Formal Languages and Automatons

Attila Házy

6 October, 2022

Alphabet, word

- A (generally finite), not empty set is called **alphabet**.
- Items of an alphabet (items comprising the set)are called **symbols** (characters, letters, punctuation marks).
- The finite sequence of items chosen from an alphabet is called a **word** over the specific alphabet. Words are denoted by a Greek letter. e.g.: α is a word over the A alphabet. ($\alpha \ll A$)
- The length of an α word over an alphabet is the number of symbols in it.
- The word ε over an alphabet is called empty word. The symbol of the empty word is usually ϵ or ε a Greek letter (epsilon).

Finite Words

If A is a finite not empty set, it can bee seen as an alphabet. As we have mentioned earlier items of an alphabet are called letters or symbols. The sequence chosen from the elements $a_0, a_1, \ldots, a_n$ of set A are called words over alphabet A. Also, as you could see earlier the length of such words is the same as the number of symbols in them.
This can be given in the $|a_1 \ldots a_n|$, or the $L(a_1 \ldots a_n)$ forms but it is much easier to simply denote words with letters $\alpha, \beta, \ldots$.
Then the length of the word is given in the $L(\alpha)$, or in the $|\alpha|$ form.

Finite Words

A specific word comprises of the symbols of the particular alphabet raised to a power:
A^* means all the words over the A alphabet (including the empty word).

$$A^+ = A \setminus \{\varepsilon\},$$

and

$$A^n = \{\alpha \in A^* \mid |\alpha| = n\} = \{a_1, a_2, \ldots, a_n | a_i \in A\}.$$

This implies that the A^+ is the set of words over A, except for the empty word. A^n means the set of words with the length of n and $A^0 = \{\varepsilon\}$, where $|\varepsilon|$, namely $L(\varepsilon) = 0$.

Concatenation

The first operation on words is concatenation (multiplication of words), which simply means that we form new words from two or more words (these can be seen as parts of a compound) forming a compound.

Concatenation of the words α and β over alphabet *A* is the word γ over alphabet *A* which we get by writing the symbols of word β after the symbols of word α. Concatenation is denoted with $+$.

Example: If $\alpha =$ "apple" and $\beta =$ "tree" then $\alpha + \beta =$ "appletree".

Concatenation

If you want to define operations informally, then the following definition will be appropriate:

Definition 1 (Concatenation)

Consider α, and β words over the *A* alphabet, namely words constructed from symbols of the alphabet. The result of $\alpha + \beta$ (or simply $\alpha\beta$) is the concatenation of the two words, so that $\gamma = \alpha\beta$, where $|\gamma| = |\alpha| + |\beta|$, so the length of the new word is the sum of the length of the two components.

Concatenation

Now, let us have a look at the fully formal definition:

Definition 2 (Concatenation)

If $\alpha = a_1, a_2, \ldots, a_n$, and $\beta = b_1, b_2, \ldots, b_m$ are words over alphabet A then:

$$\gamma = \alpha\beta = a_1 a_2 \ldots a_n b_1 b_2 \ldots b_m.$$

Properties of Concatenation

Consider $\alpha \ll A$ (word α over alphabet A):

- $\alpha^0 = \varepsilon$ (any word to the power of zero is the empty word).
- $\alpha^n = \alpha + \alpha^{n-1}$ ($n \geq 1$) (any word to the power of n is the n times concatenation of the word)
- word α is the prefix of γ and since the length of α is not zero ($|\alpha| \neq 0$), this is a real prefix
- word β is the suffix of γ and since the length of β is not zero ($|\beta| \neq 0$), it is a real suffix
- the operation is associative so $\alpha(\beta\gamma)$ is equivalent with the $(\alpha\beta)\gamma$ operation.
- the operation is not commutative so $\alpha\beta \neq \beta\alpha$.
- the operation has a neutral element so $\varepsilon\alpha = \alpha\varepsilon$, and it is monoid with the A^* alphabet or more precisely with the set

Raising Words to a Power

Definition 3 (Power of Words)

$$\alpha^0 = \varepsilon$$

and

$$\alpha^n = \alpha^{n-1}\alpha.$$

Then, if $n \geq 1$, namely the nth power of word α is the n times concatenation of the word.

From this operation we can also conclude several things:

- word α is primitive if it is not the nth power of any other word, namely α is primitive if $\alpha = \beta^n$ and $\beta \neq \varepsilon \Rightarrow n = 1$. For example $\alpha = abcdefgh$ is primitive but word 123123123 is not because $\alpha = (123)^3$
- Words α, and β are each others' conjugates, if there is a $\alpha = \gamma\delta$, and $\beta = \delta\gamma$.
- $\alpha = a_1, a_2, \ldots, a_n$ is periodic if there is a $k > 1$ number, so that for the $a_i = a_{i+k}, i = 1, 2, \ldots, n-k$ values, so that k is the period of word α. The smallest period of word $\alpha = 1231231$ is 3.

Reversal of Words

Definition 4 (Reversal of Words)
In case of word $\alpha = a_1, a_2, \ldots, a_m$ word $\alpha^T = a_m, a_{m-1}, \ldots, a_1$ is the reversal of α. If $\alpha^T = \alpha$, the word is a palindrome.

It can also be derived from the above that $(\alpha^T)^T = \alpha$, so by reversing the word α twice we get the original word.

Examples

word *abccba* is a palindrome word texts *asantatnasa*, or *amoreroma* are also palindrome texts and upper case and smaller case letters are considered equivalent.

Subwords

Definition 5 (Subword)

Word β is subword of word α if there are words γ, and δ in a way that $\alpha = \gamma\beta\delta$, and $\gamma\delta \neq \varepsilon$, namely if β is a real subword of α.

Subwords

Definition 6 (Subwords with Various Length)

Denote the set of k length subwords of word α by $R_k(\alpha)$. $R(\alpha)$ is the set of all such subwords so

$$R(\alpha) = \bigcup_{k=1}^{|\alpha|} R_k(\alpha).$$

Example

Let $\alpha = abcd$. Then

- the 1 length subwords of the word are $R_1(\alpha) = \{a, b, c, d\}$,
- the 2 length subwords are $R_2(\alpha) = \{ab, bc, cd\}$,
- the 3 length are $R_3(\alpha) = \{abc, bcd\}$,
- and the only 4 length subword is the word itself $R_4(\alpha) = \{abcd\} = \alpha$.

The complexity of words is based on the analysis of their subwords. Based on the form of the word and its subwords, we can define the complexity of the word. The complexity of a word is the multiplicity and variety of its subwords.

Complexity of Words

Definition 7 (Complexity of Words)
The complexity of a word is the number of its subwords of different length. The number of k length subwords of word α is $r_\alpha(k)$.

Maximal Complexity

Definition 8 (Maximal Complexity)
Maximal complexity can only be interpreted on finite words and

$$Max(\alpha) = \max\{r_\alpha(k) \mid k \neq 1\}, \qquad \alpha \in A*,$$

where $A*$ is the Kleene star derived from the particular alphabet. (On infinite words we can interpret bottom or top maximal complexity.)

As a word can have maximal complexity, it can also have global maximal complexity shown in the definition below

Global Maximal Complexity

Definition 9 (Global Maximal Complexity)
Global maximal complexity is the sum of the number of nonempty subwords of a word, namely

$$Tb(\alpha) = \sum_{i=1}^{|\alpha|} r_\alpha(i), \qquad \alpha \in A*.$$

Complexity of Sentences

Now we deal with the complexity of sentences of programs more precisely the language constructions of various programming languages characterizing the particular paradigm. Every programming language contains numerous language elements which can be embedded and which elements can be used one after the other. We can create more complex constructions like functions or methods which also consist of various language elements.

Complexity of Sentences

In the complexity of programs we measure the quality of the source text based on which we can get an insight to its structure, characteristics and the joint complexity of programming elements. Based on complexity we can estimate the cost of testing, developing and changing the program text. Complexity of software can be measured based on the complexity (structure) and size of the program. We can observe the source text in development phases (process metrics), or the ready program based on its usability. This kind of analysis features the end product (product metrics), but it is strongly tied to the source text and to the model based on which the source text was built.

Complexity of Sentences

Structural complexity can also be measured based on the cost of development (cost metrics), or based on the cost of effort (effort metrics) or based on the advancement of development (advancement), or based on reliability (non-reliability (number of errors)). You can measure the source text by defining the rate of reusability numerically (reusable) or you can measure functionality functionality, or usability, however, all complexity metrics focus on the three concepts below:

- size
- complexity
- style.

Problems with Complexity

Thomas J. McCabe pointed out how important the analysis of the structure of the source code was in 1976.
In his article McCabe describes that even the ideal 50 line long modules with 25 consecutive IF THEN ELSE constructions include 33.5 million branches. Such a high number of branches can not be tested within the length of a human lifetime and thus it is impossible to verify the propriety of the program .
The problem reveals that the complexity of programs, the number of control structures, the depth of embedding and all the other measurable attributes of the source code have an important impact on the cost of testing, debugging and modifying.

McCabe's Cyclomatic Complexity Number

The value of the complexity metric of mc_cabe is the same as the number of basic paths defined in the control graph, namely it is the same as the number of possible outputs of the function disregarding the paths of functions within the function.
The Mc Cabe cyclomatic number originally was developed to measure subroutines of procedural languages.

The cyclomatic number of programs is defined as follows:

McCabe's Cyclomatic Complexity Number

Definition 10 [Mc Cabe's cyclomatic number]
The cyclomatic number $V(G)$ of control graph $G = (v, e)$ is

$$V(G) = e - v + 2p,$$

where p denotes the number of graph components, which is the same as the number of linearly coherent cycles in a highly coherent graph.

McCabe's Cyclomatic Complexity Number

Example

Let us have a look at a concrete example of applying a cyclomatic number. Consider our program has 4 conditional branches and a conditional loop with a complex condition, with precisely 2 conditions.

Then the cyclomatic number is the number of conditional choices, so that we add one to the number of conditional decisions and count the complex condition twice. We must do so because we must count all the decisions in our program, so the result of our calculation in this program is **seven**.

Infinite Words

Besides finite words we can also interpret infinite words, which can also be constructed from items of an alphabet, like finite ones.

$$\alpha_w = a_1 a_2 \dots a_n \dots$$

infinite words constructed from $\forall a \in A$ symbols are right infinite, namely the α_w word is right infinite.

Infinite Words

Definition 11 (Infinite Words)

Consider A_w to denote the set of right infinite words, and the set of finite and infinite words over the A alphabet is denote

$$A_{all} = A^* \cup A_w.$$

In this case, the case of infinite words, we can also interpret concepts of subword, prefix and suffix.

Besides words we can also carry out operations with alphabets. These operations are important because through their understanding we can get to the definition of formal languages.

Operations with Alphabets

Definition 12

If A and B are two alphabets, then

$$A * B := \{ab \mid a \in A, b \in B\}.$$

This operation is called **complex multiplication**.

Note

So the complex product of two alphabets is an alphabet whose characters are couplets having the first symbol from the first alphabet and the second one from the second alphabet.

Example

Let $A := \{a, b\}$, and $B := \{0, 1\}$. Then

$$C := A * B := \{a0, a1, b0, b1\}.$$

Based on this, the word over alphabet C is for example $\alpha =$ "$a0b0a1$", and $L(\alpha) = 3$, as that word comprises of three symbols from C" $a0$", a "$b0$", and "$a1$".
At the same time however for example word "$a0aba1$" can not be a word over C because it can not be constructed using the symbols of C only.

Definition

Definition 13

Consider an A alphabet. $A^0 := \varepsilon$, so the zeroth power of every alphabet is a set with one element the ε (empty word).

Definition

Definition 14

$A^n := A * A^{n-1}$ where $n \geq 1$. So the nth power of an alphabet is the n times complex product of the alphabet.

$A^0 = \varepsilon$ is necessary since $A^1 = A * A^0$, and we must get back A!

Note

Based on the above mentioned the 3rd power of the A alphabet is an alphabet whose every element consists of three characters.
Generally: the nth power is an alphabet whose every element has the length of n.

Example

If $A = \{a, b\}$, then $A^2 = \{aa, ab, ba, bb\}$. This set can also be seen as the set of words with a length of 2 over the A alphabet.

Definition

Definition 15

Let $V := \{\alpha \,|\, \alpha \ll A \text{ and } L(\alpha) = 1$. So consider set V the set of words of one **length** over the A alphabet. It is denoted $V^{*1} \ll A$ or V for short.

Definition

Definition 16

The **contextual product** over the V set

$$V \otimes V := \{\alpha\beta \,|\, \alpha \in V \text{ and } \beta \in V\}$$

is the set containing words which are constructed from words in a way that we concatenate every one of them with each other.

Note

In fact set V consists of words with the length of one. Words with a length of 2 comprise set $V \otimes V$.

Definition

Definition 17

$V^0 := \{\varepsilon\}$, and $V^n := V \otimes V^{n-1}$ if $n \geq 1$. Then

$$V* := \bigcup_{i=0}^{\infty} V^i = V^0 \cup V^1 \cup V^2 \cup \ldots$$

set is called the **Kleene star** of "V".

Its elements are the empty word, the words with the length of one and words with the length of two etc...

Definition

Definition 18

The

$$V+ := \bigcup_{i=1}^{\infty} V^i = V^1 \cup V^2 \cup V^3 \cup \dots$$

set is the **positive closure** of of "V".

It is easy to see: $V* = V+ \cup \varepsilon$.

Example

Elements of $V+$ are words with the length of one,length of two etc., so $V+$ does not include the empty word.

- If $V := \{a, b\}$. Then

$$V* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

and

$$V+ = \{a, b, aa, ab, ba, bb, aaa, aab, \ldots\}$$

- $\alpha \in V*$ means that α is a word of arbitrary length $L(\alpha) \geq 0$.
- $\alpha \in V+$ means that α is a word of arbitrary length but it can not be empty $L(\alpha) \geq 1$.
- If $V = \{0, 1\}$, then $V*$ is the set of binary numbers (and contains ε too).
- If $V = \{0\}$, $W = \{1\}$, then $(V \cup W)* = \{(01)^n | n \in \mathbb{N}\}$.

Example

Let $S := \{0, 1, \ldots, 9\}$ as a basis, let us write the regular expression that matches every integer number:

$$(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)^+$$

Note the $(+)$ sign at the end of the expression, which is there to denote the positive closure of the set (expression). Positive closure means that the expression can not generate the empty word, namely you must always have at least one digit, or any number of digits in an arbitrary order.

Definiton

Definition 19

Consider $V^{*1} \ll A$. We call an $L \subseteq V*$ set a formal language over alphabet A.

Definiton

Definition 19
Consider $V^{*1} \ll A$. We call an $L \subseteq V*$ set a formal language over alphabet A.

Note

In fact a formal language is the subset of the set of words of arbitrary length over a particular alphabet, namely a formal language is a defined set of words constructed from symbols of a particular alphabet.

Note

A formal language can consist of a finite or infinite number of words and it can contain the empty word as well.

Examples

- $A := \{a, b\}$, $V^{*1} \ll A$. Then the

$$L := \{a, ab, abb, abbb, abbbb, \ldots\}$$

language is a formal language over A alphabet which contains an infinite number of words (a language containing words beginning with a and continuing with an arbitrary number of bs).

- $A := \{a, b\}$, $V^{*1} \ll A$. Then the

$$L := \{ab, ba, abab, baab, aabb, \ldots\}$$

is a formal language over alphabet A containing an infinite number of words (words in which there are as many as as bs).

Definition

Definition 20

If L_1, L_2 are two formal languages, then

$$L_1 * L_2 := \{\alpha\beta \,|\, \alpha \in L_1 \text{ and } \beta \in L_2\}.$$

This operation is called **contextual multiplication** of languages.

Contextual multiplication is distributive:

$$L_1 * (L_2 \cup L_3) = L_1 * L_2 \cup L_1 * L_3.$$

A formal language can be defined in many ways:

- With enumeration.
- We can give one or more attributes the words of the language all share, but other words do not.
- With the textual description of the rules for constructing words.
- With the mathematical definition of the rules for constructing words.
- With Generative Grammar.

Enumeration of the elements is not the most effective tool and is only possible in case of finite languages, but it is not always simple even with them.

$$L_1 = \{a, b, c, d, \ldots, z\},$$
$$L_2 = \{0, 1, 2, 3, 4, \ldots, 9\},$$
or
$$L_2 = a, ab, abc, \ldots.$$

Textual description could be a little bit better solution but it has the drawback of ambiguity and it is also very hard to create an algorithm or program based on it. To comprehend all this, let us have a look at the example:
Consider L_1 a language that includes integer numbers but only the ones that are greater than three...

Another example is when we define a language, with some attributes, using a mathematical formula.
When we define a language with some attributes, like in the following example.

- Consider $L_3 := L_1 * L_2$ a language (a language of odd numbers that include at least one even digit). This form contains textual definition which can be completely omitted in case of mathematical formulas.
- $L := \{0^n 1 0^n \mid n \geq 1\}$, namely there is a 1 in the middle of the number with the same amount of 0s before and after it.
- $L_4 = \{a^n b^n \mid n = 1, 2, 3, \ldots\}$

Exercises

- Raise the word "pear" to its 4th power.
- Give the number of subwords in word: "abcabcdef"
- Decide which word has a greater complexity, "ababcdeabc" or "1232312345".
- Give the Descartes product of the following two alphabets: $A = \{0, 1\}$ and $B = \{a, b\}$.
- Give the complex product of the two sets defined in the previous exercise.
- Define the set which contains even natural numbers.
- Give the textual definition of the following language: $L_1 = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.
- Give the mathematical definition of the language which consists of words with the length of 2 where the first