Annales Mathematicae et Informaticae
Accepted manuscript
DOI: 10.33039/ami.2023.08.004
URL: https://ami.uni-eszterhazy.hu

# On language classes accepted by stateless $5' \to 3'$ Watson-Crick finite automata

**Benedek Nagy**[ab]

[a]Eastern Mediterranean University,
Faculty of Arts and Sciences,
Department of Mathematics,
Famagusta, North Cyprus, Mersin-10, Turkey

[b]Department of Computer Science,
Institute of Mathematics and Informatics,
Eszterházy Károly Catholic University,
Eger, Hungary
nbenedek.inf@gmail.com

**Abstract.** Watson-Crick automata are belonging to the natural computing paradigm as these finite automata are working on strings representing DNA molecules. Watson-Crick automata have two reading heads, and in the $5' \to 3'$ models these two heads start from the two extremes of the input. This is well motivated by the fact that DNA strands have $5'$ and $3'$ ends based on the fact which carbon atoms of the sugar group is used in the covalent bonds to continue the strand. However, in the two stranded DNA, the directions of the strands are opposite, so that, if an enzyme would read the strand it may read each strand in its $5'$ to $3'$ direction, which means physically opposite directions starting from the two extremes of the molecule. On the other hand, enzymes may not have inner states, thus those Watson-Crick automata which are stateless (i.e. have exactly one state) are more realistic from this point of view. In this paper these stateless $5' \to 3'$ Watson-Crick automata are studied and some properties of the language classes accepted by their variants are proven. We show hierarchy results, and also a "pumping", i.e., iteration result for these languages that can be used to prove that some languages may not be in the class accepted by the class of stateless $5' \to 3'$ Watson-Crick automata.

*Keywords:* Bio-computing, stateless finite automata, linear languages

*AMS Subject Classification:* AMS 68Q45: Formal languages and automata, 68Q07 Biologically inspired models of computation

Submitted: July 14, 2023
Accepted: August 7, 2023
Published online: August 16, 2023

# 1. Introduction

Finite automata are one of the oldest models of computing. The classes of both deterministic and nondeterministic variants are able to accept exactly the class of regular languages. Amar and Putzolu have generalised the concept and they have defined a special class of linear context-free language class, the so-called even-linear grammars and languages [1, 2].

Finite automata are very popular since they are very simple comparing them to other more sophisticated models. During the last decades, many kinds of extensions of finite automata are studied and proven to be applicable to accept larger classes of languages than the class of regular languages, but still have a moderate complexity. One of the branches of DNA computing is working with automata models accepting DNA molecules (or their formal representations), these automata are named as Watson-Crick automata [3, 6, 28]. These automata have two reading heads, one for each strand of the double stranded DNA. They can be used also for bioinformatic problems [30]. On the other hand, the strings have two extremes, namely their beginning and ends, which gives the rise of the 2-head models processing the input from their beginning and their end in a kind of parallel manner [13, 16]. Some of these 2-head models are known as $5' \to 3'$ Watson-Crick automata by a biological motivation describing these models to accept DNA molecules instead of ordinary words [17–19, 24, 26]. As usual at Watson-Crick automata, various extensions/restrictions could be applied on the model, e.g., string reading feature, or having only accepting states or having only one state. Generally, this 2-head model of computing, by finishing the computation at latest when the heads are in the same position, characterizes exactly the class of linear context-free languages [13, 16, 17, 26]. Regular-like expressions for linear context-free languages are shown in [29] to suggest the feeling that linear context-free languages can really be imagined as a superclass of the regular languages. Special variants capable to accept some special subclasses of the class of linear context-free languages of these automata models are also studied, e.g., the so-called even-linear languages (see, e.g., their importance in various applications [31, 32]) are accepted by a model, the so-called both-head stepping $5' \to 3'$ Watson-Crick finite automata, in which the two heads must move together in a synchronous way [14]. On the other hand, opposite to the ordinary finite state automata, the deterministic counterpart of the 2-head model is weaker, and the language class 2detLIN is accepted by them [25, 27]. Recently two more variants of the model have been investigated: In the state deterministic $5' \to 3'$ Watson-Crick automata the state of the next configuration depends only on the actual state and it does not depend on the read symbol(s) [22]. These automata can easily be characterized by their graphs. On the other hand, in quasi-deterministic $5' \to 3'$ Watson-Crick automata, in any computation, the state of the next configuration is deterministically computed, however the configuration itself is not [21]. These automata behave somewhat between the classical deterministic and nondeterministic models of the $5' \to 3'$ Watson-Crick automata. In [12], the non-sensing $5' \to 3'$ Watson-Crick automata are studied, in which both heads read the entire input (but, of course, in opposite direction). It was shown that Turing machine computations can be coded into the input which gives, on the one hand, a characterization of the class of recursively enumerable class of languages by the model, and on the other hand, the undecidability of some of the simple problems for the accepted languages. Moreover, infinite hierarchies of language classes were shown according to the number of allowed runs on the input, when in each run the entire input is processed by both heads.

From a biological point of view, the stateless variants, i.e., $5' \rightarrow 3'$ Watson-Crick automata with a sole state make more sense than models with several states. Thus, in this paper, we consider these variants. Usually, finite state automata store the information about the already processed input in their states. To store one bit information, one needs two states. Automata with a sole state referred as stateless, as they cannot store any information in their state. Thus, usually, not to allow to accept all possible inputs, these automata are incomplete, in the sense that they are not accepting those inputs that cannot be processed, i.e., they get stuck during (all) the computation(s) on these inputs. Actually, the things are a little bit more complex here, since actually, stateless automata at least have the information that the already processed part of the input can be processed.

We recall the well-known fact that stateless automata are as efficient as automata with any finitely many number of states in case of pushdown automata [9]. Furthermore, a similar fact has been proven for the $5' \rightarrow 3'$ Watson-Crick pushdown automata [15]. On the other hand, stateless variants of $5' \rightarrow 3'$ Watson-Crick multicounter machines were studied in [4, 5, 7, 8] by obtaining various hierarchies of the accepted language classes.

Pumping and iteration lemmas are well-known for various subclasses of the context-free languages [9]. In general, they give necessary conditions for the languages belonging to a given class, and thus, by their help, we may prove that a given language is definitely not belonging to the class we are interested in. They are usually proven by considering derivation trees, or for many subclasses, including, e.g., the class of regular languages, by arguments based on the finite automata model. There are variants of these theorems for some special subclasses of the class of linear context-free languages [10, 20]. In this paper, as one of our main results, we provide an iteration result for the languages of stateless $5' \rightarrow 3'$ Watson-Crick automata.

In the next section, we formally define our model. In Section 3 we show some examples. In Section 4 we present our main theorems and also we show how they can be applied. Conclusions and some future topics of research close the paper.

## 2. Formal definitions

In this section we define formally our model. We note here that in the literature, the definition may also include the so-called Watson-Crick complementarity relation defined on the alphabet. Since in the nature, it is a symmetric bijective relation, we simplify our model not to play with it. This can be done, since, on the one hand, in [11] it is shown that at Watson-Crick automata this relation does not play any important role, as the same language class can be accepted by using the identity instead of a more general complementarity relation. On the other hand, as we will see, in the sensing $5' \to 3'$ Watson-Crick automata, every position of the double stranded DNA is read by at most one of the heads, and thus, the relation on the letters at the same position of the two strands cannot play any role in the accepted languages.

**Definition 2.1.** A Watson-Crick finite automaton (a WK automaton) is a 5-tuple $A = (T, Q, q_0, F, \delta)$, where:

- $T$ is the (input) alphabet, (e.g., the letters standing for possible bases of the nucleotides),
- the finite set of states $Q$, the initial state $q_0 \in Q$ and the set of final (also called accepting) states $F \subseteq Q$,
- the transition mapping $\delta$ is of the form $\delta : Q \times T^* \times T^* \to 2^Q$, such that it is non-empty only for finitely many triplets $(q, u, v), q \in Q, u, v \in T^*$.

The computation by WK automata goes through configurations as follows.

**Definition 2.2.** A configuration is a pair $(q, w)$ containing $q$, the current state and $w$, the unprocessed part of the input.

In sensing $5' \to 3'$ WK automata, for any $w', x, y \in T^*, q, q' \in Q$, we write a step of the computation between two configurations as follows: $(q, xw'y) \Rightarrow (q', w')$ if and only if $q' \in \delta(q, x, y)$.

We denote the reflexive and transitive closure of the relation $\Rightarrow$ by $\Rightarrow^*$, and this is the computation relation on configurations.

Further, for an input $w \in T^*$, an accepting computation is a sequence of steps $(q_0, w) \Rightarrow^* (q_f, \lambda)$ for some $q_f \in F$.

As usual, we use automata for accepting languages, thus we have:

**Definition 2.3.** The language accepted by a sensing $5' \to 3'$ WK automaton consists of all words that are accepted by the automaton.

By comparing traditional finite state automata with sensing $5' \to 3'$ WK automata, there are two main differences. Both of those can be seen in the transition function. The first difference, as we have already mentioned, is that the sensing $5' \to 3'$ WK automata have two reading heads, thus the domain of the transition function contains triplets. The other difference, coming from biological motivations, is that the WK automata may read strings in a transition, not only letters. This is motivated by the fact that enzymes may be attached to the strands, and thus, read a longer part of the input in a computation step. On the other hand, to keep the model still finite, it is allowed to have transitions only for finitely many triplets of the domain, since it is not feasible to allow to read strings with an unlimited length, as enzymes must also have a finite size.

The above definitions can be used in general for any $5' \to 3'$ WK automata. However, there are some restricted variants, and actually, in this paper, we are focusing on some of these variants.

**Definition 2.4.** A Watson-Crick finite automaton is STATELESS if $Q = F = \{q_0\}$.

A Watson-Crick finite automaton is SIMPLE if $\delta : (Q \times ((\lambda, T^*) \cup (T^*, \lambda))) \to 2^Q$, i.e., at most one heads reads in a step.

A Watson-Crick finite automaton is ONE-LIMITED if $\delta : (Q \times ((\lambda, T) \cup (T, \lambda))) \to 2^Q$, i.e., exactly one letter is being read in each step.

The notation NWK is used for the stateless automata, as N stands for "no states". Further, the notation NSWK and N1WK is used for stateless simple and stateless one-limited automata, respectively.

By definition, clearly all N1WK automata are NSWK automata, and all NSWK automata are NWK automata at the same time.

We may also have other types of restrictions based on the sequences of computation steps:

**Definition 2.5.** A Watson-Crick finite automaton is DETERMINISTIC, if for any of its possible configurations there is at most one possible step to continue the computation.

A Watson-Crick finite automaton is STATE-DETERMINISTIC, if for each of its states $q \in Q$, if there is a transition from $q$ and it goes to state $p$, i.e., $p \in \delta(q, u, v)$, then every transition from $q$ goes to $p$.

A Watson-Crick finite automaton is QUASI-DETERMINISTIC, if for each possible configuration $(q, w)$, if $(q, w) \Rightarrow (p, u)$ and also $(q, w) \Rightarrow (r, v)$, then $p = r$ must hold.

We note here that in some cases, e.g., [13, 16] the 2-head automata are defined in a way that they may able to read the input only letter by letter. Generally, if the automaton could have many states, that is not a problem, the string-reading feature of our model can be resolved by adding some new states and doing the computation on the input letter by letter. This can be done also in the deterministic case as proven in [25]. On the other hand, if we consider only automata with a sole state, the string-reading feature becomes essential in our models. Without allowing to read strings in a transition only very limited number of languages would be accepted by 2-head stateless automata.

## 3. Examples

In this section, for better understanding these computational models, we give some examples.

**Example 3.1.** The regular language $(01)^*$ is accepted by the deterministic NSWK automaton with state $q$ having only transition $q \in \delta(q, 01, \lambda)$ (since the automaton has only a sole state, we briefly say that it has a transition with $(01, \lambda)$ without mentioning its state). This automaton uses only its left head during the entire computation on its input. Clearly the whole input can be processed if and only if it is in the regular language $(01)^*$.

**Example 3.2.** The deterministic NWK accepts the language $\{0^n1^{3n}\}$ having only transition by $(0, 111)$. In each computation step, the left head (starting from the beginning of the input) is reading a 0, while the right head (starting from the end of the input) is reading 111. Consequently, when the heads meet and the computation is finished, any nonempty input accepted must have the form that all 0s precede all the 1s, and the number of 1s is exactly three times as many as the number of 0s. This language is a non-regular linear context-free language.

**Example 3.3.** The deterministic NWK with two transitions $(0, 0)$ and $(1, 1)$ accepts the language of even palindromes over $\{0, 1\}$, i.e., the language $\{u \cdot u^R \mid u \in \{0, 1\}^*\}$ where $u^R$ is the reversal of the word $u$. This language is a well-known non-regular linear context-free language.

**Example 3.4.** The regular language $0^*1^*$ is accepted by the nondeterministic N1WK automaton having two transitions by $(0, \lambda)$ and by $(\lambda, 1)$. In each step of the computation, either the left head reads a 0 (from the beginning of the remaining input) or the right head reads a 1 (from the end of the remaining input). Observe that in fact, this automaton is not deterministic.

From the definitions of state-deterministic, quasi-deterministic and deterministic variants (see also [21, 22, 25, 27]) we can infer the following:

**Proposition 3.5.** *Every NWK automaton is state-deterministic and quasi-deterministic. Further, the class of NSWK automata coincides with the class of state-deterministic NSWK automata and also with the class of quasi-deterministic NSWK automata. Moreover, the class of N1WK automata coincides with the class of state-deterministic N1WK automata and with the class of quasi-deterministic N1WK automata.*

On the other hand, based on the examples shown above, we can infer also the following result about these models.

**Proposition 3.6.** *There are NWK, NSWK and N1WK automata that are not deterministic.*

Thus actually, we can consider six classes of stateless $5' \to 3'$ WK automata in the sequel. We show the hierarchy of the language classes accepted by them in Figure 1. However, to put also the class of regular languages into this hierarchy we may use our new results presented in the next section.

# 4. Main results

In this section, we concentrate on the NWK automata in general, thus the results of this section are applicable for each of the above mentioned subcases of the model as well.

Based on the transitions used in an NWK automaton we can define some further concepts.

**Definition 4.1.** Let an NWK automaton $A = (T, \{q\}, q, \{q\}, \delta)$ be given. By definition, it has finitely many transitions $\delta(q, \ell_i, r_i) = \{q\}$ defined, let denote this number by $n$. Let us have an alphabet $V = \{v_1, \ldots, v_n\}$ with $n$ elements, and let us assign the elements of $V$ to the transitions of the automaton in a bijective way: $v_i \leftrightarrow (\ell_i, r_i)$. Let $\varphi, \mu : V \to T^*$ be the mappings defined as $\varphi(v_i) = \ell_i$ and analogously, $\mu(v_i) = r_i^R$, where $R$ stand for the reversal of the word.

We refer to $\varphi$ and $\mu$ as the FORWARD and the BACKWARD MORPHISMS of the automaton $A$ and its accepted language $L$.

Now we are ready to claim one of our new results about the languages accepted by these models.

**Theorem 4.2.** *Let $A$ be an NWK automaton over alphabet $T$. Then there is a finite alphabet $V$, and there exist the forward and backward morphisms $\varphi, \mu : V \to T^*$ such that the language accepted by $A$ can be written as $\{\varphi(x)\mu(x^R) \mid x \in V^*\}$, where $x^R$ is the reversal of the word $x$.*

***Proof.*** Clearly, for each word $w \in T^*$ accepted by the automaton, there exists an accepting computation that can be described by the sequence of transitions $x \in V^*$. Moreover, in stateless automata every word of $V^*$ is describing an accepting computation (of some input word $w$). In this computation the left head is reading the word defined by $\varphi(x)$ and as the right head is reading from the right, it is reading $\mu(x^R)$ during the computation. □

Furthermore, we state the following “pumping”-like theorem.

**Theorem 4.3.** *Let $A$ be an NWK automaton over $T$. For any word $w$ accepted by $A$, there is a factorisation $w = u \cdot v$, such that $u^i v^i$ is also accepted by $A$ for any $i \in \mathbb{N}$.*

***Proof.*** Let us consider any word $w$ accepted by $A$. Then, by Theorem 4.2, an/the accepting computation on $w$ can be described by $x \in V^*$. Further, $w = \varphi(x)\mu(x^R)$ with the associated morphisms. Considering the words of the form $x^i \in V^*$, they describe accepting computations of the words of the form $\varphi(x^i) \cdot (\mu((x^i)^R) = (\varphi(x))^i \cdot (\mu(x^R))^i$ which, with the choice of $u = \varphi(x)$ and $v = \mu(x^R)$, can be written as $u^i v^i$ as the theorem states. □

The theorem can also be seen as follows: the repetition of the computation implies a kind of insertion operation on the accepted words.

As we have seen, there are some non-regular languages that are accepted with deterministic NWK automata. Let us see an example what Theorem 4.3 means for an accepted language.

**Example 4.4.** Let us consider the language $L$ of even palindromes shown in Example 3.3. Let $V = \{a, b\}$, $\varphi(a) = 0$, $\varphi(b) = 1$ and $\mu(a) = 0$, $\mu(b) = 1$. Then

$w = 00100100 \in L$, and in fact, $x = aaba \in V^*$ has the property that $\varphi(x) = 0010$ and $\mu(x^R) = 0100$. Therefore $w = \varphi(x) \cdot \mu(x^R)$. We may obtain the words
$(\varphi(x))^2 \cdot (\mu(x^R))^2 = 0010001001000100$,
$(\varphi(x))^3 \cdot (\mu(x^R))^3 = 001000100010010001001000$,
$(\varphi(x))^i \cdot (\mu(x^R))^i$ (for any $i \in \mathbb{N}$) based on the words $x^2, x^3$, $x^i$.

On the other hand, now we present another possible, maybe more useful application of Theorem 4.3.

**Proposition 4.5.** *The regular language $a^*bba^*$ is not accepted by any NWK automata.*

***Proof.*** As $a^*bba^*$ does not satisfy the conditions of the previous theorem, as the number of $b$s cannot be "pumped", thus, obviously it cannot be a language that is accepted by any NWK automata. □

Thus, our result can efficiently be used to show that some languages are not acceptable by any NWK automata. From, e.g., Example 3.3 and Proposition 4.5 we can infer the incomparability of the class of regular languages and the class of languages accepted by stateless WK automata under set theoretical inclusion.

# 5. Conclusion and future work

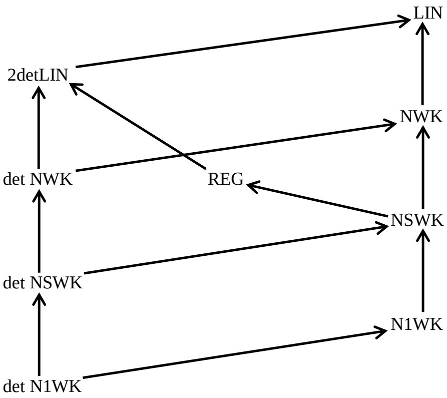

**Figure 1.** A hierarchy of language classes accepted by stateless sensing $5' \to 3'$ WK automata.

Figure 1 shows the hierarchy of the language classes of our model in a Hasse diagram (the automaton class here denoting the accepted language class). Classes not having directed path between them are incomparable under set theoretic inclusion relation. REG denotes the class of regular, LIN, the class of linear context-free languages (this is the class that is accepted by sensing $5' \to 3'$ WK automata) and 2detLIN the class of languages accepted by deterministic sensing $5' \to 3'$ WK automata.

In this paper, we have shown a new iteration theorem for the languages accepted by stateless $5' \to 3'$ Watson-Crick automata. This theorem is based on two newly defined morphisms. This approach could also be fruitful to analyse further properties of the corresponding language classes. We recall that a somewhat related topic, finite state $5' \to 3'$ Watson-Crick transducers (automata with output) were discussed in [23], where the description was also used some special functions that can be in relation to our newly defined morphisms.

It is a task of a future work to develop other specific iteration theorems and for other specific classes of languages accepted by variants of Watson-Crick automata and to describe some new properties of those language classes based on these and related results.

## References

[1] V. Amar, G. R. Putzolu: *Generalizations of Regular Events*, Inf. Control. 8.1 (1965), pp. 56–63, doi: 10.1016/S0019-9958(65)90275-5.

[2] V. Amar, G. R. Putzolu: *On a Family of Linear Grammars*, Inf. Control. 7.3 (1964), pp. 283–291, doi: 10.1016/S0019-9958(64)90294-3.

[3] E. Czeizler, E. Czeizler: *A Short Survey on Watson-Crick Automata*, Bull. EATCS 88 (2006), pp. 104–119.

[4] Ö. Egecioglu, L. Hegedüs, B. Nagy: *Hierarchies of Stateless Multicounter $5' \to 3'$ Watson-Crick Automata Languages*, Fundam. Informaticae 110.1-4 (2011), pp. 111–123, doi: 10.3233/FI-2011-531.

[5] Ö. Egecioglu, L. Hegedüs, B. Nagy: *Stateless multicounter $5' \to 3'$ Watson-Crick automata*, in: Fifth International Conference on Bio-Inspired Computing: Theories and Applications, BIC-TA 2010, University of Hunan, Liverpool Hope University, Liverpool, United Kingdom / Changsha, China, September 8-10 and September 23-26, 2010, IEEE, 2010, pp. 1599–1606, doi: 10.1109/BICTA.2010.5645263.

[6] R. Freund, G. Paun, G. Rozenberg, A. Salomaa: *Watson-Crick finite automata*, in: DNA Based Computers, Proceedings of a DIMACS Workshop, Philadelphia, Pennsylvania, USA, June 23-25, 1997, ed. by H. Rubin, D. H. Wood, vol. 48, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, DIMACS/AMS, 1997, pp. 297–327, doi: 10.1090/dimacs/048/22.

[7] L. Hegedüs, B. Nagy: *On String Reading Stateless Multicounter $5' \to 3'$ Watson-Crick Automata - (Extended Abstract)*, in: Unconventional Computation and Natural Computation - 12th International Conference, UCNC 2013, Milan, Italy, July 1-5, 2013. Proceedings, ed. by G. Mauri, A. Dennunzio, L. Manzoni, A. E. Porreca, vol. 7956, Lecture Notes in Computer Science, Springer, 2013, pp. 257–258, doi: 10.1007/978-3-642-39074-6_29.

[8] L. Hegedüs, B. Nagy, Ö. Egecioglu: *Stateless multicounter $5' \to 3'$ Watson-Crick automata: the deterministic case*, Nat. Comput. 11.3 (2012), pp. 361–368, doi: 10.1007/s11047-011-9290-9.

[9] J. E. Hopcroft, J. D. Ullman: *Introduction to Automata Theory, Languages and Computation, Second Edition*, Addison-Wesley, 2000.

[10] G. Horváth, B. Nagy: *Pumping lemmas for linear and nonlinear context-free languages*, Acta Univ. Sapientiae, Informatica 2.2 (2010), pp. 194–209, arXiv: 1012.0023, url: http://arxiv.org/abs/1012.0023.

[11] D. Kuske, P. Weigel: *The Role of the Complementarity Relation in Watson-Crick Automata and Sticker Systems*, in: Developments in Language Theory, 8th International Conference, DLT 2004, Auckland, New Zealand, December 13-17, 2004, Proceedings, ed. by C. Calude, E. Calude, M. J. Dinneen, vol. 3340, Lecture Notes in Computer Science, Springer, 2004, pp. 272–283, doi: 10.1007/978-3-540-30550-7_23.

[12] P. Leupold, B. Nagy: *$5'\to 3'$ Watson-Crick AutomataWith Several Runs*, Fundam. Informaticae 104.1-2 (2010), pp. 71–91, doi: 10.3233/FI-2010-336.

[13] R. Loukanova: *Linear Context Free Languages*, in: Theoretical Aspects of Computing - ICTAC 2007, 4th International Colloquium, Macau, China, September 26-28, 2007, Proceedings, ed. by C. B. Jones, Z. Liu, J. Woodcock, vol. 4711, Lecture Notes in Computer Science, Springer, 2007, pp. 351–365, doi: 10.1007/978-3-540-75292-9_24.

[14] B. Nagy: $5'\to 3'$ *Sensing Watson-Crick Finite Automata*, in: Sequence and Genome Analysis II – Methods and Applications, ed. by G. Fung, iConcept Press, 2010, pp. 39–56.

[15] B. Nagy: *$5'\to 3'$ Watson-Crick pushdown automata*, Inf. Sci. 537 (2020), pp. 452–466, doi: 10.1016/j.ins.2020.06.031.

[16] B. Nagy: *A class of 2-head finite automata for linear languages*, Triangle 8 (Languages. Mathematical Approaches) (2012), pp. 89–99.

[17] B. Nagy: *On* $5'\to 3'$ *Sensing Watson-Crick Finite Automata*, in: DNA Computing, 13th International Meeting on DNA Computing, DNA13, Memphis, TN, USA, June 4-8, 2007, Revised Selected Papers, ed. by M. H. Garzon, H. Yan, vol. 4848, Lecture Notes in Computer Science, Springer, 2008, pp. 256–262, doi: 10.1007/978-3-540-77962-9_27.

[18] B. Nagy: *On a hierarchy of* $5'\to 3'$ *sensing Watson-Crick finite automata languages*, J. Log. Comput. 23.4 (2013), pp. 855–872, doi: 10.1093/logcom/exr049.

[19] B. Nagy: *On a hierarchy of* $5'\to 3'$ *sensing WK finite automata languages*, in: Mathematical Theory and Computational Practice, CiE, Abstract Booklet, Heidelberg, Germany, 2009, pp. 266–275.

[20] B. Nagy: *Pumping lemmas for special linear languages*, in: ICAI 2010, Eger, Hungary, vol. II. 2010, pp. 73–81.

[21] B. Nagy: *Quasi-deterministic* $5'\to 3'$ *Watson-Crick Automata*, in: Proceedings 12th International Workshop on Non-Classical Models of Automata and Applications, NCMA 2022, Debrecen, Hungary, August 26-27, 2022, ed. by H. Bordihn, G. Horváth, G. Vaszil, vol. 367, EPTCS, 2022, pp. 160–176, doi: 10.4204/EPTCS.367.11.

[22] B. Nagy: *State-deterministic* $5'\to 3'$ *Watson-Crick automata*, Nat. Comput. 20.4 (2021), pp. 725–737, doi: 10.1007/s11047-021-09865-z.

[23] B. Nagy, Z. Kovács: *On deterministic 1-limited $5'\to 3'$ sensing Watson-Crick finite-state transducers*, RAIRO Theor. Informatics Appl. 55 (2021), p. 5, doi: 10.1051/ita/2021007.

[24] B. Nagy, S. Parchami: $5'\to 3'$ *Watson-Crick automata languages-without sensing parameter*, Nat. Comput. 21.4 (2022), pp. 679–691, doi: 10.1007/s11047-021-09869-9.

[25] B. Nagy, S. Parchami: *On deterministic sensing* $5'\to 3'$ *Watson-Crick finite automata: a full hierarchy in 2detLIN*, Acta Informatica 58.3 (2021), pp. 153–175, doi: 10.1007/s00236-019-00362-6.

[26] B. Nagy, S. Parchami, H. M. M. Sadeghi: *A New Sensing* $5'\to 3'$ *Watson-Crick Automata Concept*, in: Proceedings 15th International Conference on Automata and Formal Languages, AFL 2017, Debrecen, Hungary, September 4-6, 2017, ed. by E. Csuhaj-Varjú, P. Dömösi, G. Vaszil, vol. 252, EPTCS, 2017, pp. 195–204, doi: 10.4204/EPTCS.252.19.

[27] S. Parchami, B. Nagy: *Deterministic Sensing* $5'\to 3'$ *Watson-Crick Automata Without Sensing Parameter*, in: Unconventional Computation and Natural Computation - 17th International Conference, UCNC 2018, Fontainebleau, France, June 25-29, 2018, Proceedings, ed. by S. Stepney, S. Verlan, vol. 10867, Lecture Notes in Computer Science, Springer, 2018, pp. 173–187, doi: 10.1007/978-3-319-92435-9_13.

[28] G. Paun, G. Rozenberg, A. Salomaa: *DNA Computing - New Computing Paradigms*, Texts in Theoretical Computer Science. An EATCS Series, Springer, 1998, isbn: 978-3-540-64196-4, doi: 10.1007/978-3-662-03563-4.

[29] J. M. Sempere: *On a Class of Regular-like Expressions for Linear Languages*, J. Autom. Lang. Comb. 5.3 (2000), pp. 343–354, doi: 10.25596/jalc-2000-343.

[30] J. M. Sempere: *On the application of Watson-Crick finite automata for the resolution of bioinformatic problems*, in: Tenth Workshop on Non-Classical Models of Automata and Applications, NCMA 2018, Košice, Slovakia, August 21-22, 2018, ed. by R. Freund, M. Hospodár, G. Jirásková, G. Pighizzini, Österreichische Computer Gesellschaft, 2018, pp. 29–30.

[31] J. M. Sempere, P. García: *A Characterization of Even Linear Languages and its Application to the Learning Problem*, in: Grammatical Inference and Applications, Second International Colloquium, ICGI-94, Alicante, Spain, September 21-23, 1994, Proceedings, ed. by R. C. Carrasco, J. Oncina, vol. 862, Lecture Notes in Computer Science, Springer, 1994, pp. 38–44, doi: 10.1007/3-540-58473-0_135.

[32] J. M. Sempere, P. García: *Learning Locally Testable Even Linear Languages from Positive Data*, in: Grammatical Inference: Algorithms and Applications, 6th International Colloquium: ICGI 2002, Amsterdam, The Netherlands, September 23-25, 2002, Proceedings, ed. by P. W. Adriaans, H. Fernau, M. van Zaanen, vol. 2484, Lecture Notes in Computer Science, Springer, 2002, pp. 225–236, doi: 10.1007/3-540-45790-9_18.