UF MATH CIRCLE SEPTEMBER 9, 2023

1. Finite State Automata

Believe it or not, the following is a machine:

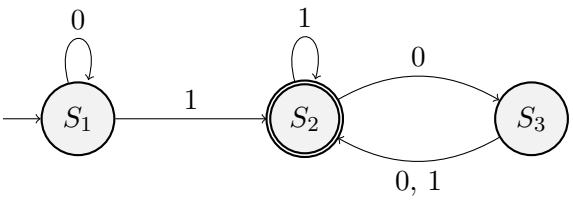

Figure 1. A Finite State Automata

The circles represent the three *states* of the *finite state automata*. You feed an input (in this case, a *string* of 0's and 1's into the machine. The machine *transitions* between states based on the next *letter* of the input as indicated by the arrows. The *starting state* is S_1 (indicated by the incoming arrow), and the *accepting state* is S_2 (indicated by the double circle). A string of 0's and 1's is accepted if after using all of the symbols of the input the current state is an accepting state.

Example 1. The string 0110 is not accepted.
start: the machine starts in S_1
read 0: it transitions from S_1 to S_1
read 1: it transitions from S_1 to S_2
read 1: it transitions from S_2 to S_2
read 0: it transitions from S_2 to S_3.
It's used all four letters of the string and is not in an accepting state, so 0110 is not accepted.

(1) Is the string 01100 accepted? What about 0? What about 11?

(2) Can you find a string of length ten which is accepted?

Remark 2. Details: the *alphabet* is the set of symbols used to form strings. It can be any finite set you like (digits, letters, symbols, etc.) The automata must have a finite number of states, and only one starting state. There can be multiple accepting states.

Sometimes we leave out arrows to simplify the diagram: if there is no transition matching the next letter of the input, the machine does not accept the input.

2. ϵ-ARROWS

The special symbol ϵ means "nothing". It denotes a string with no letters. When it appears on an arrow, it means that the automata may transition *without* using the next letter of the input. This is not mandatory.

If there are ϵ-arrows, there can be multiple ending states depending on how ϵ-arrows are used. A string is *accepted* if (at least) one of these ending states is an accepting state. A string is not accepted only if there no possible paths to an accepting state.

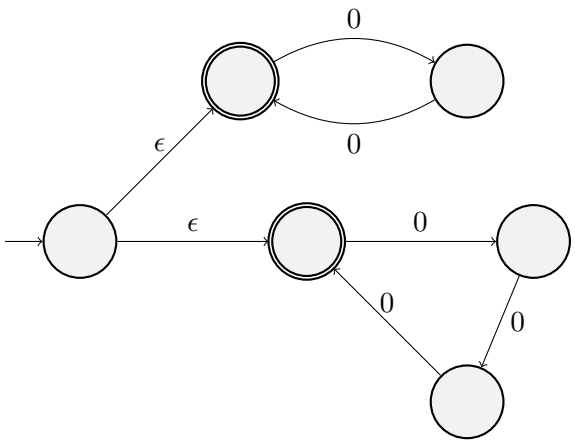

FIGURE 2. A Finite State Automata with ϵ-arrows

Example 3. Figure 2 shows a finite state automata where the alphabet is just 0. The string 00 is accepted, but the string 00000 is not accepted.

(3) Is 000 accepted?

(4) Are there any strings which are accepted regardless of which ϵ-arrow is used?

(5) Which strings are accepted by the finite state automata in Figure 2? (This will depend on the number of 0's)

Remark 4. Similarly to ϵ-arrows, sometimes it is convenient to allow multiple transitions out of a state all labeled by the same symbol. The string is accepted if there is some choice of transitions which ends at an accepting state. Automata with ϵ-arrows or this behavior are called *non-deterministic* automata, reflecting the necessity of choices. Ones without are *deterministic*.

3. Building Automata

Draw a finite state automata which accepts the specified strings and rejects all others.

(6) Alphabet: a, b. Only the strings aa,ab, and bba (and nothing else).

(7) Alphabet: 0,1. Strings with an odd number of 0's.

(8) Alphabet: a,b. Strings where each a is immediately followed by b.

(9) Alphabet digits 0-9. Strings representing integers which are multiples of 5.

Here are more to play with. The shorthand a^n means n copies of a next to each other: $aaaa\ldots a$.

(10) Alphabet: 0, 1. Strings with an odd number of 0's and even number of 1's.
(11) Alphabet: Q and D (representing quarter and dollar). Strings where the total value is at least 175 cents. (If there are x Q's and y $D's$, then $25x + 100y \geq 175$.)
(12) Alphabet: a,b. String where the number of a's is odd and also the number of b's is not a multiple of 3.
(13) Alphabet: a,b. Strings of the form $a^n b^m a^k$ for some non-negative integers n, m, k (i.e some number of a's followed by some number of b's followed by some number of a's).
(14) Alphabet a,b. Strings of the form $a^n b^m$ OR $b^n a^m$.
(15) Alphabet 0, 1. Strings of the form $0^n 1^n$ (n 0's followed by the same number of 1's)
(16) Alphabet 0, 1. Strings that start with 1 and end with 0.
(17) Alphabet a,b. Strings which never include the smaller string aba inside of them.
(18) Alphabet a,b,c. Strings which start with a, end with c, and contain at least one b.
(19) Alphabet 0-9. Strings representing integers which are multiples of 25.
(20) Alphabet 0-9. Strings which are multiples of 3.
(21) Alphabet @ !. Strings where number of @'s minus the number of !'s is a multiple of 3.
(22) Alphabet letters A-Z. Strings which are palindromes (read the same forwards and backwards).
(23) Alphabet 0,1 Strings of the form $1^k y$, where y is a string of 0's and $1's$ containing at least k 1's for some positive integer k.
(24) Alphabet 0,1 Strings of the form $1^k y$, where y is a string of 0's and $1's$ containing at most k 1's for some positive integer k.
(25) Alphabet 0-9,+,= Strings of the form $x = y + z$ where x, y, z are integers and x is the sum of y and z.

4. Extra Challenges

The *language* recognized by a finite automata is the set of all strings it accepts. A language is recognized by a finite state automata is called a regular language.

(101) Show that the union of two regular languages L_1, L_2 is regular. It is denoted $L_1 \cup L_2$, and consists of strings in L_1 or L_2 (or both). I.e given two finite state automata construct a new finite state automata which accepts precisely the strings accepted by either machine.

(102) Show that the concatenation of two regular languages L_1, L_2 is regular. It is denoted L_1L_2, and consists of strings obtained by taking a string of L_1 and appending a string of L_2.
(So for example, if L_1 is the language of strings that only contain a's, and L_2 is the language of strings that only contain b's, then L_1L_2 is the language of strings of the form aaa...abbb...b, with any number of a's and b's.)

(103) Show the star of a regular language L is regular. It is denoted L^*, and consists of all strings formed by putting together any number of strings in L. Any number can include 0.
(So for example, if $L_1 = \{ab, b\}$ then L_1^* is the language of strings which don't have consecutive a's.)

(104) Let $\Sigma = \{a, b\}$ be the alphabet. What is a simple description of the language

$$a\Sigma^* a \cup b\Sigma^* b \cup a \cup b$$

(105) Draw a finite automata which recognizes the language $((101)^*(01)^*) \cup \{1\}$.

Remark 5. The operators of union, concatenation, and star define what are called regular expressions. The languages which can be described by regular expressions are precisely those which are recognized by finite state automata.

(106) Suppose L is a regular language, recognized by a finite state automata with N states. Show that any string in L whose length is more than N will cause the finite state automata to repeat a state S.

(107) Break that string up as xyz, where x causes the automata to reach S, y causes it transition from S to S, and z causes it to transition to an accepting state. Show that $xyyz$, $xyyyz$, etc. are also accepted by the automata.

(108) Show that the language $\{0^n1^n : n \geq 0\}$ (n 0's followed by the same number of 1's) is not regular i.e. not recognized by any finite automata.

(109) What is the relationship between the following finite automata?

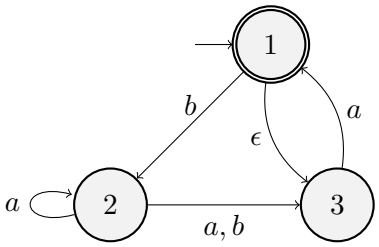

FIGURE 3. A Non-Deterministic Finite Automata

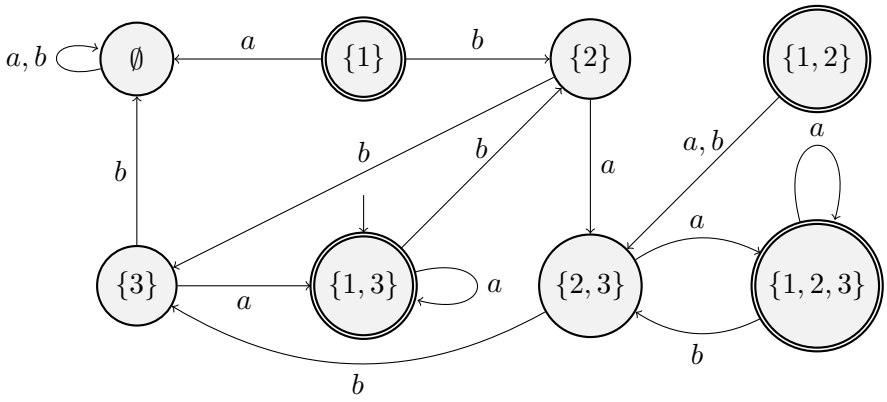

FIGURE 4. A Deterministic Finite Automata