

Lecture 1.2: Basic concepts and Terminology-1

Presenter: Azwad Anjum Islam (AAI)

Scribe: Mujtahid Al-Islam Akon (AKO)

In this lesson, you will learn about some of the basic concepts and terminologies related to the Automata Theory.

Families of Automata

There are four major families of automaton based on their capabilities & limitations:

- a. Finite State Machine /Automaton (FSM)
- b. Pushdown Automaton (PDA)
- c. Linear Bounded Automaton (LBA) and
- d. Turing Machine (TM)

These automata are not mutually exclusive. Their proper relationship is depicted below-

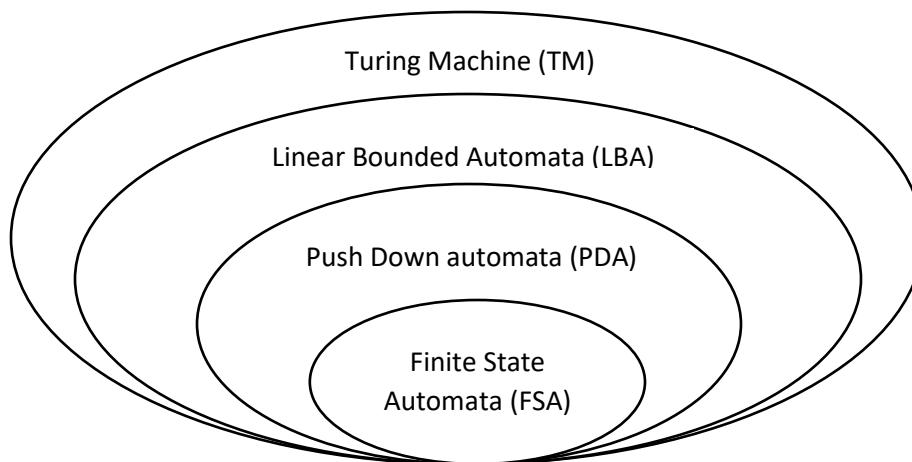


Figure 1: Relation among different families of automata

- The most basic type of automaton is the finite state machine.
- Then the pushdown automaton which encompasses the finite state machine i.e. anything that a finite state machine can do can also be done in a pushdown automaton.
- Then the linear bounded automaton
- At the topmost there is the Turing machine which is the most generic and the most powerful type of automaton. Anything that can be solved using a computer in theory can also be solved using a Turing machine.

The final state machines can further be divided into two parts-

- a. **Finite state machines with output:** they take an input and do some processing and finally generate an output. These are further divided into-
 - i. Mealy machine

- ii. Moore machine
- b. **Finite state machines without output:** they do not have an output and will only give you either a yes or no answer. We shall learn this type at length in this course. These can further be divided into two types-
 - i. Deterministic FSA
 - ii. Non-deterministic FSA

Finite state machines without output

A finite state machine without output works on the following principle-

- First take an **input**
- **Process** the input based on some predefined rules
- Either **accept** (a Yes) or **reject** (a No) the input.

The input can be anything e.g.-

- a string of characters e.g. HELLO
- a stream of numbers e.g. 011011221012
- a string of operations e.g. left right right right. This type of inputs is nothing but a set of strings.
- etc. ...

Some Definitions

Symbols: The smallest building block in the theory of computation and a symbol can be anything e.g.

- Characters like a, b, अ, %, ∞, £, ÷, 😊
- Numbers like 0, 1, ↘
- Even pictograms like ⚡ ↴ ⚡
- Etc.

Strings: Strings are actually what goes as inputs into the automata. A string is a sequence of symbols e.g-

- A sequence of characters like hello
- A sequence of numbers like 010010
- A sequence of pictograms ⚡ ⚡ ↴ ⚡ ↴ ↴
- Etc.

Few more definitions related to strings:

- **Empty String:** The string that has a length 0 i.e. this string has no characters. This is often denoted by ϵ . Remember that an absence of string is still a string which is ϵ .
- **Substring:** any consecutive part of a string e.g. ϵ , ALO, LOH, OMORA, ALOHOMORA, all are some of the substrings generated from the string ALOHOMORA.
- **Subsequences:** any part of a string that's in the right order i.e. they are part of the string too, however, they don't have to be consecutive but they must be in the right order.
 - ϵ , ALH, MR, ALORA, LOMA, ALOHOMORA, all are subsequences of ALOHOMORA.

- **AHL** cannot be a subsequence of **ALOHOMORA** because you cannot find A, H & L in the same order in **ALOHOMORA**.
- **Prefix:** any length of substring starting from the beginning. E.g. ϵ , A, AL, ALOHO, ALOHOMORA are some of the prefixes of the string **ALOHOMORA**.
- **Suffix:** last part of a string of any length. E.g. ϵ , A, RA, MORA, ALOHOMORA, each of them is a suffix of the string **ALOHOMORA**.

Notice that ϵ and **ALOHOMORA**, each of them is considered a substring, a subsequence, a prefix and a suffix of **ALOHOMORA**, however, they are not **proper substring**, **proper subsequence** etc. They are called **improper subsequences or substrings or prefixes or suffixes**.

- **Concatenation of strings:** Concatenation means joining together. When we concatenate two strings, we put one string after the other. If we concatenate the two strings 'xy' and 'ab' in this order, we will get 'xyab'.
- **Identity of concatenation** is ϵ i.e. if we concatenate any string with epsilon on either side of a string, we just end up with string itself.
- **Self-concatenation:** Concatenation can be done with itself.

Let, a string W be ab (also expressed as W^1). Then, concatenation of W with W itself will be abab and can be expressed as WW or W^2 .

Similarly,

$$W^3 = WWW = ababab$$

$$W^4 = WWWW = abababab$$

.

.

.

$$W^\infty = WWW\ldots = ababab\ldots$$

Find out by yourself: What does W^0 mean?

Alphabets: Alphabet is a finite set of all the symbols that a language can have. Alphabets are often expressed by ' Σ '. Some examples of alphabet include-

- Alphabet for English numerals = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
- Alphabet for English language = {a, b, ..., z, A, B, ... Z, ..., !, ?, ...}

Language: a language is a set of string. Any set of strings will be called a language. A language can either be –

- Finite: The number of elements in the language will be finite e.g. {a, bb, aba}
- Infinite: The number of elements will be infinite e.g. {a, aa, aaa, aaaa,}, {0, 10, 101, 0100, 0000,}, etc.

All the strings in a language (either finite or infinite) generally maintains a common property. This property may or may not always be visible but in most cases this common property exists. For example-

- {a, aa, aaa, aaaa,} => a string having all possible combinations of 'a'
- {1, 2, 3, 4, 5, ...} => natural numbers
- {0, 1, 10, 11, 100, ...} => binary numbers.

- $\{6, 28, 496, 8128, \dots\}$ => No obvious common property, however, there is one. It is the sequence of all the [perfect numbers](#).

Languages are generally infinite. For example- English language has infinite words (strings). However, languages don't always need to be infinite. For example- a baby only knows a certain number of words e.g. two words like papa or mama. So, the language of that child is finite.
