Two-way Nanoscale automata

Debayan Ganguly\textsuperscript{1}, Kingshuk Chatterjee\textsuperscript{2}, Kumar Sankar Ray\textsuperscript{3}

\textsuperscript{1} Government College of Engineering and Leather Technology, Block-LB, Sector –III, Kolkata-700106

\textsuperscript{2} Government College of Engineering and Ceramic technology, Kolkata-700010

\textsuperscript{3} Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata-700108

debayan3737@gmail.com\textsuperscript{1}, kingshukchatterjee@gmail.com\textsuperscript{2}, ksray@isical.ac.in\textsuperscript{3}

Abstract. In this paper, we show the all final subclass of two-way Watson-Crick automata have the same computational power as the classical two-way Watson-Crick automata. Here we compare the computational power of two-way Watson-Crick automata and two-way Quantum finite automata and we observe that two-way Watson-Crick automata can accept the language $L=\{ ww \mid w \in \{a,b\}^* \}$ which two way quantum finite automata cannot accept.

Keywords: quantum finite state automata, two-way quantum finite state automata, Watson-Crick automata, two-way Watson-Crick automata, non deterministic Watson Crick automata, one-way Watson crick quantum finite state.

1 Introduction

The number of transistors in a dense integrated circuit doubles approximately every eighteen months which is known as Moore's law [1]. It is predicted that nanoscale model, will replace the existing classical model in the coming years. Finite state automaton is one of the simplest models of computation for a classical computer. Similarly, quantum finite state automaton and Watson-Crick automata are the simplest nanoscale model for quantum and DNA computing respectively. The basic models of quantum finite state automata are discussed in [2-4]. In search of increasing the language accepting capabilities, different models of 1QFA have been proposed [5-13] by researchers. In 1997, Kondacs and Watrous [3] defined the quantum analogue of 2-way deterministic finite state automata named 2-way quantum finite automata (2QFA). It is more powerful than the 2-way deterministic finite automata. In 2QFA model, the tape head can read the input tape bi-directionally, or it can be stationary. It is more powerful than the classical model because it allows quantum parallelism with
the superposition of states on the input tape. Moreover, 2QFA recognizes non-context free languages in linear time, some context-free languages with one-sided error and regular languages.

The first such automata which exploited the DNA property were the Watson-Crick automata [14-15]. Watson-Crick automata are finite automata having two independent heads working on double strands where the characters on the corresponding positions of the two strands are connected by a complementarity relation similar to the Watson-Crick complementarity relation. The movement of the heads although independent of each other is controlled by a single state. Czeizler et.al. [16] introduced the deterministic variants of Watson Crick automata. State complexity of Watson-Crick automata is discussed in [17] and [18]. A two-way Watson-Crick automaton is similar in concept to a two-way finite automaton. The only difference between them is that in two-way Watson-Crick automaton the input tape is double stranded and the content of the second tape is determined in a similar manner as Watson-Crick automaton. The idea of two-way Watson-Crick automata (2NWK) were introduced in [18] but no formal structure is stated.

In this paper, we define subclasses of two-way Watson-Crick automata and show that the all final variant has the same computational power as the classical model. In the last section we show that Watson Crick automata and one-way Watson Crick quantum finite state automata accepts the context sensitive language $L = \{ww | w \in \{a, b\}^*\}$ which is not accepted by two-way quantum finite state automata.

## 2 Preliminaries and Definitions

In this Section, we state some important definitions related to the quantum finite automata and Watson-Crick automata.

**Definition 1:** Let $M = (Q, \Sigma, \delta, q_0, Q_a, Q_r)$ be a 2-way QFA consists of 6-tuple where $Q$ is a finite set of states, $\Sigma$ is an input alphabet, $\delta$ is a transition function, $q_0 \in Q$ is a starting state, $Q_a \subseteq Q$ and $Q_r \subseteq Q$ are sets of accepting and rejecting states.

The automaton halts when it reaches the set of states $Q_a$ or $Q_r$. It continues processing the input when it is in a state which belongs to the set of states $Q_{non}$. The symbols # and $ are used as the left and the right end marker to identify the beginning and ending input word. The working alphabet of $M$ is $\Gamma = \Sigma \cup \{#, \}$.

Transition function $\delta$ is defined as: $Q \times \Gamma \times Q \times D \rightarrow C$, where $\Gamma = \Sigma \cup \{#, \}$ and $D = \{\rightarrow, \uparrow, \rightarrow\}$ represent the left, stationary and right direction of tape head. Transition function must satisfy the following conditions: local probability and orthogonality condition, first separability condition and second separability condition respectively which are shown in [3].

**Definition 2:** Depending on the type of states and transition rules there are two types or subclasses of two-way quantum finite automata. A two-way quantum automaton $M = (Q, \Sigma, \delta, q_0, Q_a, Q_r)$ is
1) stateless(2 NQFA ): If it has only one state, i.e. Q=F=\{ q_0 \};
2) all-final(2 FQFA ): If all the states are final, i.e. Q=F;

Definition 3: The symbol V denotes a finite alphabet.

Definition 4: The set of all finite words over V is denoted by V^*, which includes the empty word \( \lambda \). The symbol V^*\neq V^* \{-\lambda\} denotes the set of all non-empty words over the alphabet V.

Definition 5: For w \in V^*, the length of w is denoted by |w|.

Definition 6: Let u \in V^* and v \in V^* be two words and if there is some word x \in V^*, such that v=ux, then u is a prefix of v, denoted by u \preceq v.

Definition 7: Two words, u and v are prefix comparable denoted by u\sim_p v, if u is a prefix of v or vice versa.

Definition 8: When u is not a prefix of v and v is not a prefix of u then u and v are not prefix comparable which is denoted by u\not\sim_p v.

Next we state the important models and definitions associated with Watson-Crick automata.

Definition 9: A non-deterministic Watson-Crick automaton is a 6-tuple of the form M=(V, \rho, Q, q_0, F, \delta) where V is an alphabet set, the symbol Q denotes the set of states, the symmetric complementarity relation \rho \subseteq V \times V is called the Watson-Crick complementarity relation, q_0 is the initial state and F \subseteq Q is the set of final states. The function \delta contains a finite number of transition rules of the form q(w_1, w_2) \rightarrow q', which denotes that the machine in state q parses w_1 in upper strand and w_2 in lower strand and goes to state q' where w_1, w_2 \in V^*. The symbol [w_1, w_2] is different from \( (w_1, w_2) \). While \( [w_1, w_2] \) is just a pair of strings written in that form instead of \( (w_1, w_2) \), the symbol [w_1, w_2] denotes that the two strands are of same length i.e. |w_1|=|w_2| and the corresponding symbols in two strands are complementarity in the sense given by the relation \rho. The symbol \( \left\lfloor V \right\rfloor_\rho = \{ [a, b] : a, b \in V, (a, b) \in \rho \} \) and \( WK_\rho(V) = \left\lfloor V \right\rfloor_\rho \) denotes the Watson-Crick domain associated with V and \rho.

A transition in a Watson-Crick finite automaton can be defined as follows:

For \( \left( x_1, x_2 \right), \left( u_1, u_2 \right), \left( w_1, w_2 \right) \in \left\lfloor V \right\rfloor_\rho \) such that \( x_1 u_1 w_1 x_2 u_2 w_2 \in WK_\rho(V) \) and q, q' \in Q, \( x \rho y, x \rho y \) if there is transition rule \( q \left( x_1, x_2 \right) \rightarrow q' \) in \( \delta \) and \( \Rightarrow \) denotes the transitive and reflexive closure of \( \rightarrow \). The language accepted by a Watson-Crick automaton M in the upper strand is L(M)={w_1 \in V^* |q_0(w_1, w_2) \Rightarrow [w_1, w_2] q, with q \in F, w_2 \in V^* \} \). Excluding the beginning and \( a\sim_p a \) or \( a\not\sim_p a \).

Definition 10: Deterministic Watson-Crick automaton is a Watson-Crick automaton for which if there are two transitions of the form \( q(w_1, w_2) \rightarrow q' \) and \( q'(w_1', w_2') \rightarrow q'' \) then \( u\sim_p u' \) or \( u\not\sim_p u' \).

Definition 11: A Watson-Crick quantum finite automaton(1WKQFA)[20] is a nine tuple M=(Q,V,\delta,q_0,Q_{acc},Q_{rej},\#,$) where Q is a finite set of states, V is the input alphabet, \( \delta \) is the transition function, \( q_0 \in Q \) is the initial state, \( Q_{acc} \subseteq Q \) and \( Q_{rej} \subseteq Q \) are sets of accepting and rejecting states. The complementarity relation \rho is similar to
Watson-Crick complementarity relation. The states in \( Q_{\text{acc}} \) and \( Q_{\text{rej}} \) are called halting states and the states in \( Q_{\text{non}} = (Q_{\text{acc}} \cup Q_{\text{rej}}) \) are called the non-halting states. The symbols '#' and 'S' do not belong to \( V \). We use '#' and 'S' as the left and right end-markers respectively. The working alphabet of \( M \) is \( \Gamma = V \cup \{\#, S\} \).

3 Two-way non-deterministic Watson-Crick automata

Two-way non-deterministic Watson-Crick automaton[21] is a 8 tuple, \( M = (V, \#, S, \rho, Q, q_0, F, \delta) \) where \( V \) is alphabet, \( \#, S \notin V \) are the beginning and the end marker respectively. Set of states is denoted by \( Q \), \( \rho \) is the symmetric complementarity relation same as that of Watson-Crick automata, \( q_0 \) is the initial state and \( F \subseteq Q \) is the set of final states. \( \delta \) contains finite number of transition rules of the form \( \delta(q, w_1, w_2, \text{dir}_1, \text{dir}_2) \rightarrow q' \), which denotes that the machine in state \( q \) parses \( w_1 \) in upper strand in \( \text{dir}_1 \) direction and \( w_2 \) in lower strand in \( \text{dir}_2 \) direction and goes to state \( q' \) where \( w_1, w_2 \in V \{\#, S\}^* \) and \( \text{dir}_1, \text{dir}_2 \in \{L, R, 0\} \) where \( L \) signifies that the head is reading the word in the left direction, \( R \) signifies that the head is reading the word in right direction and if a head reads the empty word \( \lambda \) it remains in its current position denoted by \( 0 \) with the restriction that if \( w_1 \) or \( w_2 = V^* \# \) then the corresponding \( \text{dir}_1 \) or \( \text{dir}_2 \neq L \). The above restriction ensures that the reading heads do not go past the input word on the left side.

Accepting conditions

\( W_1 \) is accepted by \( M \), if starting in state \( q_0 \) (initial state) with \( \begin{bmatrix} \#w_1 \$ \\ \#w_2 \$ \end{bmatrix} \) on the double stranded input tape and the two heads at the left end of \( \#w_1 \$ \) and \( \#w_2 \$ \). \( M \) eventually enters a final state and both the heads fall off the right hand side of the double stranded input tape.

The word \( w_1 \) is rejected if one of the following 3 conditions occurs:

i. The two-way Watson-Crick automaton enters a loop which is indentified in a similar way as loops in two-way finite automaton.

ii. When both the heads fall off the right hand side of the input tape and the machine is in a non-final state.

iii. If the machine comes to a halt (i.e. there are no transition rules that can be applied for that particular state and inputs the heads are reading) before the heads fall off the right hand side of the input tape.

Definition 12: Depending on the type of states and transition rules there are four subclasses of two-way Watson-Crick automata similar to Watson-Crick automata.

A two-way Watson-Crick automaton \( M = (V, \#, S, \rho, Q, q_0, F, \delta) \) is

1) stateless(2NWK): If it has only one state, i.e. \( Q = F = \{q_0\} \);

2) all-final(2FWK): If all the states are final, i.e. \( Q = F \);

3) simple(2WK): If at each step the automaton reads either from the upper strand or from the lower strand, i.e. for any transition rule \( q \left( \begin{bmatrix} w_1, \text{dir}_1 \\ w_2, \text{dir}_2 \end{bmatrix} \right) \rightarrow q' \), either \( w_1 = \lambda \) or \( w_2 = \lambda \);
4) $I$-limited(21WK): If for any transition rule $q_{(w_1, w_2, d, r)} \rightarrow q'$, we have $|w_1w_2|=1$.

4 Equivalence of Subclasses of two-way Watson-Crick automata

In this Section, we show the equivalence of different subclasses of two-way Watson-Crick automata.

Theorem 1: All final two-way Watson-Crick automata have the same computational power as two-way Watson-Crick automata.

Proof: Let $M=(V, \#, S, \rho, Q, q_0, F, \delta)$ be a two-way non-deterministic Watson-Crick automaton. We introduce an all final two-way Watson-Crick automaton $M'=(V, \#, S, \rho, Q', q_0, \delta')$.

Each transition rule $t$ of the form $q_{(w_1, w_2, d, r)} \rightarrow q'$ in $\delta$ where $w_1=a_1a_2...a_n$ where $|w_1|=n$ and $w_2=b_1b_2...b_m$ where $|w_2|=m$ and $a_n, b_n \in V \cup \{\#, \}$ falls under one of the five classes. The classes are defined as follows:

Class 1: Transition rules of the form $q_{(w_1, w_2, d, r)} \rightarrow q'$ in $\delta$ where $w_1=a_1a_2...a_n$ where $|w_1|=n$ and $w_2=b_1b_2...b_m$ where $|w_2|=m$ and $a_n$ and $b_n \neq \#$, i.e. $w_1$ and $w_2$ do not have $\#$ at their ends.

Class 2: Transition rules of the form $q_{(w_1, w_2, d, r)} \rightarrow q'$ in $\delta$ where $w_1=a_1a_2...a_n$ where $|w_1|=n$ and $w_2=b_1b_2...b_m$ where $|w_2|=m$ and $a_n$ and $b_n = \#$, i.e. $w_1$ and $w_2$ both have $\#$ at their ends.

Class 3: Transition rules of the form $q_{(w_1, w_2, d, r)} \rightarrow q'$ in $\delta$ where $w_1=a_1a_2...a_n$ where $|w_1|=n$ and $w_2=b_1b_2...b_m$ where $|w_2|=m$, $a_n = \#$ and $b_n \neq \#$, i.e. $w_1$ has $\#$ at its end and $w_2$ does not have $\#$ at its end.

Class 4: Transition rules of the form $q_{(w_1, w_2, d, r)} \rightarrow q'$ in $\delta$ where $w_1=a_1a_2...a_n$ where $|w_1|=n$ and $w_2=b_1b_2...b_m$ where $|w_2|=m$, $a_n \neq \#$ and $b_n = \#$, i.e. $w_1$ does not have $\#$ at its end and $w_2$ has $\#$ at its end.

Class 5: Either transition rules of the form $q_{(s, L)} \rightarrow q'$ in $\delta$ or transition rules of the form $q_{(s, L)} \rightarrow q'$ in $\delta$ or transition rules of the form $q_{(s, L)} \rightarrow q'$ in $\delta$.

The transition rules of $M$ are modified as follows to form the transition rules of $M'$. 

Transition rules of M which fall in class 1 and class 5 are kept same in M'.

For transition rules of M which belong to class 2 two instances can occur;

case 1: For transition $q \xrightarrow{w_1, R} q'$, where $q'$ is a final state. In this case the transition rules are kept same in M'.

case 2: For transition $q \xrightarrow{w_1, R} q'$, where $q'$ is a non-final state. In this case the transition rules of M are modified as follows for M'.

For each transition rule $q \xrightarrow{w_1, R} q'$ in M belonging to class 2 where $q'$ is a non-final state, $q \xrightarrow{w_1, R} q''$ where $w_1 = w_1' S$ and $w_2 = w_2' S$ are introduced in M' and there is no transition from $q''$ in M'. These new rules in M' ensure that if the heads go off the right end of the tape in M when M is in a non-final state then M' would go to state $q''$ and would not accept the string as there is no transition from $q''$ and so the heads of M' will not fall off the right end of the tape. The above stated rules ensure the heads do not fall off the right end of the tape for M' when M does not accept the word. As M' is all final if the heads go off the right end of the tape it will accept the given string.

For transition rules of M which belong to class 3 the following modifications are needed. Class 3 also has two instances similar to class 2.

case 1: For transition $q \xrightarrow{w_1, R} q'$, where $q'$ is a final state. In this case the transition rules are kept same in M'.

case 2: For transition $q \xrightarrow{w_1, R} q'$, where $q'$ is a non-final state. In this case the transition rules of M are modified as follows for M'.

For each transition rule $q \xrightarrow{w_1, R} q'$ in M belonging to class 3 where $q'$ is a non-final state,

$q \xrightarrow{w_1, R} q'_{u5}$ where $w_i = w_i' S$ is introduced in M' where $q'_{u5}$ denotes that the head on the upper strand has gone past the right end marker $S$ in the original machine M on application of the above transition rule.

Only rules having $\lambda$ on the upper strand are applied to $q'_{u5}$ because in the actual machine M if the above rules of class 3 are applied then the upper head would have gone past the right end of the tape. So only rules having $\lambda$ on the upper head can be applied to the original machine M at this stage. As M' replicates M similar thing is done in M' too.
Thus, all the transition rules that can be applied to \( q' \) in \( M \) with \( \lambda \) on the upper strand and \( w_2=a_1a_2...a_n \) and \( a_n \neq \$ \) in the lower strand can also be applied to \( q'_{qS} \) in \( M' \). Rules having \( \lambda \) on the upper strand and \( w_2=a_1a_2...a_n \) and \( a_n=\$ \) in the lower strand where the transition goes to a final state are applied to \( q' \) in \( M' \). Finally for rules with \( \lambda \) on the upper strand and \( w_2=a_1a_2...a_n \) and \( a_n=\$ \) in the lower strand where the transition goes to a non-final state, the rules of the form \( q'_{wL,R} \vdash a_0 \rightarrow q_{aS} \) where \( w_2=\$ \) are introduced in \( M' \) and there are no transition rules from \( q_{aS} \) these rules ensure that when \( M \) reaches the end of the string on a non-final state then \( M' \) goes to \( q_{aS} \) and \( M' \) does not accept the string as there is no transition from \( q_{aS} \). The above stated rules ensure the heads do not fall off the right end of the tape for \( M' \) when heads off \( M \) fall off the right end and the state to which \( M \) goes is non-final.

Class 4 rules are handled in a similar way to class 3 rules.

All the new states are introduced in \( Q' \) along with states in \( Q \).

It is evident from the transition rules introduced in \( M' \) that \( M' \) accepts the same language as \( M \).

Thus, for every two-way Watson-Crick automaton we obtain an all final two-way Watson-Crick automaton which accepts the same language and every all final two-way Watson-Crick automaton is a two-way Watson-Crick automaton. Therefore, two-way Watson-Crick automaton and all-final two-way Watson-Crick automaton have the same computational power.

Example 1: There is a deterministic two-way Watson-Crick finite automaton that accepts the context sensitive language \( L=\{ww|w\in \{a, b\}^+\} \).

Proof: \( M=(V,\#,S,\rho,Q,q_0,F,\delta) \) is a deterministic two-way Watson-Crick automaton with non-injective complementarity relation \( \rho \) that accepts the context sensitive language \( L=\{ww|w\in \{a, b\}^+\} \) where \( Q=\{q_0, q_1, q_2, q_3, q_4, q_5\} \) and \( F=\{q_5\} \).

We define the transitions involved in \( M \) as follows:

\[
\begin{align*}
\delta(q_0, (\text{a}), 1, 1) &= q_0, \\
\delta(q_0, (\text{a}), 1, 0, -1) &= q_1, \\
\delta(q_1, (\text{a}), 0, -1) &= q_1, \\
\delta(q_3, (\text{a}), 0, 1) &= q_3, \\
\delta(q_3, (\text{a}), 0, -1) &= q_4, \\
\delta(q_4, (\text{a}), 0, -1) &= q_5.
\end{align*}
\]

The above automata works in the following manner after reading the first end marker \#. The upper head reads a symbol and the lower head reads two symbols. Now if the length of the input word \( |w| \) is odd then the lower head will never reach \$ because it is reading two symbols at a time. Thus it will terminate in a non final state. Thus all odd strings are eliminated. If the input string is even, then the upper head stops at the first middle element (as it is even length string, there are two middle element). Now the
lower head is moved to the beginning of the string and the first element is matched with the \((1^{st}} \text{ middle element + 1)}^{th}\) element and this matching continues until $ is reached on upper head. If there is a mismatch somewhere then no transition is defined thus automaton rejects the input. The upper head will reach $ only if the two halves match i.e. the input is of the form ww. Then the lower head also moves to $ to accept the input. Then the automation only accepts string of the form ww.

**Theorem 2:** $L_{2NWK} \neq L_{2QFA}$, where $L_{2NWK}$ is the set of all languages accepted by two-way non-deterministic Watson-Crick automata and $L_{2QFA}$ is the set of all languages accepted by two-way quantum automata.

**Proof:** From Example 1, we know that there is a two-way Watson-Crick automaton that accepts the context sensitive language $L=\{ww|w \in \{a,b\}^+\}$ and from paper[22] we know that the context sensitive language $L=\{ww|w \in \{a,b\}^+\}$ is not accepted by two-way quantum automaton which proves the above Theorem.

**Theorem 3:** $L_{1WKQFA} \neq L_{2QFA}$, where $L_{1WKQFA}$ is the set of all languages accepted by Watson-Crick quantum finite automata and $L_{2QFA}$ is the set of all languages accepted by two-way quantum automata.

**Proof:** From [20], we know that there is a one-way Watson-Crick quantum finite automaton that accepts the context sensitive language $L=\{ww|w \in \{a,b\}^+\}$ and from paper[22] we know that the context sensitive language $L=\{ww|w \in \{a,b\}^+\}$ is not accepted by two-way quantum automaton which proves the above Theorem.

5 Conclusion

We investigated the computational power of different subclasses of two-way non-deterministic Watson-Crick automata. We show that all final two-way Watson-Crick automata and two-way Watson-Crick automata have the same computational power. In the last section we showed that Watson Crick automata and 1-way Watson crick quantum finite state automata accepts a language $L=\{ww|w \in \{a,b\}^+\}$ which is not accepted by two-way quantum finite state automata.

References

1. Moore, G., Lithography and the Future of Moore's Law, Proceedings of SPIE, Vol. 2437, May 1995.
2. Moore C and Crutchfield J., Quantum automata and quantum grammars, Theoretical Computer Science, 237(2)(2000); Los Alamos Preprint Archive, 1997, page 275-306.
3. Kondacs A and Watrous J., On the power of quantum finite state automata, in: Proc. 38th Annu. Symp. on Foundations of Computer Science; Miami, Florida, 1997, page 66-75.
4. Ambainis A, Freivalds R, One-way quantum finite automata: strengths, weakness and generalizations, IEEE 39th Annual Symposium on Foundations of Computer Science, 1998, page 332-342.
5. Ambainis, A., Bonner, R., Freivalds, R. and Kikusts, A. Probabilities to accept languages by Quantum finite automata, COCOON, 1999, page 174-183.
6. Ambainis, A., Bcandry, M., Golovkins, M., Kikusts, A., Mercer, M., and Therien, D. Algebraic results on Quantum automata, STACS, 2004, page 93-104.
7. Bertoni, A., Mereghetti, C, and Palano, B. Quantum Computing: 1-way quantum automata, Developments language theory, 2003, page 1-20.
8. Ambainis, A., Watrous, J., Two way finite automata with quantum and classical states, Theoretical Computer Science 287(1)(2002), 2002, page 299-311.
9. Ambainis, A., Beaudry, M., Golovkins, M., Kikusts, A., Mercer, M., Therien, D. Algebraic results on quantum automata. In: Diekert, V., Habib, M. (eds.) STACS, LNCS, vol. 2996, Springer, Heidelberg, 2004, page 93-104.
10. Dzelme, I. Kvantu Automati Jauktajiem Stavokliem, Technical Report, University of Latvia, 2003.
11. Nayak, A., Optimal lower bounds for quantum automata and random access codes. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, 1999, page 369-377.
12. Hirvensalo, M., Various aspects of finite quantum automata, International Conference on Developments in Language Theory: Developments in Language Theory, pp 21-33, 2008.
13. D. Ganguly, K. Chatterjee, K. S. Ray, 1-Way Multihead Quantum Finite State Automata, applied mathematics 4 (10A), 2016.
14. Freund R., G. Päun, G. Rozenberg, A. Salomaa, Watson-Crick Finite Automata, Proc 3rd DIMACS Workshop on DNA Based Computers, Philadelphia, 297-328, 1997.
15. G. Paun, G. Rozenberg, A. Salomaa,” DNA Computing: New Computing Paradigms,” Springer-Verlag, Berlin, 1998.
16. E. Czeizler, E. Czeizler, L. Kari, K. Salomaa, Watson-Crick automata: determinism and state complexity, In proceeding of: 10th International Workshop on Descriptional Complexity of Formal Systems, DCFS 2008, July 16-18, pp.121-133, 2008.
17. E. Czeizler, E. Czeizler, L. Kari, K. Salomaa, On the descriptional complexity of Watson-Crick automata, Theoretical Computer Science, Volume 410, Issue 35, Pages 3250–3260, 28 August 2009.
18. K. S. Ray, K. Chatterjee, D. Ganguly, State complexity of deterministic Watson–Crick automata and time varying Watson–Crick automata, Natural Computing, February 2015.
19. Paun G., G. Rozenberg, A. Salomaa, DNA Computing. New Computing Paradigms, Springer, Berlin, 1998.
20. D. Ganguly, K. Chatterjee, K. S. Ray, Watson-Crick Quantum Finite Automata, Acta Informatica, Springer, 2020, https://doi.org/10.1007/s00236-020-00370-x.
21. K. S. Ray, K. Chatterjee, D. Ganguly, Equivalence of Subclasses of Two-Way Non-Deterministic Watson-Crick Automata, Applied Mathematics, Vol.4, No.10A, October 2013.
22. A. S. Bhatia, A. Kumar, On the power of two-way multihead quantum finite automata, RAIRO-Theor. Inf. Appl. 53, 19–35, January, 2019. https://doi.org/10.1051/ita/2018020