Original Research Article

Deaf Machine Theory

Abstract

In this paper author presents an abstract model represents an ideal conceptual model to describe one edge of computation theory. The deaf machine is an automaton that has not any acceptance state, Deaf machine may have one or many normal states or even have infinite states (uncountable normal states). The deaf machine can not recognize any languages, either formal or other. So, there is at least one finite state machine, FSM without accepting state, cannot recognize any language neither accepts an empty string Ɛ.

Keywords: Theory of Computation, Computer Science, Deaf Machine, Automata, FSM.

1. Deaf Machine in Computation Theory

The computational theory in computer science examines the possibility of solving problems efficiently through a computer and examines what the computer can calculate currently and what they can develop to solve problems. Computational theory deals with the mathematical models of computing. To produce a systematic study of computation, computer scientists form an abstract mathematical model of computers called model of computation [2, 3]. One of these model is the deterministic finite automata DFA that works with finite state machine FSM. This model consists of five tuples, the first tuple is represents the set of states that is governed by rules of transition from one state to another according to input symbol, the movement is done by the transition function as an example the transition function of moving the process from state x to the next state y, when the input symbol is 1 is $\delta(x, 1) = y$.[4, 5] from definition: Let machine M is a deterministic finite automaton, then M has a 5-tuple, $(Q, \Sigma, \delta, q_0, F)$, [6, 7] consisting of:

1. a finite set of states (Q)
2. a finite set of input symbols called the alphabet (Σ)
3. a transition function ($\delta : Q \times \Sigma \rightarrow Q$)
4. an initial or start state ($q_0 \in Q$)
5. a set of accept states ($F \subseteq Q$)

Let machine M figure 1, If alphabet of the machine $\Sigma = \{0, 1\}$, then:

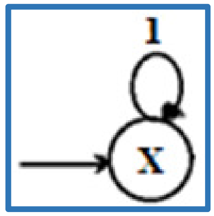

Figure 1- Machine M.

M= ({x}, {1}, δ, x, {})

δ:

	1
X	x

δ (x, 1) = x, the transition function represents the movement from one state to another except the case of machine M. The important part that is presented here, is the accept states set, it is allowed to be {} or ф (there is not accept states) [1]. This thing is the problem and the solution at the same time, the problem is the machine does not recognize any language neither empty string Ɛ nor the empty language. As an example the healthy human ears, but the person does not recognize any human language. He can hear but he cannot understand. From the other side the solution is the DFA machine has not any accept state, but it accomplishes a model that represents the deaf machine in computation theory [4, 5]. Because this abstract model represents an ideal conceptual model to describe one edge of computation theory. The deaf machine is an automaton that has not any acceptance state, figure 1 for example. Deaf machine may have one or many normal states or even have infinite states (uncountable normal states) [8, 9 and 10]. The deaf machine can not recognize any languages, either formal or other.

2. Conclusion

Deaf machine is an abstract model represents an ideal conceptual model to describe one edge of computation theory. The deaf machine is an automaton that has not any acceptance state. Deaf machine may have one or many normal states or even have infinite states (uncountable normal states). The deaf machine can not recognize any languages, either formal or other. This leads to conclude that there is at least one finite state machine, FSM without accepting state, cannot recognize any language neither accepts an empty string. One example of an application of this model is to halt or break function in programming.

3. References

1. Sipser. 2013, *Introduction to the Theory of Computation*, 3rd edition, Cengage Learning, P 36.
2. Rabin ‹Michael O. June 2012. Turing, Church, Gödel, Computability, Complexity and Randomization: A Personal View.
3. AHO, A. V., SETHI, R., AND ULLMAN, J. D.1986, Compilers: Principles, Techniques, Tools. Addison-Wesley.
4. Donald Monk. 1976. *Mathematical Logic*. Springer-Verlag. ISBN 9780387901701.
5. Henry Gordon Rice. 1953. "*Classes of Recursively Enumerable Sets and Their Decision Problems*". Transactions of the American Mathematical Society. American Mathematical Society. 74 (2): 358–366. JSTOR 1990888. doi: 10.2307/1990888.
6. LUBY, M. Pseudo randomness and Cryptographic Applications. Princeton University Press, 1996.
7. LUND, C., FORTNOW, L., KARLOFF, H., AND NISAN, N. Algebraic methods for interactive proof systems. Journal of the ACM 39, 4 (1992), 859–868.
8. MILLER, G. L. Riemann’s hypothesis and tests for primality. Journal of Computer and System Sciences 13 (1976), 300–317.
9. NIVEN, I., AND ZUCKERMAN, H. S. An Introduction to the Theory of Numbers, 4th ed. John Wiley& Sons, 1980.
10. S. Barry Cooper (2004). Computability Theory. Chapman and Hall/CRC. ISBN 1-58488-237-9.