda \\ a \end{pmatrix})$.

2. $p \in \delta_l(q,\lambda)$ if and only if $p \in \delta(q, \begin{pmatrix} a \\ \lambda \end{pmatrix})$.

Example 2. Let us take the WKFA of example 1. Then A_u is defined by the following transitions

$\delta_u(q_0,a) = \{ q_a \}$ $\delta_u(q_a,a) = \{ q_a \}$

$\delta_u(q_a,b) = \{ q_{bb} \}$ $\delta_u(q_{bb},\lambda) = \{q_b \}$

$\delta_u(q_b,b) = \{ q_{bbb} \}$ $\delta_u(q_{bbb},\lambda) = \{ q_b \}$

$\delta_u(q_b,c) = \{ q_{cc} \}$ $\delta_u(q_{cc},\lambda) = \{ q_c \}$

$\delta_u(q_c,c) = \{ q_{ccc} \}$ $\delta_u(q_{ccc},\lambda) = \{ q_c \}$

$\delta_u(q_c,\lambda) = \{q_f \}$

In the previous definitions, the states q_{bb}, q_{bbb}, q_{cc}

and q_{ccc} have been introduced in order to obtain an equivalent 1-*limited* WKFA from the one proposed initially. In this case $L(A_u) = a^+b^+c^+$. The same holds for $L(A_l)$.

Observe that, in both automata A_u and A_l, the empty transitions correspond to the case that the WKFA is working in the other strand, so the finite automaton ignores all the movements in that way.

Now, the definitions of regular reversibility come from a natural way of looking up to the FA A_u and A_l. We will say that a 1-*limited* WKFA is upper (lower) reversible if the language accepted by A_u (resp. A_l) is reversible. Observe that this definition implies the existence of different classes of languages accepted by WKFA which have regular reversibility. These classes are defined as follows

1. The class k-REV_u of languages accepted by 1-*limited* WKFA which have k-reversibility in the upper strand.

2. The class k-REV_l of languages accepted by 1-*limited* WKFA which have k-reversibility in the lower strand.

We can make a step further the definition of a new kind of regular reversibility in every strand by introducing a combination of classes considered up to now in an isolated way. Let us take the finite automata A_l and A_u proposed before. Observe that if $L(A_l)$ is in j-REV, then $L(A_l)$ belongs to k-REV for every $j \leq k$. The same holds for A_u. So, we can combine different language classes in the upper and the lower strand and they define new classes (k,j)-REV of languages accepted by 1-*limited* WKFA which have k-reversibility in the upper strand and j-reversibility in the lower strand.

V. CONCLUSION

In this work we have introduced regular reversibility in Watson-Crick finite automata. Due to the representation theorem that we have introduced in section II, and the reduction to regular languages, we can introduce different characteristic features in WKFA by translating them from regular languages. This allows the inference of some restricted models of WKFA in order to apply them to practical approaches. In addition, it defines new language classes accepted by WKFA which will be explored in the near future.

REFERENCES

[1] Freund R, Păun G, Rozenberg G, Salomaa A (1999) Watson-Crick finite automata. Proceedings of DNA Based Computers III DIMACS Workshop, 297-327. The American Mathematical Society.

[2] Czeizler E, Czeiler E (2006) A Short Survey on Watson-Crick Finite Automata. Bulletin of the EATCS 88: 104-119.

[3] Angluin D (1982) Inference of Reversible Languages. Journal of the Association for Computer Machinery Vol.29 No.3: 741-765

[4] Hopcroft J, Ullman J (1979) Introduction to Automata Theory, Languages and Computation. Addison Wesley Publishing Co.

[5] Rozenberg G, Salomaa A (eds.) (1997) Handbook of Formal Languages Vol. 1. Springer.

[6] Păun Gh, Rozenberg G, Salomaa A (1998) DNA computing. New computing paradigms. Springer

[7] Sempere JM (2004) A Representation Theorem for Languages accepted by Watson-Crick Finite Automata. Bulletin of the EATCS 83: 187-191.

[8] Sempere JM, García P (1994) A Characterization of Even Linear Languages and its Application to the Learning Problem. Proceedings of ICGI 1994, LNAI 862, 28-44. Springer-Verlag.

[9] Sempere JM, García P (2002) Learning Locally Testable Even Linear Languages from Positive Data. Proceedings of ICGI 2002, LNAI 2484, 225-236. Springer-Verlag.