Volume 2, Issue 9, September 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Quantum Finite Automata: a Language Acceptor Model

Tirtharaj Dash*
Department of Computer Science and Engineering
Veer Surendra Sai University of Technology
Burla-768018, India

Tanistha Nayak
Department of Information Technology
National Institute of Science and Technology
Berhampur-761008, India

Abstract— There exist an computational gap between the models that allowed to use quantum effects and that are not using quantum effects. Quantum Finite Automata (QFA), Quantum Pushdown Automata (QPDA), Quantum branching are several computational models. Some computational models are more powerful than classical counterparts and some are not. In this survey work, we have drawn a comparison among 1-way QFA, 2-way QFA and 1.5-way QFA. Also we have discussed about the strengths and weakness of all these QFA models and the power of Quantum Finite State Automata.

Keywords— Quantum Finite Automata (QFA);Quantum Pushdown Automata(QPDA); 1-way Quantum Finite Automata (1QFA) ;2-way Quantum Finite Automata (2QFA);1.5-way Quantum Finite Automata (1.5QFA)

I. Introduction

A language is set of strings of symbols. In this sense we can define a programming language as a set of symbols only. In the environment of Computer Science and Engineering, there are four classes of languages called Chomsky Formal languages. The four classes can be given as (i) Regular languages (Type 3) (ii) Context-free language (Type 2) (iii) Context Sensitive language (Type 1) (iv)Phrase structure (Type 0). These formal languages are characterized by grammars, which are essentially a set of rewrite rules for generating strings belonging to a language [1, 2]. Also there is various kind of computing devices called automata which process these types of languages and generate some output if required. Thus formal languages can also be characterized by the computing devices which process them. These formal languages and automata capture the essence of various computing devices which process them. A finite state automaton involves states and transitions among states in response to inputs. They are useful for building several different kinds of software, including the lexical analysis component of a compiler and systems for verifying the correctness of circuits or protocols. They also serve the control unit in many physical systems including vending machines, elevators, automatic traffic signals and computer microprocessors. The Regular languages are the simplest of the four formal languages, together with regular expressions which are the methods of representing regular languages in a mathematical way [3, 4, 18].

It is quite possible that the first implementations of quantum computers will not be fully quantum mechanical. Instead, they may have two parts viz. a Quantum part and a Classical part with a communication between these two parts. In this case, the quantum part will be considerably more expensive than the classical part. Therefore, it will be useful to make the quantum part as small as possible even if it leads to slight (reasonable) increases in the size of the classical part [4]. This motivates the study of systems with a small quantum mechanical part.

Quantum finite automata can be classified into three types. These are listed below.

(i) 1-way quantum finite automata
(ii) 1.5 way quantum finite automata
(iii) 2-way quantum finite automata.

1-way and 2-way Quantum Finite Automata are quantum analogous to deterministic, nondeterministic and probabilistic finite automata. The first ever developed quantum automatons are 1-way quantum finite automaton and 2-way quantum finite automaton. These automata are discussed below as different sections followed by a conclusion section [5].

II. 1-way Quantum Finite Automata (1QFA)

One-way Quantum finite automata (1-QFA) proposed by Moore and Crutchfield. 1-way Quantum Finite Automata is discussed first. One way quantum finite automaton is a very reasonable model of computation. The finite dimensional state-space of a QFA corresponds to a system with finitely many particles. A classical device can read symbols from the input and apply the corresponding transformation to quantum mechanical part. A quantum automaton (real time) consists of following elements: A Hilbert space H, An initial state vector $S_{init} \in H$ with $|S_{init}|^2 = 1$, A subspace $H_{accept} \subset H$ and an operator P accept that projects on to it, An input alphabet A, A unitary transition matrix U_a for each symbol $a \in A$. 1-way quantum automaton means the head moves only one direction and acceptance power of this is reduced to proper subset of regular language. There are two different models of one-way quantum finite automata [5, 6]. They are (i) measure-once quantum finite automata (ii) measure many quantum finite automata. Measure once model is defined when it is restricted to accept with bounded error. 1-way QFA performs measurement on its configuration. The major difference between measure-once model and measure many model is that, measure-once model computes at the end of computation whereas measure many model measures computes at the end of each transition. So, the acceptance capability of measure-many automaton is more powerful than measure-once automaton. The languages accepted by measure-once automaton is closed under the inverse homeomorphisms and Boolean operations. A measure-once quantum finite automaton is defined by a 5-tuples. It can be described as $M=(Q,\sum,\delta,q_0,q_{acc},F)$ where Q is a finite set of states, $\sum$ is an input alphabet, δ is transition function i.e. $\delta:Q\times\Sigma\times Q\rightarrow\check{C}$, q_0 is starting state, $q_{acc} \in Q$ is accepting state, F is accepting states. A measure many quantum finite automaton consists of six (6) tuples. $M= (Q,\sum,\delta,q_0,\sum,\delta,q_0,q_{acc}, q_{rej},)$ where Q is a finite set of state, $\sum$ is a finite input alphabet with an end marker symbol \$, δ is an unitary transition function with $\delta:Q\times\Sigma\times Q\rightarrow\check{C}$, q_0 is the initial configuration of M. The set Q is divided in three sets i.e. Q_{acc} is set of accepting states, Q_{rej} is set of rejecting states, Q_{non} is set of non halting states. Measure- many quantum finite automaton measures every transition i.e. it measures every transition with respect to three subsets that correspond to three subsets i.e. (i) Q_{acc} (ii) Q_{rej} (iii) Q_{non}: E_{acc} = span({|q> |q Є q_{acc}}), E_{rej} = span({|q> |q Є q_{rej}}), E_{non} = span({|q> |q Є q_{non}}. If the computation M is in the Q_{non} state, then the computation continues, if the computation is in E_{acc} state then M accepts, else M rejects. After every measurement the superposition collapses into measured subspaces and renormalized. In one-way quantum finite automaton we only require one end marker at the end of the tape. Measure many model is strictly more powerful than the measure-once model but it is more difficult to characterize. The two classes of language, those accepted with or without bounded error are closed under inverse homomorphism and complementation. A one-way quantum automaton is a reasonable model of computation. The languages accepted by one-way quantum finite automaton is a proper subset of regular languages. 1-way QFA is more powerful than 1-way reversible finite automata. Obviously, 1-way QFA is more powerful than 1-way reversible automata. If a QFA gives a correct answer with larger probability (greater than 7/9), then it can be replaced by a 1-way reversible automaton [5, 7]. One-way QFA is more space efficient than other automaton. For example, if there is a 1-QFA that can check whether the number of input received from the input is divisible by a prime P with only O (log p) states. But any other probabilistic automaton requires P states. This explains that 1-way QFA is space efficient.

A 1-way reversible finite automaton (RFA) is a quantum finite automaton (QFA) with $\delta(q_1, a,q_2) \in \{0,1\}$ for all q_1,a,q_2. RFA can also be defined as a deterministic automata where, for q_2, a there is at most one state q_1 such that reading a symbol a leads to state q_1 to q_2. To introduce probabilism into finite automata without losing reversibility is quite hard. There are some probabilistic choices that are consistent with reversibility. The language $L = \{a^{2n+3}| a \in N\}$ cannot be recognised by 1-way reversible finite automata (RFA). However, this can be recognized by 1-way quantum finite automata. Therefore we define 1-way quantum finite automata with probabilistic choices (PRFAs) [8,9,10]. A PRFA is probabilistic finite automata such that for any state q_1 and $a \in \acute{\Gamma}$, there is at most one state q_2 such that the probability of passing from q_2 to q_1 after reading 'a' is nonzero.

Drawback of One-Way Quantum Finite Automata

If L is a language recognized by 1 way quantum finite automata (1-QFA) with N states, then it can be recognized by 1-way deterministic automata (1-DFA) with $2^{O(N)}$ states. Therefore transformations of 1-QFA into classical counterpart causes exponentially increase in size. Space efficient of 1-QFA is less powerful than classical counterpart. However, Deterministic Finite Automata (DFA) is more powerful than 1-QFA. For removing the drawback of 1QFA, 2QFA is developed.

III. 2-WAY QUANTUM FINITE AUTOMATA(2-QFA)

The 2-way Quantum Finite Automata (2-QFA) consists of finite state control and a 2 way tape head which scans read only tape. The 2-QFA can be defined as $M = (Q,\sum,\delta,Q_0, Q_{acc}, Q_{rej})$ where : Q is Finite set of state, $\sum$ is input alphabet, δ is transition state function, Q_0 is initial state, $Q_{acc} \subset Q$ is set of accepting state, $Q_{rej} \subset Q$ is set of rejecting state. The tape alphabet $\acute{\Gamma} = \Sigma \cup \{\#,\$\}$ where left end and right end marker respectively. The transition function can be defined as $\delta: Q\times T\times Q\times\{-1,0,1\}\rightarrow C$ if q and $q' \in Q$, $б \in T$, $d \in \{-1,0,1\}$.
Then δ (q, б, q', d) represents the amplitude with which a machine in state q moves to state q' by scanning the symbol б and with move direction d. Two way Quantum Finite Automata (2QFA) can stimulate any nondeterministic automaton and also can recognize some non regular languages.

This also suffers from some disadvantages, i.e. it allows superposition where the head can be multiple positions simultaneously. For implementing this we need at least *O(logn)* qubits to store the position of head, where n represents the length of input tape. Quantum computer may not be fully quantum mechanical. There may be classical part and quantum part where quantum part may be expensive than classical part. Therefore we propose 2-way finite automata with classical and counterpart which acts as an intermediate between 1QFA and 2QFA. Let us consider the following languages: $L_1 =\{x \in (a,b)^*|x=x^R\}$ and $L_2 =\{a^n b^n|n \in N\}$. These languages can be solved by 2-way quantum finite automaton with classical and counterpart. While solving these two languages, we find that the quantum part of a machine consists of only single qubit [11,12]. That qubit represents and process certain information regarding input. A two-way classical quantum finite automaton (2QCFA) has to access fixed size of quantum resister by which it performs quantum transformation and measurement. The transformations and measurements are determined by local descriptions of the classical portion of the machine, and the results of the measurements may determine the manner in which the classical part of the machine evolves.

A two-way finite automaton with quantum and classical States (2QCFA) has 9-tuples. M =(Q,S,Σ,θ,δ,Q_0, S_0, S_{acc}, S_{rej}) where Q,S are finite set of states quantum and classical states respectively. Σ is finite alphabet, θ. δ are the transition functions that defined M. Q_0 is initial state, S_0 is initial classical state. S_{acc}, S_{rej} are accepting and rejecting state. Γ́ is tape alphabet of M where $\acute{\Gamma} = \Sigma \subset \{¢, \$\}$. $¢ \notin \Sigma$ is known as left end marker and $\$ \notin \Sigma$ is right end marker. The function θ defines quantum portion of internal state. The language L_2 is non-regular language, but it is recognized by 2-way probabilistic finite automaton [13,14,15]. 2-way Quantum Finite Automaton (2QFA) uses superposition where the head of QFA is in different places and different components of the superposition.

Let us consider another language which will describe how the language L ={$a^m b^n$|m,n>0}. The transition diagram is given in figure 1.

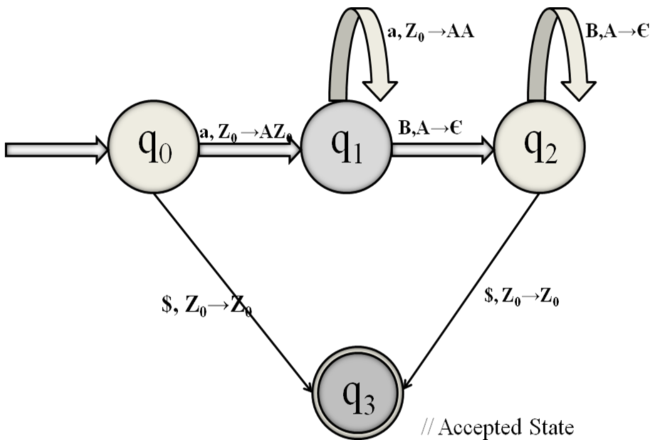

Fig 1. Transition diagram of L ={$a^m b^n$|m,n>0}

Algorithm:

Step 1: First check whether the input string is in the form of $a^m b^n$, If not then *Reject*.

Step 2: Else repeat the following the following step.

Step 3: Move the tape head on symbol ¢ and set the quantum state to |q_0>. While the currently scanned symbol is not $, do the following.

3. 1. If the currently scanned symbol is ‘a’, then perform “add that symbol into stack.”

3.2. If the currently scanned symbol is ‘b’, then delete that symbol from that stack.

Step 4: Then Measure the quantum state (i.e. on symbol $). If the result is $, then accept, otherwise *Reject*.

Step 5: Finish

Another language that could be considered is a Palindrome string. Palindrome can be of two types i.e (i) odd palindrome (ii) even palindrome. Odd and Even palindrome diagram with their transition diagrams are given in figure 2 and 3 respectively.

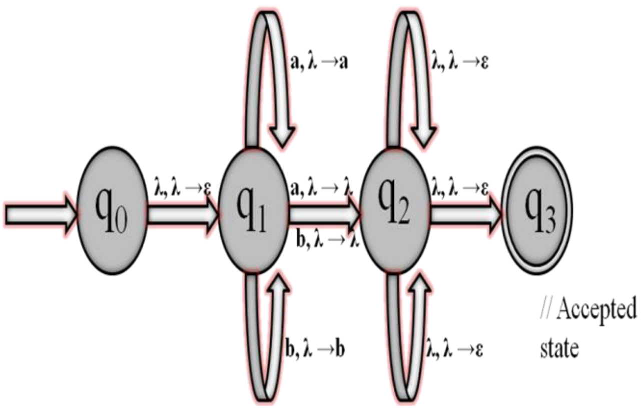

Fig 2. Transition diagram of an odd palindrome

The following describing the even palindrome ={SS^R}. There is no middle character in an even length palindrome.

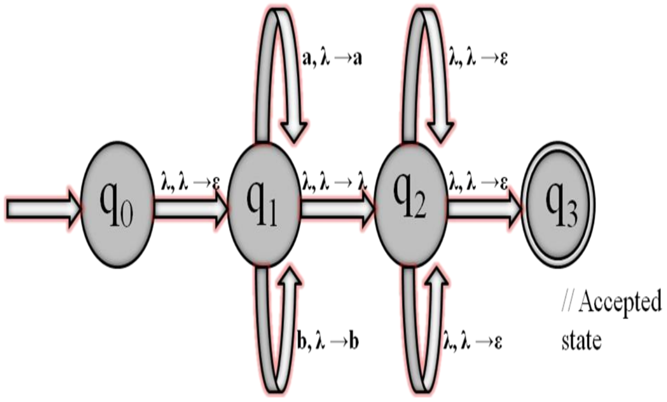

Fig 3. Transition diagram of even length palindrome

To overcome the problems of 1-QFA and 2-QFA and to make it more powerful A.Kondacs and J.Watrous proposed some ways which can be followed for the development of the QFAs. [19].

They are classified into five types, (i) 1-way QFA with classical automata (ii) 1-way QFA with 1-counter (iii) 1-way QFA with multiple counter (iv) 1-QFA scanning the input strings with multiple times (v) 2-QFA with one counter (vi) 2-QFA with classical automata.

1-way with Multiple Counters:
1-way k-counter means this automaton has k counter and each counter initialized into Zero [15,16]. 1-way k-counter can be defined as M = $(Q,\sum,\delta,Q_0, Q_{acc}, Q_{rej})$. The transition function cab be defined as follows.

$$\delta: Q\times\acute{\Gamma}\times S\times S\times Q\times\{-1,0,1\}\times\{-1,0,1\}\rightarrow C \text{ with } \acute{\Gamma} = \Sigma \bigcup \{\#,\$\} \text{ and } S=(0,1).$$

If we consider K=3,for fixed input, the configuration of the 1-way 3-counter automaton may be defined as (i) the internal state of the automaton (ii) the value of first counter (iii) the value of the second counter (iii) the value of the third counter.

Let us take one automaton $L_3= \{a^m b^n c^{mn} \mid m, n \text{ are real number}\}$. Here we are using three counters i.e. m, n, z. The steps of the automaton are given below.

Steps:

1. Initialize the value of three counters to ‘0’. That means m = n = z = 0.
2. By reading “a” increment the value of counter “m” and counter “z”. The value of counter n remains constant.
3. By reading “b” increment the value of “n” and the value of “m” and “z” remain constant.
4. By reading “C” decrement the value of “n”.
5. When the value of ‘n’=0 and the value of ‘m’ ≠ 0, then decrement the value of ‘m’ and assign ’n’=’z’.
6. Go to step 4.
7. Repeat the process until the reading head reads right end marker ‘$’.
8. When ‘$’ is read if the value of the counter ‘n’=0 and the value of counter ‘m’=0, then accept otherwise reject.

Let us take an input string “aabbbcccccc”. The input string is “#aabbbcccccc$”. The steps are given below.

Steps:

1. In the first step “#” is read. So the counter are initialized into m=1, n=0 and Z=1.
2. In the second step ‘a’ is read. So the counter values are m=2, n=0, Z=2.
3. In the third step ‘a’ is read. Counter values will be m=2, n=0, Z=2.
4. In the forth step ‘b’ is read. Counter values will be m=2, n=1, Z=2.
5. In the fifth step ‘b’ is read. Counter values are m=2, n=2, Z=2.
6. In the sixth step ‘b’ is read. Counter values are m=2, n=3, Z=2.
7. In seventh step ‘c’ is read, then counter values are m=1, n=2, z=2.
8. In eighth step ‘c’ is read, then counter values are m=1, n=1, z=2.
9. In ninth step ‘c’ is read, then counter values are m=1, n=0, z=2.
10. Again ‘c’ is read, the counter values will be m=1, n=-1, z=2.
11. Again ‘c’ is read, the counter values will be m=0, n=1, z=2.
12. Now ‘c’ is read, the counter values will be m=0, n=0, Z=2.

When all counters will be Zero, then the input string will be "Accepted" or otherwise Rejected.

IV. 1.5-WAY QUANTUM FINITE AUTOMATA (1.5QFA)

1.5 way Quantum Finite Automata was defined by Amano and Iwama. It is a special case of 2-way Quantum Finite Automata (2QFA) which cannot move its head to the left.1.5-way Quantum Finite Automata (1.5QFA) can recognized some non-context free languages [16,17]. 1.5 way QFA can be defined as M= $(Q,\sum,\delta,Q_0, Q_{acc}, Q_{rej})$, where Q is set of states, $\sum$ is set of input symbols including left and right end marker, δ is transition function, $Q_0 \subset Q$, Q_{acc} is set of accepting state, Q_{rej} is set of rejecting state where $Q_{acc} \bigcup Q_{rej} = \Phi$. The transition function can be defined as $\delta: Q\times\sum\times Q\times\{0,1\}\rightarrow \acute{C}$. Informally 1.5 can be defined as computational model of Quantum Pushdown Automata with classical counterpart (QCPA) without a stack i.e.1.5 QFA is a quantum part of QCPA.The non-context-free language L $=\{a^m\ d\ b^n\ d\ c^n\ |n>0\}$ can be recognized by 1.5QFA with probability less than ⅔. It can also recognize L = $\{ a^n\ d\ b^m\ d\ c^n\ |n>0\}$ and L = $\{ a^n\ d\ b^n\ d\ c^m\ |n>0\}$.

We are comparing all the powers of all the languages discussed above [5, 19] and is given in figure 4 below.

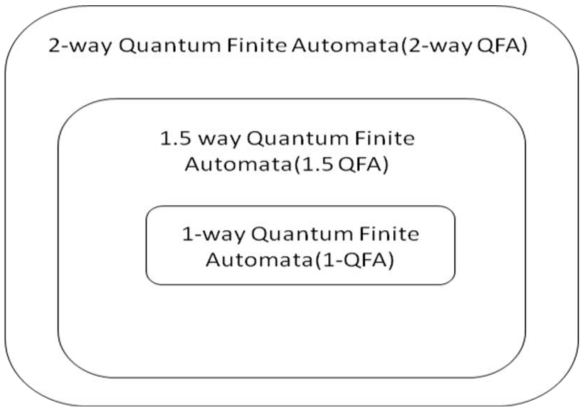
2-way Quantum Finite Automata(2-way QFA)
1.5 way Quantum Finite Automata(1.5 QFA)
1-way Quantum Finite Automata(1-QFA)

Fig 4: Comparsion among 1-way, 1.5-way and 2-way QFA

V. CONCLUSIONS

In this paper, we discussed about the Quantum Finite Automata (QFA). Here we focused the developing algorithms by one-way quantum k-counter automata. This automaton is increasing the power of 1QFA and 2QFA. Here we focused several methods for constructing the methods of 2QFA's for recognizing the non-regular languages. Some more method can also be possible for developing the power of both 1QFA and 2QFA such as 2QFA with classical finite K-counter automata and 2QFA with K-counter. And it will be a best idea to test the steps and ideas with more number of language test cases to check the efficiency of the automaton. The authors are currently working on this.

REFERENCES

[1] C. Moore, J.P. Crutchfield, Quantum automata and quantum grammars, Theoret. Comput. Sci. 237 (1–2) (2000) 275–306.
[2] A. Kondacs, J.H.Watrous, On the power of quantum finite state automata, in: 38th Annual Symp. Foundations of Computer Science, 1997, pp. 66–75б.
[3] T. Nayak, T. Dash, A Comparative Study on Quantum Pushdown Automata, Turing Machine and Quantum Turing Machine, International Journal of Computer Science and Information Technologies, Vol.-3(1), 2012, pp. 2932-2935.
[4] T. Nayak, T. Dash, Quantum Finite Automata, Quantum Pushdown Automata & Quantum Turing Machine: A Study, International Journal of Computer Science and Information Technologies, Vol.-3(2), 2012, pp. 3765-3769.
[5] T. Dash, T. Nayak, Comparative Analysis on Turing Machine and Quantum Turing Machine, Journal of Global Research in Computer Science, Vol.-3(5), pp. 51-56.
[6] D. Aharonov, A. Kitaev, N. Nisan, Quantum circuits with mixed states, in proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, pages 20–30, 1998.
[7] A. Ambainis and R. Freivalds, 1-way quantum finite automata: strengths, weaknesses and generalizations, in proceedings of the 39th Annual Symposium on Foundations of Computer Science, pages 332–341, 1998.
[8] A. Ambainis, A. Nayak, A. Ta-Shma, U. Vazirani, Dense quantum coding and a lower bound for 1-way quantum automata, in proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, pages 376–383, 1999.
[9] A. Brodsky, N. Pippenger, Characterizations of 1-way quantum finite automata, Los Alamos Preprint Archive, quant-ph/9903014, 1999.
[10] C. Dwork, L. Stockmeyer, A time-complexity gap for two-way probabilistic finite state automata, SIAM Journal of Computing, 19:1011–1023, 1990.
[11] C. Dwork, L. Stockmeyer, Finite state verifiers I: the p ower of interaction, Journal of the ACM, 39(4):800–828, 1992.
[12] W. Feller, An Introduction to Probability Theory and Its Applications, vol. I. Wiley, 1967.
[13] R. Freivalds, Probabilistic two-way machines, in proceedings of the International Symposium on Mathematical Foundations of Computer Science, volume 188 of Lecture Notes in Computer Science, pp. 33–45. Springer-Verlag, 1981.
[14] A. Greenberg, A. Weiss, A lower bound for probabilistic algorithms for finite state machines, Journal of Computer and System Sciences, 33(1):88–105, 1986.
[15] L. Grover, A fast quantum mechanical algorithm for database search, in proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212–219, 1996.

[16] A. Holevo, Bounds for the quantity of information transmitted by a quantum communication channel, Problemy Peredachi Informatsii, English translation in Problems of Information Transmission 9, 9(3):3–11, 1973.
[17] J. Ka¸neps, R. Freivalds, Running time to recognize nonregular languages by 2-way probabilistic automata, in proceedings of the 18th International Colloquium on Automata, Languages and Programming, volume 510 of Lecture Notes in Computer Science, pages 174–185, 1991.
[18] A. Kitaev, Quantum computations: algorithms and error correction, Russian Mathematical Surveys, 52(6):1191–1249, 1997.
[19] A. Kondacs, J. Watrous, On the power of quantum finite state automata, in proceedings of the 38th Annual Symposium on Foundations of Computer Science, pp. 66–75, 1997.

AUTHORS' BIOGRAPHY

Mr. Tirtharaj Dash is currently an M.Tech Scholar in the Department of Computer Science and Engineering at Veer Surendra Sai University of Technology (Formerly UCE), Burla, India. He completed his B.Tech (IT) from National Institute of Science and Technology, Berhampur in the year 2012. He has contributed around 15 research and review papers in Conferences and reputed journals. His areas of research include Pattern Recognition, Algorithms, Soft Computing, Parallel Processing and Quantum Computation.

Ms. Tanistha Nayak graduated from National Institute of Science and Technology with a Bachelor Degree in Technology (B.Tech-IT) in the year 2012. Her research interests are based on the techniques of Soft Computing approaches and Quantum Computing. She has already published around 12 research and review papers on these fields in Conferences and journals.