SEMIGROUPS AND AUTOMATA

Dr. Anand Jha
Deptt. of Mathematics,
+2 Sarvodaya High School, Darbhanga-846004.

Abstract

The development of technology in the area of electromechanical and electronic machines, particularly computers has had a great influence on automata theory. Here we are mainly concerned with one sub-discipline of automata theory, namely the algebraic theory of automata, which uses algebraic concepts to formalize and study certain types of finite state machines. One of the main algebraic tools used to do this is the theory of semigroup.

Key words : Semigroup, Automata, homomorphism.

Introduction :

We meet automata or machines in various forms such as calculating machines, computer, money changing device. We shall indicate what is common to all automata and describe an abstract model which will be amenable to mathematical treatment. We study close relationship between automata and semigroup.

1. Definition :

A semiautomaton is a triple S = (Z, A, δ), consisting of two nonempty sets Z (the set of states) and A (the input alphabet), and a function δ : Z x A → Z, called the next-state function of S.

The above definition is very much an abstraction of automata in the usual sense. Historically, the theory of automata developed from concrete automata in communication techniques, nowadays it is a fundamental science. If we want "outputs", then we have to study automata rather than semiautomata.

2. Definition :

An automaton is a quintuple A = (Z, A, B, δ, λ) where (Z, A, δ) is a semiautomaton, B is a nonempty set called the output alphabet and λ : Z x A→ B is a function, called the output function.

If z $\in$ Z and a $\in$ A, then we interpret δ(z, a) $\in$ Z as the next state into which z is transformed by the input a. We consider λ(z, a) $\in$ B as the output of z resulting from the input a. Thus if the automaton is in state z and receives the input a, then it changes to state δ(z, a), producing an output λ(z, a).

3. Definition :

A (semi-)automaton is finite if all sets Z, A and B are finite; finite automata are also called Mealy automata. If a special state $z_0 \in Z$ is fixed, then the (semi-) automaton is called initial and z_0 is the initial state. We write (Z, A, δ, z_0) in this case. An automaton with λ depending only on z is called a Moore automaton.

In practical examples it often happens that states are realized by collections of switching elements each of which has only two states (e.g., current-no current), denoted by 1 and 0. Thus Z will be the Cartesian product of several copies of Z_2. Similarly for A and B. Sometimes δ and λ are given by formulas. Very often, however, in finite automata, δ and λ are given by tables.

4. Example (Cafeteria Automaton) :

We consider the following situation in a student's life. The student is angry or bored or happy; the cafeteria is closed, offers junk food or good dishes. If the cafeteria is closed, it does not change the student's mood. Junk food offers "lower" it by one "degree" (if he is already angry, then no change), good food creates general happiness for him. Also there are two outputs b_1, b_2 with interpretation

b_1 ……. "student shouts", b_2 …… "student is quiet"

we assume that the student only shouts if he is angry and if the cafeteria only offers bad food. Otherwise he is quiet, even in state z_3.

We try to describe this rather limited view of a student's life in terms of an automaton S = (Z, A, B, δ, λ). We define Z = {z_1, z_2, z_3} and A = {a_1, a_2, a_3}, with

z_1 … "student is angry". a_1 …... "the cafeteria is closed"

z_2 … "student is bored". a_2 …... "the cafeteria offers junk food"

z_3 … "student is happy". a_3 …... "the cafeteria offers good food"

we then obtain

δ	a_1	a_2	a_3
z_1	z_1	z_1	z_3
z_2	z_2	z_1	z_3
z_3	z_3	z_2	z_3

λ	a_1	a_2	a_3
z_1	b_2	b_1	b_2
z_2	b_2	b_2	b_2
z_3	b_2	b_2	b_2

In a computer it would be rather artificial to consider only single input signals. Programs consist of a sequence of elements of an input alphabet. Thus it is reasonable to consider the set of all finite sequences of elements of the set A, including the empty sequence A. In our study of automata we extend the input set A to the free monoid A* with A as identity.

We also extend δ and λ from Z x A to Z x A* by defining for $z \in Z$ and $a_1, a_2, \dots, a_r \in A$: $\delta^*(z, A) := z$, $\delta^*(z, a_1) := \delta(z, a_1)$, $\delta^*(z, a_1 a_2) := \delta(\delta^*(z, a_1), a_2)$ etc. and $\lambda^*(z, A) := A$, $\lambda^*(z, a_1) := \lambda(z, a_1)$, $\lambda^*(z, a_1 a_2) := \lambda(z, a_1)\ \lambda^*(\delta(z, a_1), a_2)$ and so on. In this way we obtain functions δ^* : $Z \times A^* \to Z$ and λ^*: $Z \times A^* \to B^*$. The semiautomaton S = (Z, A, δ) (the automaton A = (Z, A, B, δ, λ)) is thus extended to the new semiautomaton S* := (Z, A*, δ^*) (automaton A^* = (Z, A*, B*, δ^*, λ^*), respectively). We can easily describe the action of S and A if we let $z \in Z$ and $a_1, a_2, \dots \in A$.

$z_1 := z,$

$z_2 := \delta(z_1, a_1)$

$z_3 := \delta^*(z_1, a_1 a_2) = \delta^*(\delta(z_1, a_1), a_2) = \delta(z_2, a_2),$

$z_4 := \delta(z_3, a_3), \dots\dots$

If the (semi-)automaton is in state z and an input sequence $a_1a_2\dots a_r \in A^*$ operates, then the states are changed from $z = z_1$ to $z_2, z_3, \dots$ until the final state z_{r+1} is obtained. The resulting output sequence is $\lambda(z_1,a_1)\ \lambda(z_2,a_2) \dots \lambda(z_r,a_r)$.

BIBLIOGRAPHY :

1. Heriken, R. (1994). The Universal Turing Machine: A Half-Century Survey, Wilen: Springer-Verlag.
2. Lidl, R. and G. Pilz (1984). Applied Abstract Algebra (1st ed.). New York: Springer-Verlag. Russian edition: Ekaterinenburg, 1997.
3. Lidl, R. and H. Niederreiter (1994). Introduction to Finite Fields their Applications. Cambridge University Press.
4. Mendelson, E, (1970). Boolean Algebras and Switching Circuits. Schaum's Outline Series. New York: McGraw-Hill.
5. Spindler, K.H. (1994). Abstract Algebra with Applications, Vols I, II. New York : Dekker.