**Operations on Formal Language**

Formal Languages

A language can be seen as a system suitable for expression of certain ideas, facts and concepts. For formalizing the notion of a language one must cover all the varieties of languages such as natural (human) languages and programming languages. Let us look at some common features across the languages.

One may broadly see that a language is a collection of sentences; a sentence is a sequence of words; and a word is a combination of syllables. If one considers a language that has a script, then it can be observed that a word is a sequence of symbols of its underlying alphabet. It is observed that a formal learning of a language has the following three steps.

1. Learning its alphabet - the symbols that are used in the language.

2. Its words - as various sequences of symbols of its alphabet.

3. Formation of sentences - sequence of various words that follow certain rules of the language.

In this learning, step 3 is the most difficult part. Let us postpone to discuss construction of sentences and concentrate on steps 1 and 2. For the time being instead of completely ignoring about sentences one may look at the common features of a word and a sentence to agree upon both are just sequence of some symbols of the underlying alphabet. For example, the English sentence.

"The English articles - a, an and the – are categorized into two types: indefinite and definite."

may be treated as a sequence of symbols from the Roman alphabet along with enough punctuation marks such as comma, full-stop, colon and further one more special symbol, namely *blank-space* which is used to separate two words. Thus, abstractly, a sentence or a word may be interchangeably used for a sequence of symbols from an alphabet. With this discussion we start with the basic definitions of alphabets and strings and then we introduce the notion of language formally.

Further, in this chapter, we introduce some of the operations on languages and discuss algebraic properties of languages with respect to those operations. We end the chapter with an introduction to finite representation of languages via regular expressions.

**OPERATIONS:**

**Languages :**A language over an alphabet is a set of strings over  that alphabet. Therefore, a language *L* is any subset of Σ\*. That is, any L⊆ Σ\* is a language.

**Example :**

1. *F* is the empty language.
2. Σ\* is a language for any Σ.
3. {*e*} is a language for any Σ.  Note that, Ø ≠ { *e* }. Because the language F does not contain any string but {*e*} contains one string of length zero.
4. The set of all strings over { 0, 1 } containing equal number of 0's and 1's.
5. The set of all strings over {*a*, *b*, *c*} that starts with *a*.

**Convention :** Capital letters *A*, *B*, *C*, *L*, etc. with or without subscripts are normally used to denote languages.

**Set operations on languages :** Since languages are set of strings we can apply set operations to languages. Here are some simple examples (though there is nothing new in it).

**Union:**A string https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-29.gif   iff  https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-30.gif or https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-31.gif

**Example:**  { 0, 11, 01, 011 } https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif { 1, 01, 110 } = { 0, 11, 01, 011, 111 }

**Intersection:**  A string  https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-33.gif   iff  https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-34.gif  and https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-35.gif.  
  
**Example:** {0, 11, 01, 01} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-36.gif {1, 01, 110} = {01}

**Complemen**t**:** Usually, https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-20.gif is the universe that a complement is taken with respect to. Thus for a language *L*, the complement is *L*(bar) = {https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-37.gif | https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-38.gif }.

**Example:**Let *L* = {*x*| |x| is even}. Then its complement is the language {https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-37.gif | |*x*| is odd }.  
Similarly we can define other usual set operations on languages like relative complement, symmetric difference, etc.

**Reversal of a language :**  
The reversal of a language *L*, denoted as https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-39.gif, is defined as:  https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-40.gif.

**Example:**

1. Let *L* = {0, 11, 01, 011}. Then  https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-39.gif  = {0, 11, 10, 110}.
2. Let *L* = { https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-43.gif | *n* is an integer}. Then https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-39.gif = { https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-43.gif | *n* is an integer}.

**Language concatenation**: The concatenation of languages https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-41.gif and https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-42.gif is defined as  
https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-44.gif = {*xy* | *https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-45.gif* and *https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-46.gif*}.

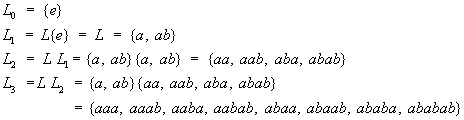
**Example:**  {*a*, *ab*} { *b*, *ba* } = { *ab*, *aba*, *abb*, *abba* }.

Note that ,   
     1.    https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-47.gif   in general.  
      2.    https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-48.gif  
      3.    https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-49.gif

**Iterated concatenation of languages :**  Since we can concatenate two languages, we also repeat this to concatenate any number of languages. Or we can concatenate a language with itself any number of times. The operation https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-62.gif denotes the concatenation of *L* with itself *n* times. This is defined formally as follows:

**https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-50.gif  
https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-52.gif**

**Example :**  Let *L* = {*a*, *ab*}. Then according to the definition, we have



and so on.

**Kleene's Star operation:**The Kleene star operation on a language *L*, denoted as https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-54.gif is defined as follows**:**

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-54.gif= ( Union *n* in *N*) https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-55.gif

    = https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-56.gif

    = {*x* | *x* is the concatenation of zero or more strings from *L*}

Thus https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-54.gif is the set of all strings derivable by any number of concatenations of strings in *L*. It is also useful to define

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-57.gif= https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-58.gif, i.e., all strings derivable by one or more concatenations of strings in *L*. That is

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-57.gif =(Union *n* in *N* and *n* >0) https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-59.gif  
      = https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-60.gif

**Example :**  Let  *L* = { *a*, *ab* }. Then we have,

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-54.gif = https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-61.gif

      = {*e*} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif {*a*, *ab*} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif {*aa*, *aab*, *aba*, *abab*} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif…

https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-57.gif = https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-60.gif

     = {*a*, *ab*} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif {*aa*, *aab*, *aba*, *abab*} https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-32.gif…

**Note :**  *e* is in https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-54.gif, for every language *L*, including*.*

The previously introduced definition of https://nptel.ac.in/content/storage2/courses/106103070/modules/module-1/lecture-1/topic-1/eqns/e-20.gif is an instance of Kleene star.

**RELEVANT READING MATERIAL AND REFERENCES:**

**Source Notes:**

1. <https://nptel.ac.in/courses/106/103/106103070/>
2. <https://www.iitg.ac.in/dgoswami/Flat-Notes.pdf>

**Lecture Video:**

1. https://www.youtube.com/watch?v=YDZWHFznLFc

**Online Notes:**

1. <https://www.iitg.ac.in/dgoswami/Flat-Notes.pdf>

**Text Book Reading:**

1. Martin J.C., “Introduction *to Languages and Theory of Computation*”, Tata McGraw-Hill Publishing Company Limited, 3rd Edition.
2. Hopcroft J.E. and Ullman J.D., “Introduction *to Automata Theory Languages and Computation*”, Narosa Publications.