Complexity analysis of sticker systems by means of comparison with multihead finite automata

Peter Weigel

Institut für Informatik, Fachbereich Mathematik/Informatik,
Martin-Luther-Universität Halle-Wittenberg, D-06120 Halle/Saale, Germany
email: mail@stickersysteme.de, url: http://www.stickersysteme.de

Februar 2005

Abstract

The complexity analysis of sticker systems and Chomsky grammars done in [KPG98], [FPR98], [PR98], [PRS98] and [WK05] are incomplete. Here we will extend these analysis by multihead finite automata and therewith close all open relations between sticker language families and Chomsky language families.

Key words: dna-computing, sticker system, Chomsky grammar, multihead finite automaton, complexity analysis

1 Introduction

In [Adl94] L. M. Adleman gives a procedure for solving the *Hamiltonian Path Problem (HPP)* based on DNA strands. This procedure, known as *Adleman's Experiment*, can be considered as the basis of the concept of *sticker systems*, which was introduced in [KPG98] as *regular sticker systems*. Sticker systems with capabilities of synchronizing the extension on the left and right were mentioned first in [FPR98] as *bidirectional sticker systems*. In [PR98] both concepts were merged to a new concept called *sticker systems*. The definition of sticker systems from [PR98] and a lot of results and proofs from [KPG98], [FPR98] and [PR98] were thereafter summarized and supplemented in [PRS98].

Sticker systems are one of many ways for theoretical analysis of properties and capabilities of DNA strands, which are interesting for language, computation or complexity theories. Because the constructs of sticker systems, which can be considered as grammars, are completely different from Chomsky grammars, an explicit analysis is needed. In [KPG98], [FPR98], [PR98], [PRS98] and [WK05] there are a lot of proofs for relations between sticker language families among themselves and Chomsky language families.

Here we will extend the concept of sticker systems presented in [PRS98] a little bit and transfer some results from [PRS98] about sticker systems to the new extended concept. Additionally, we will give evidence that sticker systems can be simulated by 4-headed multihead finite automata and we will investigate how one can reduce the count of heads by restricting sticker systems. In this framework, we will prove the relations $LIN \not\subseteq OSL(n)$, $CF \not\subseteq ASL(n)$ and $ASL(n) \subset CS$ and therewith close all open questions concerning the relationship between sticker language families and Chomsky language families.

Beside the concept of *sticker systems*, in [PRS98] there were also presented *Watson-Crick finite automata*, *Insertion-Deletion systems* and different kinds of *splicing systems* (see [PRS98, Chapter 4, 5, 6, 7-11]). The roots of all these concepts can be found in the field of *DNA-Computing*. But for all these concepts there exists equivalent or similar concepts, which are already known 'classic' concepts. Fore example, Insertion-Deletion systems can be considered as Chomsky grammars $G = (T, N, S, P)$ with productions (S, w), (uv, uwv) and (uwv, uv) with $u, v, w \in (T \cup (N \setminus \{S\}))^*$. According to [PRS98, Lemma 5.8] Watson-Crick finite automata are equivalent to (0.2)-headed simple oneway multihead finite automata. Inevitably, we have to ask for the existence of a 'classic' concept, which is equivalent or rather similar to sticker systems. We will answer this question in Remark 5.2.

This work is based on the diploma thesis [Wei04] and only an abridged version of the analysis given there. Some parts of the diploma thesis are already included in the publications [KW04] and [WK05].

It must be mentioned, that we did two mistakes in [KW04]. At the end of the proof of [KW04, Theorem 6] it must $C^k(S') \cap \rho^* = \overline{C^k(S)} \cap \Delta_V^*$ be replaced by $C^k(S') \cap \Delta_V^* = \overline{C^k(S) \cap \rho^*}$. The second mistake was, that we omitted to mention, that [KW04, Theorem 6] is a transcription and generalization of [PRS98, Lemma 5.8]. The proof, that the complementarity $\rho = \{(x, x) : x \in V\}$ suffices for Watson-Crick finite automata as well, see [KW04, Theorem 4], is a trivial generalization, because we can get this result by connecting the constructions from [PRS98, Lemma 5.7], [PRS98, Lemma 5.8] and their reversions. Additionally, it must be mentioned, that this omission was also done in [Sem04], because [Sem04, Theorem 2.1] is a trivial generalization of [PRS98, Lemma 5.12].

2 Basic definitions

By $\mathbb{N}$ we denote the set of non-negative integers. The set of all subsets of a set A is denoted by $P(A)$. The *empty set* is denoted by $\emptyset$.

An *alphabet* is a nonempty, finite set of abstract symbols. The elements of an alphabet are called *letters*. Let Σ be an alphabet. A *word* over Σ is a finite sequence of letters of Σ. Σ^* denotes the set of all words over the alphabet Σ including the *empty word* ε. We define $\Sigma^+ := \Sigma^* \setminus \{\varepsilon\}$. A *language* L over the alphabet Σ is a subset of Σ^*. The *complement* of a language L is denoted by L^{co} and defined by $L^{co} := \{w \in \Sigma^* : w \notin L\}$. Let $k, i \in \mathbb{N}$ and $w = a_1 a_2 \dots a_k \in \Sigma^*$ be a word. We call $w^R := a_k \dots a_2 a_1$ the *reversion*, $|w| := k$ the *length* and $w[i] := a_i$ the i-th *letter* of w if $1 \le i \le k$. Additionally, we define $w[i] := \varepsilon$ for $i < 1$ or $i > k$. The *concatenation* of two words u and v with $u = a_1 \dots a_k$ and $v = b_1 \dots b_m$ is defined by $u \cdot v := a_1 \dots a_k b_1 \dots b_m$.

Let $k \in \mathbb{N}$. Σ^k denotes the k-fold cartesian product of a nonempty set Σ. The elements of Σ^k are called *vectors*. Let $\Sigma_1, \Sigma_2, \dots, \Sigma_k$ be a finite sequence of nonempty sets and $x = (x_1, x_2, \dots, x_k) \in \Sigma_1 \times \Sigma_2 \times \dots \times \Sigma_k$. We call $x[i] := x_i$ the i-th *component* of x if $1 \le i \le k$.

Let $k \in \mathbb{N}$ and $\lambda : A^k \to B$ be a partial mapping from A^k into B. In general, we define the extension $\lambda : P(A)^k \to P(B)$ by

$$\lambda(X_1, X_2, \dots, X_k) := \left\{ \lambda(x_1, x_2, \dots, x_k) : \begin{array}{c} x_i \in X_i \text{ for } 1 \le i \le k, \\ \lambda(x_1, x_2, \dots, x_k) \text{ is defined} \end{array} \right\}.$$

The *reversion* A^R and the *concatenation* $A \cdot B$ of two languages A and B are therewith defined.

2.1 Sticker systems

We now want to define the concept of sticker systems analogously to [WK05]. Be aware of the differences in the definition of the dominoes.

Let V be an alphabet and $\rho \subseteq V \times V$ be a symmetrical binary relation. There are $\binom{V^*}{V^*} := \left\{ \binom{u}{v} : u, v \in V^* \right\}$ the set of all *pairs of words* of V^* and $\begin{bmatrix} V^* \\ V^* \end{bmatrix}_\rho := \left\{ \binom{u}{v} \in \binom{V^*}{V^*} : |u| = |v|, (u[i], v[i]) \in \rho \text{ for } 1 \le i \le |u| \right\}$ the set of all *pairs of complementary words* of V^*. For $\binom{u}{v} \in \begin{bmatrix} V^* \\ V^* \end{bmatrix}_\rho$ we write $\begin{bmatrix} u \\ v \end{bmatrix}_\rho$. The concatenation $\binom{x_1}{x_2} \cdot \binom{y_1}{y_2}$ is $\binom{x_1 \cdot y_1}{x_2 \cdot y_2}$. Analogously, we write $\begin{bmatrix} x_1 \cdot y_1 \\ x_2 \cdot y_2 \end{bmatrix}_\rho$ for $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_\rho \cdot \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}_\rho$.

The set of all *dominoes* $W_\rho(V)$ is defined by $W_\rho(V) := S_\rho(V) \cup LR_\rho(V)$ where $S_\rho(V) := \binom{V^*}{V^*}$ is called set of *simple dominoes*, $O_\rho(V) := \left\{ \binom{u}{v} \in S_\rho(V) : u = \varepsilon \text{ or } v = \varepsilon \right\}$ is called set of *one-stranded dominoes*, $E_\rho(V) := S_\rho(V) \setminus O_\rho(V)$ is called set of *extended dominoes*, $LR_\rho(V) := O_\rho(V) \times \left(\begin{bmatrix} V^* \\ V^* \end{bmatrix}_\rho \setminus \left\{ \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}_\rho \right\} \right) \times O_\rho(V)$ is called set of *non-simple dominoes* and $WK_\rho(V) := \left\{ \binom{\varepsilon}{\varepsilon} \right\} \times \left(\begin{bmatrix} V^* \\ V^* \end{bmatrix}_\rho \setminus \left\{ \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}_\rho \right\} \right) \times \left\{ \binom{\varepsilon}{\varepsilon} \right\}$ is called set of *complete dominoes*. The domino $\binom{\varepsilon}{\varepsilon}$ is also identified by ε and therewith we can write $\begin{bmatrix} x \\ y \end{bmatrix}_\rho$ instead of $\binom{\varepsilon}{\varepsilon} \begin{bmatrix} x \\ y \end{bmatrix}_\rho \binom{\varepsilon}{\varepsilon}$.

Let $x \in LR_\rho(V)$ and $y \in S_\rho(V)$ be two dominoes. x_1^t, x_1^b, x_2^t, x_2^b, x_3^t, x_3^b, y^t and y^b denote the single components of x and y with $x = \left(\binom{x_1^t}{x_1^b}, \begin{bmatrix} x_2^t \\ x_2^b \end{bmatrix}_\rho, \binom{x_3^t}{x_3^b} \right)$ and $y = \binom{y^t}{y^b}$. Additionally, we call $x_2 := \begin{bmatrix} x_2^t \\ x_2^b \end{bmatrix}_\rho$ the *centerpiece*, $x_1 := \binom{x_1^t}{x_1^b}$ the *left* and $x_3 := \binom{x_3^t}{x_3^b}$ the *right delay* of x. We write $x = x_1 x_2 x_3$ instead of $x = (x_1, x_2, x_3)$. The words y^t and $x^t := x_1^t \cdot x_2^t \cdot x_3^t$ are called *upper strand*. Analogously, y^b and $x^b := x_1^b \cdot x_2^b \cdot x_3^b$ are called *lower strand*. The letters of the single components are called *bases*.

The *structure* of a domino x is defined by the mapping *struct* : $W_\rho(V) \to W_{\{(\sharp,\sharp)\}}(\{\sharp\})$, whereby $struct(x)$ arises from x by substituting all bases contained therein by $\sharp$.

The *delay* of a domino is defined by the mapping $d : W_\rho(V) \to \mathbb{N}$ with

$$d(x) := \begin{cases} max \left\{ |x_1^t|, |x_1^b|, |x_3^t|, |x_3^b| \right\} & \text{if } x \in LR_\rho(V), \\ max \left\{ |x^t|, |x^b| \right\} & \text{if } x \in S_\rho(V). \end{cases}$$

Let $x, y \in W_\rho(V)$ be two dominoes. The *sticking* of x and y is defined by the mapping $\mu_\rho : W_\rho(V) \times W_\rho(V) \to W_\rho(V)$ with

$$\mu_\rho(x,y) := \begin{cases} x_1 \left(x_2 \cdot \begin{bmatrix} u \\ v \end{bmatrix}_\rho \cdot y_2 \right) y_3 & \text{if } x \in LR_\rho(V), y \in LR_\rho(V), \\ & x_3 \cdot y_1 = \begin{bmatrix} u \\ v \end{bmatrix}_\rho, \\ x_1 \left(x_2 \cdot \begin{bmatrix} u \\ v \end{bmatrix}_\rho \right) w & \text{if } x \in LR_\rho(V), y \in S_\rho(V), \\ & x_3 \cdot y = \begin{bmatrix} u \\ v \end{bmatrix}_\rho \cdot w, w \in O_\rho(V), \\ w \left(\begin{bmatrix} u \\ v \end{bmatrix}_\rho \cdot x_2 \right) x_3 & \text{if } x \in S_\rho(V), y \in LR_\rho(V), \\ & x \cdot y_1 = w \cdot \begin{bmatrix} u \\ v \end{bmatrix}_\rho, w \in O_\rho(V), \\ x \cdot y & \text{if } x \in S_\rho(V), y \in S_\rho(V), \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Because μ_ρ is associative, we write $x \cdot_\rho y$ instead of $\mu_\rho(x,y)$.

A *sticker system* is a construct

$$\gamma = (V, \rho, A, D)$$

with an alphabet V, a symmetrical binary relation $\rho \subseteq V \times V$, a finite set $A \subseteq LR_\rho(V)$ and a finite set $D \subseteq W_\rho(V) \times W_\rho(V)$. The relation ρ is called *complementarity* of V. The elements of A are called *axioms* and the elements of D are called *rules*.

Let $x, y \in W_\rho(V)$ be two dominoes. We write $x \to_\gamma y$ iff there exists a rule $(u, v) \in D$ with $y = u \cdot_\rho x \cdot_\rho v$. We write $x \to_\gamma^k y$ for $x = x_0 \to_\gamma x_1 \to_\gamma x_2 \to_\gamma \cdots \to_\gamma x_k = y$ with $k \in \mathbb{N}$ and $x_i \in W_\rho(V)$ for $0 \leq i \leq k$ or rather $x \to_\gamma^* y$ iff there exists such a k and call this a *derivation* iff $x \in A$ and a *complete derivation* iff it is $y \in WK_\rho(V)$, additionally.

Let $C^0(\gamma) := A$ and $C^k(\gamma) := \left\{ y \in W_\rho(V) : \exists x \in C^{k-1}(\gamma) : x \to_\gamma y \right\}$ with $k \in \mathbb{N}$ and $k \geq 1$. $C^*(\gamma) := \bigcup_{k \in \mathbb{N}} C^k(\gamma)$ denotes the set of *dominoes generated by* γ, $LM(\gamma) := C^*(\gamma) \cap WK_\rho(V)$ the *language of molecules generated by* γ and $L(\gamma) := \{x^t : x \in LM(\gamma)\}$ the *language generated by* γ.

It is $\varepsilon \notin L(\gamma)$ for every sticker system $\gamma = (V, \rho, A, D)$. Now we extend the definition of sticker systems and allow $\binom{\varepsilon}{\varepsilon} \in A$. Thereby we have to ensure, that this special axiom will never be used for derivations. It is $\varepsilon \in L(\gamma)$ iff $\binom{\varepsilon}{\varepsilon} \in A$.

A rule $(u, v) \in D$ is called *extended* iff at least one of the both components is an extended domino, *simple* iff both dominoes are simple, *one-stranded* iff both dominoes are one-stranded, *left-sided* iff $v = \varepsilon$, *right-sided* iff $u = \varepsilon$ and *one-sided* if it is left-sided or right-sided. A derivation $x_0 \to_\gamma^* x_k$ is called *delay bounded by the bound* $d \in \mathbb{N}$ iff $d(x_i) \leq d$ for $0 \leq i \leq k$. A sticker system $\gamma = (V, \rho, A, D)$ is called *extended* iff there exist at least one extended rule in D, *simple* iff all rules of D are simple, *one-sided* iff all rules in D are one-sided, *regular* iff all rules in D are right-sided and *with bounded delay* iff there exists a constant $d \in \mathbb{N}$, such that for every domino $x \in LM(\gamma)$ there exists at least one delay bounded derivation with the delay bound d.

$ASL(n)$ denotes the family of languages generated by sticker systems. Restriction to sticker systems with bounded delay is denoted by substituting n by b. The prohibition of extended dominoes is denoted by prefixing the symbol c, which stands for *classic*. Restrictions to simple, one-sided, regular, simple and one-sided or simple and regular sticker systems are denoted by substituting A by S, O, R, SO or SR.

2.2 Chomsky grammars

By CS, CF, LIN and REG we denote the families of languages, which are generated by *context-sensitive*, *context-free*, *linear* and *regular* Chomsky grammars e.g. defined in [WW86, Section 4.1.1].

Lemma 2.1 ([WW86, Theorem 4.6]) $REG \subset LIN \subset CF \subset CS$.

2.3 Multihead finite automata

Let $k \in \mathbb{N}$, $K := \{1, \ldots, k\}$ and $x, y \in \mathbb{N}^k$. The tuple y is called *compression* of x, we write $y = comp(x)$, iff the following two conditions are sufficed:

(1) $\forall i, j \in K : x[i] \leq x[j] \iff y[i] \leq y[j]$,
(2) $\forall i \in K, y[i] \neq 0 : \exists j \in K : y[j] = y[i] - 1$.

$COMP_k := \left\{ comp(x) : x \in \mathbb{N}^k \right\}$ with $k \in \mathbb{N}$ is called set of k-compressions.

A *multihead finite automaton* is a construct

$$\mathcal{A} = (X, Z, s, F, T)$$

with an alphabet X, a finite set Z, a symbol $s \in Z$, a finite set $F \subseteq Z$ and a finite set $T \subseteq Z \times V^k \times COMP_k \times Z \times M^k$ with $V := X \cup \{\varepsilon\}$, $M := \{-1, 0, +1\}$ and $k \in \mathbb{N}$. X is called *input alphabet*, V *work alphabet*, Z set of *states*, s *initial state*, F set of *final states*, T set of *transitions* and k count of *heads*. A multihead finite automaton with k heads is called *k-headed multihead finite automaton*.

$C(\mathcal{A}) := X^* \times Z \times \mathbb{N}^k$ is called set of *configurations*[1] of $\mathcal{A}$. A configuration $x \in C(\mathcal{A})$ is called *initial* iff $x \in C_i(\mathcal{A}) := X^* \times \{s\} \times \{(1, \ldots, 1)\}$ or *final* iff $x \in C_f(\mathcal{A}) := X^* \times F \times \mathbb{N}^k$. Let $x, y \in C(\mathcal{A})$ be two configurations with $x = (w, z_x, \vec{h_x})$ and $y = (w, z_y, \vec{h_y})$. We write $x \vdash_{\mathcal{A}} y$ and call y *next configuration* of x iff there exists a transition[2] $t = (z, \vec{v}, \vec{c}, q, \vec{m}) \in T$ with $z_x = z$, $\forall i \in K : w[\vec{h_x}[i]] = \vec{v}[i]$, $comp(\vec{h_x}) = \vec{c}$, $z_y = q$, $\forall i \in K : \vec{h_y}[i] = min\{u + 1, max\{0, \vec{h_x}[i] + \vec{m}[i]\}\}$. We write $x \vdash^u_{\mathcal{A}} y$ for $x = x_0 \vdash_{\mathcal{A}} x_1 \vdash_{\mathcal{A}} x_2 \vdash_{\mathcal{A}} \cdots \vdash_{\mathcal{A}} x_u = y$ with $u \in \mathbb{N}$ and $x_i \in C(\mathcal{A})$ for $0 \leq i \leq k$ or rather $x \vdash^*_{\mathcal{A}} y$ iff there exists such a u and call this a *run* iff x is initial or a *successful run* iff y is final, additionally.

Let $C^0(\mathcal{A}) := C_i(\mathcal{A})$ and $C^k(\mathcal{A}) := \left\{y \in C(\mathcal{A}) : \exists x \in C^{k-1}(\mathcal{A}) : x \vdash_{\mathcal{A}} y\right\}$ with $k \in \mathbb{N}$ and $k \geq 1$. $C^*(\mathcal{A}) := \bigcup_{k \in \mathbb{N}} C^k(\mathcal{A})$ denotes the set of *reachable configurations*, $C^*_f(\mathcal{A}) := C^*(\mathcal{A}) \cap C_f(\mathcal{A})$ the set of *reachable final configurations* and $L(\mathcal{A}) := \left\{c[1] : c \in C^*_f(\mathcal{A})\right\}$ the *language* accepted by $\mathcal{A}$.

A k-headed multihead finite automaton $\mathcal{A} = (X, Z, s, F, T)$ is called *simple* iff for all transitions $(z, \vec{v}, \vec{c}, q, \vec{m}) \in T$ and for all $\vec{r} \in COMP_k$ it is $(z, \vec{v}, \vec{r}, q, \vec{m}) \in T$, that means the relative head position detection isn't needed. For simplification, we forget the component $\vec{c}$ or rather $\vec{r}$ and write $(z, \vec{v}, q, \vec{m}) \in T$.

A *oneway multihead finite automaton* is a multihead finite automaton with $s + t$ heads ($s, t \in \mathbb{N}$) positioning these heads together at a position p and thereafter moving head 1 to s to the left ($\vec{m}[i] \in \{-1, 0\}$) and head $s + 1$ to $s + t$ to the right ($\vec{m}[i] \in \{0, +1\}$). Let w be the input word, then we enforce $p = 1$ for $w = \varepsilon$, $p = 1$ for $s = 0$ and $p = |w|$ for $t = 0$, otherwise p is a random integer from the interval $[1, |w|]$. We call such an automaton with s left-running and t right-running heads a *$(s.t)$-headed oneway multihead finite automaton.*

AHL_k denotes the family of languages accepted by k-headed multihead finite automata. Restriction to simple multihead finite automata is denoted by substituting A by S. For $s, t \in \mathbb{N}$ we denote the family of languages accepted by $(s.t)$-headed oneway multihead finite automata by $OHL_{s.t}$. Restriction to simple oneway multihead finite automata is denoted by prefixing the symbol S. By unions over k, s or t we get the multihead languages AHL_*, SHL_*, $OHL_{s.*}$, $SOHL_{s.*}$, $OHL_{*.t}$, $SOHL_{*.t}$, $OHL_{*.*}$ and $SOHL_{*.*}$.

[1] A configuration consists of the input word currently worked with, the current state and the current positions of the k heads.

[2] A transition describes changes of states or head positions. A multihead finite automaton goes into a state q and moves its heads in conformity with $\vec{m}$, if it is currently situated in state z, $\vec{v}$ describes the letters currently read by heads 1 to k and $\vec{c}$ describes the current relative positions of the k heads to each other. The input word w cannot be manipulated and the heads cannot go beyond the end marker ε.

Theorem 2.2 *$YHL_{0.k} = YHL_{k.0}$. ($k \in \mathbb{N}$, $Y \in \{O, SO\}$)*

(Without a proof. This result can be assumed as already known. The proof uses the idea of backward simulation.)

At the basis of Theorem 2.2 we define $YHL_t := YHL_{0.t}$ for $Y \in \{O, SO\}$ and $t \in \mathbb{N} \cup \{*\}$. [3]

Remark 2.3 *The concept of multihead finite automata with capabilities of detecting relative head positions presented here is equivalent to the already known concept of multihead finite automata with capabilities of detecting head coincidences. (This capability is also called 'sensing'.).*

3 Transcription of results about 'classic' sticker systems to the extended concept defined in section 2.1

A lot of results about multihead finite automata without the capability of relative head position detection can be transcripted or generalized to multihead finite automata with the capability of relative head position detection. This also works for sticker systems: Nearly all results about sticker systems can be transcripted or rather generalized to the extended concept defined in section 2.1. The modifications of the proofs are marginal and mostly obvious. For illustration we will give some examples. This collection is not complete.

Lemma 3.1 ([WK05, Lemma 3.1]) *For every sticker system $\gamma = (V, \rho, A, D)$ there exists an effectively constructable sticker system $\gamma' = (V, \rho', A', D')$ with $L(\gamma) = L(\gamma')$ and $\rho' = \{(x, x) : x \in V\}$. Additionally, the transformation from γ to γ' holds for any property[4] of rules and derivations defined in this publication or in [PRS98].*

Proof. The proof of [WK05, Lemma 3.1] concerning $cASL(n)$ can be generalized to $ASL(n)$. Therefor one have to extend the mapping λ_ρ to $E_\rho(V)$ with $\lambda_\rho\left(\binom{u}{v}\right) := \lambda_\rho\left(\binom{u}{\varepsilon}\right) \cdot \lambda_\rho\left(\binom{\varepsilon}{v}\right)$.

Theorem 3.2 (comp. [PRS98, Theorem 4.2]) $SRSL(n) \not\subseteq CF$.

[3] In [WW86] the language family SHL_k is denoted by $2{:}k{-}NFA$ (or $2{-}NFA$ for $k = 1$). SHL_* is equivalent to $2{:}multi{-}NFA$. Analogously, one have to replace $SOHL_k$ by $1{:}k{-}NFA$ (or $1{-}NFA$ for $k = 1$) and $SOHL_*$ by $1{:}multi{-}NFA$. For [Mon80] we have to replace SHL_k by $NH(k)$ and for [YR78] $SOHL_k$ by R_k.

[4] The transformation of γ to γ' preserves the properties simple, one-sided, right-sided, with bounded delay, ...

Proof. Let $\gamma = (V, \rho, A, D)$ be the sticker system with $V = \{a, b, c, z\}$, $\rho = \{(x, x) : x \in V\}$, $A = \left\{\left[\begin{smallmatrix} z \\ z \end{smallmatrix}\right]\right\}$ and $D = \left\{\left(\binom{\varepsilon}{\varepsilon}, \binom{a}{\varepsilon}\right), \left(\binom{\varepsilon}{\varepsilon}, \binom{b}{a}\right), \left(\binom{\varepsilon}{\varepsilon}, \binom{c}{b}\right), \left(\binom{\varepsilon}{\varepsilon}, \binom{\varepsilon}{c}\right)\right\}$. It is $L(\gamma) \in SRSL(n) \setminus CF$ analogously to [PRS98, Theorem 4.2].

Theorem 3.3 ([PRS98, Theorem 4.1]) *$OSL(b) \subseteq REG$.*

Theorem 3.4 ([PRS98, Theorem 4.3]) *$ASL(b) \subseteq LIN$.*

Theorem 3.5 ([WK05, Theorem 5.2]) *$REG \not\subseteq SSL(n)$.*

4 Chomsky grammars and multihead finite automata

We will now recollect some results about Chomsky language families and multihead language families. Therewith we can get results about sticker language families and multihead language families from results about sticker language families and Chomsky language families (and conversely). For example we can get the relations $ASL(b) = SOHL_{1.1}$ and $OSL(b) = SOHL_1$ from $ASL(b) = LIN = SOHL_{1.1}$ and $OSL(b) = REG = SOHL_1$.

In this framework, we have to remind some relations between multihead finite automata among themselves.

Theorem 4.1 ([Mon80], [YR78]) *$XHL_k \subset XHL_{k+1} \subset XHL_*$. ($k \in \mathbb{N}$, $X \in \{A, S, O, SO\}$)*

Theorem 4.2 ([YR78, Introduction]) *$OHL_3 \not\subseteq SOHL_*$.*[5]

Theorem 4.3 *$AHL_k \subseteq SHL_{k+1}$. ($k \in \mathbb{N}$)*

(Without a proof. This result can be assumed as already known. However, a possible proof is given in [Wei04].)

Theorem 4.4

$$\begin{aligned} REG &= XHL_1,\ (X \in \{A, S, O, SO\}) \\ LIN &= YHL_{1.1},\ (Y \in \{O, SO\}) \\ CF &\not\supseteq SOHL_2, \\ CS &\supset AHL_*. \end{aligned}$$

Proof. Line 1 to 3 is already known.

[5] Concluding from this theorem, we get $SOHL_k \subset OHL_k$ for $k \geq 3$.

Line 4 concludes from $AHL_* \subseteq SHL_* = NL \subset NLINSPACE = CS$ according to Theorem 4.3, [WW86, Theorem 13.2(8,space)], [WW86, Section 22.3] [6] and [WW86, Theorem 12.15].

Theorem 4.5 ([WW86, Theorem 13.5(3)])

$$\begin{aligned} LIN &\not\subseteq OHL_*, \\ CF &\not\subseteq OHL_{*.*}, \\ SHL_2 &\not\subseteq OHL_{*.*}. \end{aligned}$$

Proof. Let $S_1 := \{w \in \{a,b\}^* : w = w^R\}$. It is $S_1 \in LIN$. According to [WW86, Theorem 13.5] it is $S_1 \notin SOHL_*$. We can get $S_1 \notin OHL_*$ analogously.

Let $S_2 := S_1 \cdot S_1 = \left\{v \cdot w \in \{a,b\}^* : v = v^R, w = w^R\right\}$. It is $S_2 \in CF$. Analogous to $S_1 \notin OHL_*$ it is $S_2 \notin OHL_{*.*}$. Thereby we mainly make use of the fact, that on every run on a word $v \cdot w$ one of the parts v and w is situated completely on the left or right side.

It is $S_2 \in SHL_2$. (Without a proof.)

5 Simulation of sticker systems by multihead finite automata

Theorem 5.1 $ASL(n) \subseteq OHL_{2.2}$.

Proof. In [PRS98, Chapter 5] there was presented the concept of Watson-Crick finite automata. By suitable extension of this concept, we can treat sticker systems as special extended Watson-Crick finite automata. In [PRS98, Lemma 5.8] there is shown the equivalence of Watson-Crick finite automata and (0.2)-headed simple multihead finite automata. By generalization of [PRS98, Lemma 5.8] to extended Watson-Crick finite automata and restriction to sticker systems, we get the following proof.

Let $\gamma = (V, \rho, A, D)$ be a sticker system with $\rho = \{(x,x) : x \in V\}$. The (2.2)-headed oneway multihead finite automaton $\mathcal{A} = (V, Z, s, F, T)$ works as follows:

[6] The language families *NL* and *NLINSPACE* are not defined in this publication, because they are needed only here. In short, these are families of languages, which can be accepted by *Turing machines* with one two-way input tape and one logarithmical or rather linear space bounded work tape.

Initialization

All four heads are located at the absolute position 1. It will be gone to the acceptance phase, if all four heads read the letter ε and there is $\varepsilon \in L(\gamma)$. Otherwise, a random position p will be guessed, head *top-left* (1) and *bottom-left* (2) will be placed at position p, head *top-right* (3) and *bottom-right* (4) will be placed at position $p+1$ and it will be gone to the phase of axiom selection.

Axiom selection

An arbitrary axiom $x_1x_2x_3 \in A$ will be chosen and saved in the form $(x_1x_2\binom{\varepsilon}{\varepsilon}, x_3)$. Thereafter, it will be gone to the phase of rule and axiom check.

Rule selection

An arbitrary rule $(x, y) \in D$ will be chosen and saved. Thereafter, it will be gone to the phase of rule and axiom check.

Rule and axiom check

Let the saved rule or axiom be (x, y). First, it will be considered domino $y \in W_\rho(V)$:

(1) If $y = \binom{u}{v} \in S_\rho(V)$, head *top-right* moves by $|u|$ steps to the right and ensures, that thereby word u is read. Analogously, head *bottom-right* moves by $|v|$ steps to the right and ensures, that thereby word v is read. Informations about relative positions are irrelevant.

(2) If $y = y_1y_2y_3 \in LR_\rho(V)$, then it will be first considered y_1, then y_2 and then y_3. For y_1 and y_3 it will be proceeded analogous to (1). Let $y_2 = \begin{bmatrix} u \\ u \end{bmatrix}_\rho$. Head *top-right* and *bottom-right* move by $|u|$ steps to the right and ensure, that thereby word u is read. Additionally, it will be ensured, that both heads are located one upon the other during the whole check of y_2.

Domino x is checked symmetrically to y, thereby head *top-left* and *bottom-left* moves at the basis of x to the left. Thereafter, it will be gone to the phase of acceptance or the phase of rule selection.

Acceptance

The input will be accepted, if all four heads read the letter ε.

Let $C(\gamma) := \{(x, y, z) : x \in S_\rho(V), y \in LR_\rho(V), z \in S_\rho(V), x \cdot y \cdot z \in WK_\rho(V)\}$ be the set of *configurations* of γ. Let $A := C(\gamma) \cap (S_\rho(V) \times C^*(\gamma) \times S_\rho(V))$ and $B := C^*(\mathcal{A}) \cap (V^* \times \{q\} \times \mathbb{N}^4)$ be two sets with q as exit point of the phase of rule and axiom check. Let $\lambda : C(\gamma) \to C(\mathcal{A})$ be a function, which builds a configuration of the multihead finite automaton $\mathcal{A}$ from a configuration of the sticker system γ with

$$\lambda\left((x,y,z)\right) := \left(x^t \cdot y^t \cdot z^t, q, \left(|x^t|, |x^b|, |x^t \cdot y^t| + 1, |x^b \cdot y^b| + 1\right)\right).$$

One can now prove the relation $\lambda(A) = B$ by induction over the count of rule usages and thereby show, that every derivation of the sticker system γ can be simulated by the automaton $\mathcal{A}$ and on the other hand every run of the automaton $\mathcal{A}$ can be considered as an simulation of a derivation of the sticker system γ. By including the acceptance condition and the behavior on the case $\varepsilon \in L(\gamma)$ we get $L(\mathcal{A}) = L(\gamma)$.

Remark 5.2 *Probably,* (2.2)*-headed oneway multihead finite automata are able to accept more than sticker systems, because rule usage can be better controlled by states. However, sticker systems are evidently very similar to* (2.2)*-headed oneway multihead finite automata.*

Conclusion 5.3 $SSL(n) \subseteq SOHL_{2.2}$.

Proof. Theorem 5.1 gives a proof for the inclusion $ASL(n) \subseteq OHL_{2.2}$. By restriction to simple sticker languages, we don't need the capability of relative head position detection any more. Consequently the (2.2)-headed oneway multihead finite automaton constructed there is simple. Thus, it is $SSL(n) \subseteq SOHL_{2.2}$.

Theorem 5.4 $OSL(n) \subseteq OHL_2$.

Proof. Let $\gamma = (V, \rho, A, D)$ be a sticker system with $\rho = \{(x,x) : x \in V\}$. Because of $OSL(n) \subseteq ASL(n)$ and according to Theorem 5.1, for every one-sided sticker system exists an equivalent (2.2)-headed oneway multihead finite automaton. By reconstructing this automaton we get the following (0.2)-headed oneway multihead finite automaton $\mathcal{A} = (V, Z, s, F, T)$:

Initialization

> Both heads are located at the absolute position 1. It will be gone to the acceptance phase, if both heads read the letter ε and there is $\varepsilon \in L(\gamma)$. Otherwise, it will be gone to the phase of left-sided rule selection or axiom selection.

Left-sided Rule selection

> An arbitrary left-sided rule $(x, \varepsilon) \in D$ will be chosen and saved. Thereafter, it will be gone to the phase of rule and axiom check.

Right-sided Rule selection

An arbitrary right-sided rule $(\varepsilon, x) \in D$ will be chosen and saved. Thereafter, it will be gone to the phase of rule and axiom check.

Axiom selection

An arbitrary axiom $x \in A$ will be chosen and saved in the form (ε, x). Thereafter, it will be gone to the phase of rule and axiom check.

Rule and axiom check

Let the saved rule or axiom be (x, ε) or (ε, x) with $x \in W_\rho(V)$:

(1) If $x = \binom{u}{v} \in S_\rho(V)$, head *top* moves by $|u|$ steps to the right and ensures, that thereby word u is read. Analogously, head *bottom* moves by $|v|$ steps to the right and ensures, that thereby word v is read. Informations about relative positions are irrelevant.

(2) If $x = x_1x_2x_3 \in LR_\rho(V)$, then it will be first considered x_1, then x_2 and then x_3. For x_1 and x_3 it will be proceeded analogous to (1). Let $x_2 = \begin{bmatrix} u \\ u \end{bmatrix}_\rho$. Head *top* and *bottom* move by $|u|$ steps to the right and ensure, that thereby word u is read. Additionally, it will be ensured, that both heads are located one upon the other during the whole check of x_2.

Thereafter, it will be gone to the phase of left-sided rule selection, axiom selection, right-sided rule selection or acceptance. It can only be gone to the phase of left-sided rule selection or axiom selection, if the saved rule was left-sided. Analogously, it can only be gone to the phase of right-sided rule selection or acceptance, if the saved rule was right-sided or an axiom.

Acceptance

The input will be accepted, if both heads read the letter ε.

Similarly to Theorem 5.1 one can show the relation $L(\gamma) = L(\mathcal{A})$ and therewith $OSL(n) \subseteq OHL_2$.

Conclusion 5.5 *$SRSL(n) \subseteq SOHL_2$.*

Proof. Theorem 5.4 gives evidence to $OSL(n) \subseteq OHL_2$. By restricting to simple one-sided sticker languages, the relative head position detection will only be needed for the axiom check. By restricting to simple right-sided sticker systems, we don't need this capability any longer, because the axiom selection and check are done directly after the initialization phase. Thus we get $SRSL(n) \subseteq SOHL_2$.

6 Conclusions

Now we will give some results, which are direct or indirect conclusions of the previous section. This collection isn't complete.

Corollary 6.1 $SOHL_{1.1} = LIN \not\subseteq OSL(n)$.

Proof. Because of Theorem 4.4, Theorem 4.5, definition of multihead language families and Theorem 5.4 we get $SOHL_{1.1} = LIN \not\subseteq OHL_* \supseteq OHL_2 \supseteq OSL(n)$.

Corollary 6.2 $CF \not\subseteq ASL(n)$, $SHL_2 \not\subseteq ASL(n)$.

Proof. Because of Theorem 4.5, definition of multihead language families and Theorem 5.1 we get $CF \not\subseteq OHL_{*.*}$ or rather $SHL_2 \not\subseteq OHL_{*.*}$ and $OHL_{*.*} \supseteq OHL_{2.2} \supseteq ASL(n)$.

As a conclusion of Corollary 6.2 we get $ASL(n) \subset CS$. That means, there exists at most one (known) Chomsky language, which is not a sticker language. The following result even shows, that only a few Chomsky languages are sticker languages.

Corollary 6.3 $ASL(n) \subset CS$.

Proof. It is $ASL(n) \subseteq OHL_{2.2} \subset AHL_4 \subset AHL_* \subset CS$ because of Theorem 5.1, definition of multihead language families, Theorem 4.5, Theorem 4.1 and Theorem 4.4.

Conclusion 6.4 *The language families $cASL(x)$ and $ASL(x)$ with $x \in \{b, n\}$ are not closed under concatenation.*

Proof. Because of [WK05, Theorem 4.1] and Corollary 6.2 the termed sticker language families contain $S_1 := \left\{w \in \{a,b\}^* : w = w^R\right\}$ but they don't contain $S_2 := S_1 \cdot S_1 = \left\{v \cdot w \in \{a,b\}^* : v = v^R, w = w^R\right\}$.

7 Acknowledgments

I am grateful to Ludwig Staiger, Dietrich Kuske and Jens Keilwagen for their helpful advices and suggestions.

References

[Adl94] L. M. Adleman. *Molecular computation of solutions to combinatorial problems.* Science, Vol. 226, November 1994, 1021–1024.

[KPG98] L. Kari, G. Păun, G. Rozenberg, A. Salomaa, S. Yu. *DNA-Computing, sticker systems and universality.* Acta Informatica, 35, 5, 1998, 401–420.

[FPR98] R. Freund, G. Păun, G. Rozenberg, A. Salomaa. *Bidirectional sticker systems.* Third Annual Pacific Conference on Biocomputing, Hawaii, 1998 / World Scientific, Sigapore, 1998, 535–546.

[PR98] G. Păun, G. Rozenberg. *Sticker systems.* Theoretical Computer Science, 204, 1998, 183–203.

[PRS98] G. Păun, G. Rozenberg, A. Salomaa. *DNA-Computing. New Computing Paradigms.* Springer, Berlin Heidelberg, 1998.

[Wei04] Peter Weigel. *Ausdrucksstärke von Stickersystemen. Untersuchung der Ausdrucksstärke von Stickersystemen durch Vergleich mit Chomskygrammatiken und Mehrkopfautomaten.* Diplomarbeit, Martin-Luther-Universität Halle-Wittenberg, Institut für Informatik, Halle/Saale, Oktober 2004.

[KW04] D. Kuske, P. Weigel. *The rôle of the complementarity relation in Watson-Crick automata and sticker systems.* Developments in Language Theory: 8th International Conference, DLT 2004. Auckland, New Zealand, December 13-17. Proceedings. / Lecture Notes in Computer Science, Springer, Heidelberg, 3340, 2004, 272 – 283.

[WK05] P. Weigel, J. Keilwagen. *Incomparability of simple and one-sided/regular sticker languages. www.stickersysteme.de, Februar 2005.*

[Ros66] A. L. Rosenberg. *On multihead finite automata.* IBM J.R. and D., 10, 1966, 388–394.

[Har72] J. Hartmanis. *On Non-Determinancy in Simple Computing Devices.* Acta Informatica, 1, 1972, 336-344.

[Iba73] O. H. Ibarra. *On Two-Way Multihead Automata.* Journal of Computer and System Sciences, 7, 1, February 1973, 28–36.

[IK75] O. H. Ibarra, C. E. Kim. *On 3-head vesus 2-head finite automata.* Inform. Control, 4, 1975, 193–200.

[YR78] A. C. Yao, R. L. Rivest. *$k+1$ heads are better then k.* Journal of then ACM, 25, 2, April 1978, 337–340.

[Mon80] B. Monien. *Two-Way Multihead Automata Over A One-Letter Alphabet.* R.A.I.R.O. Theoretical Informatics, 14, 1, 1980, 67–82.

[Hro83] J. Hromkovic. *One-way multihead deterministic finite automata.* Acta Informatica, 19, 4, 1983, 377–384.

[DH83] P. Duris, J. Hromkovic. *One-way simple multihead finite automata are not closed under concatenation.* Theoretical Computer Science, 27, 1983, 121–125.

[Sem04] J.M. Sempere. *A representation theorem for languages accepted by Watson-Crick finite automata.* EATCS Bulletin 83, June 2004, 187–191.

[WW86] K. Wagner, G. Wechsung. *Computational Complexity.* Deutscher Verlag der Wissenschaften, Berlin, 1986.