Language Operations and Regular Expressions

Last Updated: October 13th, 2025

1 Language Operations

Recall that a language is a set of words over some alphabet Σ. For this reason we may apply all the well-known set operations to languages, including union, intersection, complement, difference, and symmetric difference. With the exception of complement which takes only one language, each operation takes two languages, A and B, over some alphabet Σ, and from them produces a third language defined over Σ. In addition the **star** operation takes a language L and produces a new language that is the set of all possible finite concatenations of words from L. Let w be an arbitrary word over Σ. Then,

Set operations

Complement $w \in \overline{A}$ iff $w \notin A$

Union $w \in A \cup B$ iff either $w \in A$ or $w \in B$

Intersection $w \in A \cap B$ iff $w \in A$ and $w \in B$

Difference $w \in A - B$ iff $w \in A$ and $w \notin B$

Symmetric Difference $w \in A \oplus B$ iff $w \in A$ or $w \in B$, but not both

Concatenation $w \in A \circ B$ iff $w = uv$, where $u \in A$ and $v \in B$ $AB \equiv A \circ B$

Star $w \in L^*$ iff $w = \varepsilon$, or $w = u_1 u_2 \cdots u_n$, where each $u_i \in L$, $i = 1, 2, \ldots, n$.

$A = \{cat, dog, bird\}$, $B = \{walk, food, sound\}$

$AB = \{$ catwalk, catfood, catsound, dogwalk, ..., birdsound$\}$

dog | food $\in AB$? Yes

A B

Example 1.1. Let B_0 (respectively, B_1) denote the set of binary strings that begin with a 0 (respectively, 1). For each of the first five set operations listed above and when applied to B_0 and B_1 as arguments, describe the resulting set using a single sentence, and list up to five words that belong in the resulting language.

$B_0 \cup B_1 =$ All binary words except ε.

$B_0 \cap B_1 = \emptyset$

$\overline{B_0} = B_1 \cup \{\varepsilon\}$

$B_0 - B_1 = B_0$

$B_0 \oplus B_1 = B_1 \cup B_0 = B_0 \cup B_1$

(positive)

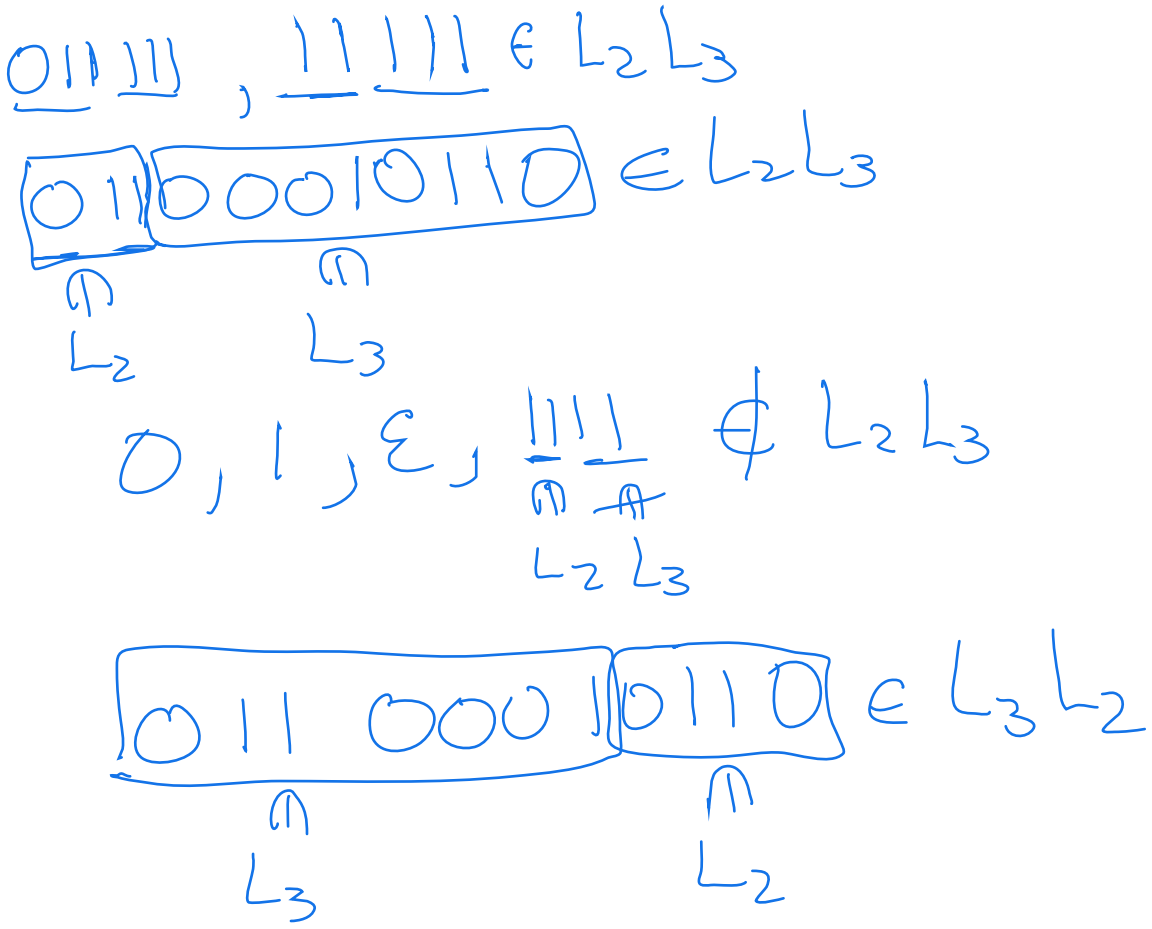
Example 1.2. Let L_2 (respectively, L_3) denote the set of binary strings whose 1-bits sum to a value that is divisible by two (respectively, three). Repeat the previous example with these two sets.

$L_2 \cup L_3$ = Binary words that either have a positive even # of 1's or a positive number of 1's that is a multiple of three.

$L_2 \cap L_3$ = Binary words with a positive # of 1's that is a multiple of 6.

Exercise: Describe L_2, $L_2 - L_3$, $L_2 \oplus L_3$.

Example 1.3. For L_2 and L_3 in Example 1.2, provide three words that are in L_2L_3, and three words that are not in L_2L_3. Explain why $L_2L_3 = L_3L_2$.



Example 1.4. We have the following star languages.

1. $\{0\}^* = \{\varepsilon, 0, 00, 000, \ldots\}$
2. $\{01, 10\}^* = \{\varepsilon, 01, 10, 0101, 0110, 1010, 1001, 010101, \ldots\}$. How many words of length n does this language have if n is even?

How many words of len $n = 8$ are there?

$2^4 = 16$ since each 2-block has two choices: either choose 01 or 10.

In general there are $2^{n/2}$ words of length n

Example 1.5. Consider the language L of all binary strings that begin with two 0's and have an odd number of 1's. Which of the following words are in L^*?

$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots$

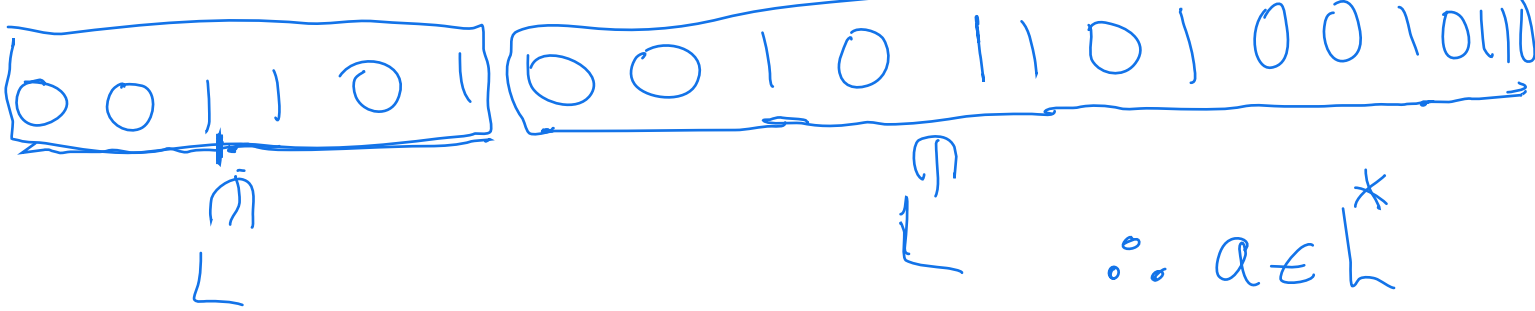
$\{\varepsilon\} \quad L \quad LOL \quad LOLOL$

a. 00110100101101001010

b. 00010110011001010101 $\in L^*$

c. 00101001101100001100

d. 001100100 $\in L^*$ since $d \in L \subseteq L^*$



L $\quad$ $\therefore a \in L^*$

$c =$ 0 0 1 0 1 0 0 1 1 0 1 1 0 0 0 0 1 1 0 0

$\notin L^*$

2 Regular Expressions

As the name suggests, a regular expression represents a way of providing a function expression (as opposed to a DFA or NFA) in order to represent a regular language. Regular expressions have many applications from desribing tokens in a programming language to providing search criteria for finding data in documents. For example, a line in a text document may be viewed as a single word over some alphabet. A regular expression then provides criteria for the lines that we wish to retrieve from the document.

Let Σ be an alphabet. We provide a structural definition for the set of all legal regular expressions defined over Σ.

2.1 Atomic Regular Expressions

1. For each letter $\mathtt{a} \in \Sigma$, $\mathtt{a}$ is a regular expression that represents the language $\{\mathtt{a}\}$.
2. ε is a regular expression that represents the language $\{\varepsilon\}$.
3. $\emptyset$ is a regular expression that represents the empty language $\emptyset$.

2.2 Compound Rules for Creating Regular Expressions

1. If R_1 and R_2 are regular expressions and $L(R_i)$ is the language represented by R_i, $i = 1, 2$, then $(R_1 \cup R_2)$ is a regular expression that represents the language $L(R_1) \cup L(R_2)$.
2. If R_1 and R_2 are regular expressions and $L(R_i)$ is the regular language represented by R_i, $i = 1, 2$, then $(R_1 \circ R_2)$ is a regular expression that represents the language $L(R_1) \circ L(R_2)$.
3. If R is a regular expression that represents the language $L(R)$ then (R^*) is also a regular expression that represents the language $L(R)^*$.

2.3 Rules for simplifying regular expression syntax

Dropping Parentheses Parentheses may be omitted when there is no ambiguity. For example, instead of writing (0^*), we may simply write 0^*.

Dropping $\circ$ Concatenation is more commonly expressed by placing side-by-side the two expressions to be concatenated. For example, instead of $0 \circ 1^*$, we may write 01^*. Note: $(01)^*$ would be used to star the expression 01.

Use of $\{\ \}$ Given symbols $a_1, a_2 \ldots, a_n \in \Sigma$, instead of writing $a_1 \cup a_2 \cup \cdots \cup a_n$, we may instead write $\{a_1, a_2, \ldots, a_n\}$, or even $(a_1, a_2, \ldots, a_n)$.

$+$ instead of $*$ L^* allows for the empty string ε. However, sometimes we want the word to have at least one character, in which case we may write L^+, which is defined as $L \circ L^*$. This forces the concatenation of words to have at least one word from L. For example, $\{0,1\}^+$ means all binary words with at least one bit.

Exponentiation For $k \geq 1$ an integer, we may write L^k in place of

$$\underbrace{L \cdots L}_{k \text{ times}}.$$

Example 2.1. Provide regular expressions for each of the following languages.

a. All binary words that begin with a 0 and end with a 1.

$0\{0,1\}^*1$

$aa\{a,b\}^*bb$

b. All binary words that have a length of at least three and its third bit is 0.

$\{0,1\}\ \{0,1\}\ 0\ \{0,1\}^*$

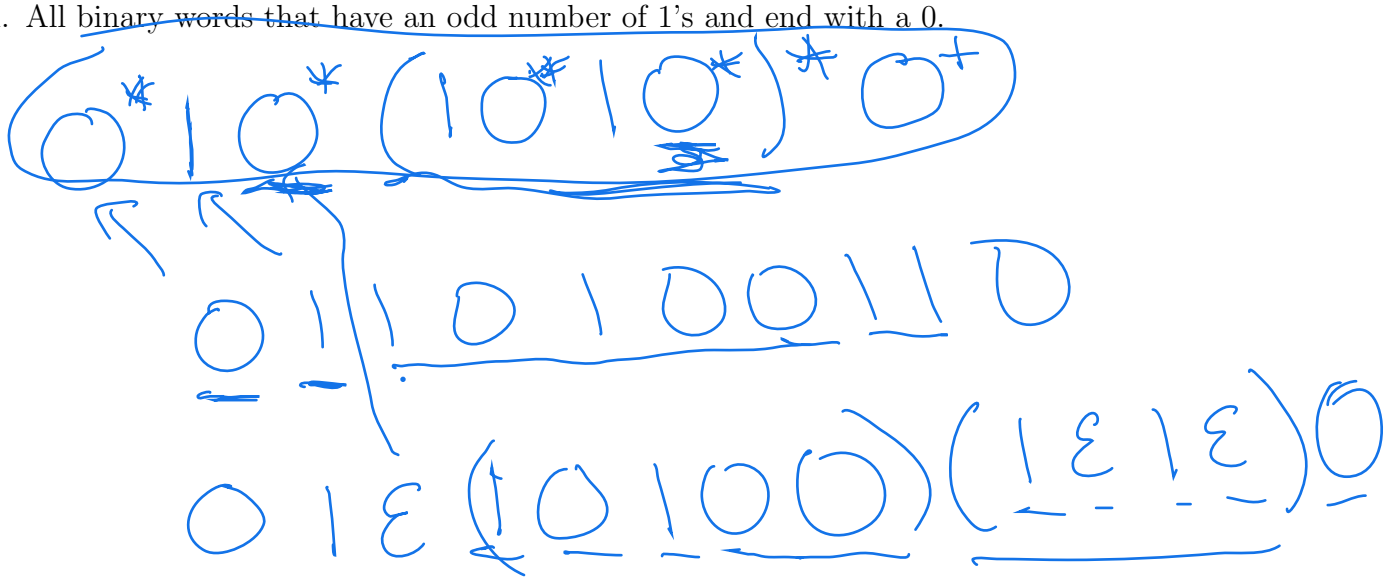
c. All binary words that have an even number of 0's.

$1^*(01^*01^*)^*$

$1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0$

$11(01101)(0101)$

d. All binary ~~words that~~ have an odd number of 1's and end with a 0.



e. The set of words w consisting of `a`'s and `b`'s, and does *not* have the form $\texttt{a}^*\texttt{b}^*$.

f. All words in {a,b}* except for aba.

Example 2.2. Provide the regular expresion that represents the language of all words w over the alphabet $\{-1, +1\}$ for which, for all $k = 1, \ldots, |w|$,

$$\left| \sum_{i=1}^{k} w_i \right| \leq 2.$$

Example 2.3. In programming, an identifier is a word that is used to identify some aspect of a program, such as a variable, constant, keyword, function name, the name of a data structure, and the names of its attributes. Consider the following rules for forming an identifier.

Rule 1. must begin with a letter

Rule 2. must end with either a letter or digit

Rule 3. must consist of alphanumeric characters (letters and digits) as well as the dash ("-") character, so long as the dash is immediately preceded (respectively, followed) by an alphanumeric character.

Assuming, an alphabet having letters $\{a, b\}$ and the single numerical digit 0, i) provide a regular expression that describes the set of all legal identifiers and ii) provide a DFA M for which $L(M)$ is the set of all legal identifiers.

Solution.

Example 2.4. Syntactically, an integer may be viewed as a sequence of one or more digits that (optionally) follows a negative symbol "-". We may use this definition to define a number represented in scientific notation. In particular, the number must

Rule 1. (optionally) begin with a negative symbol "-"

Rule 2. begin with (optionally, follwed by) a sequence of zero or more digits

Rule 3. followed by a period "."

Rule 4. followed by a sequence of zero or more digits

Rule 5. followed by an "e"

Rule 6. followed by an integer.

Rule 7. Rules 2 and 4 cannot both be satisfied by using an empty word.

Assuming, only the digits 0 and 1, provide a regular expression that describes the set of all numbers written in scientific notation.

Solution.

Regular Expression Core Exercises

1. Let L_1 denote the binary language whose words have an even number of 0's and an odd number of 1's. Let L_2 denote the binary language consisting of words whose length is divisible by three. For each of the following languages, describe using a complete sentence and provide five words that are in language.

   (a) $\overline{L}_1$

   (b) $L_1 \cup L_2$

   (c) $L_1 \cap L_2$

   (d) $L_1 - L_2$

   (e) $L_1 \oplus L_2$

2. Let L_1 denote the language whose words have exactly one 1 and an odd number of 0's, while L_2 denotes the language whose words have exactly two 1's and an even number of 0's. Provide an example of a word that is i) in both L_1L_2 and L_2L_1, and ii) in L_1L_2 but not in L_2L_1.

3. Which of the following is a member of L^*, where L is the language of binary words that contain at most two 0's and at least three 1's?

   (a) 10100011011101

   (b) ε

   (c) 01011110111000111

   (d) 110010101011100

4. Which of the following is a member of L^*, where L is the language $\{\texttt{aba}, \texttt{bbaa}\}$?

   (a) `abaababbaa`

   (b) `ababbaa`

   (c) `bbaababbaba`

   (d) `bbaabbaaabaababbaa`

5. Provide a regular expression that describes each of the following binary-word languages.

   (a) all words that begin with a 1 and end with a 0

   (b) all words that contain at least 3 1's

   (c) all words that contain 0101

   (d) all words having length at least 3 and whose third bit is a 0

   (e) all words that either start with a 1 and have odd length, or start with a 0 and have even length

   (f) all words that do *not* contain substring 110

   (g) all words having length at most 5

   (h) all words except for 11 and 111

(i) all words for which every odd bit is a 1

(j) all words that have at least two 0's and at most one 1

(k) the language consisting of ε and 0

(l) words that contain an even number of 0's or (inclusive) contain exactly two ones

(m) the language that has no words

(n) all words except the empty string

6. Provide the regular expression that represents the language of all words over the alphabet {a,b} for which one of the following is true: i) the word has at most one a or ii) the word has at least to a's and, for any two a's for which there are no other a's between them, there must be an odd number of b's between them. For example abbba and ababbbbba are two words that satisfy ii).

7. Provide the regular expresion that represents the language of all words w over the alphabet $\{-1, +1\}$ for which, for all $k = 1, \ldots, |w|$,

$$\left| \sum_{i=1}^{k} w_i \right| \leq 3.$$

Solutions to DFA Core Exercises

1. Let L_1 denote the binary language whose words have an even number of 0's and an odd number of 1's. Let L_2 denote the binary language consisting of words whose length is divisible by three. For each of the following languages, describe using a complete sentence and provide five words that are in language.

   (a) $\overline{L}_1$ is the set of all binary words that either have an odd number of 0's or (inclusive) have an even number of 1's. Examples include 0, 01, 10, 11, and 000.

   (b) $L_1 \cup L_2$ is the set of all binary words that either have an even number of 0's and an odd number of 1's or (inclusive) have a length that is divisible by three. Examples include 1, 00, 001, 010, and 100.

   (c) $L_1 \cap L_2$ is the set of all binary words that have an even number of 0's, an odd number of 1's, and a length that is divisible by three. Examples include, 001, 010, 100, 000000001, and 000000111.

   (d) $L_1 - L_2$ is the set of all binary words that have an even number of 0's, an odd number of 1's, and a length that is *not* divisible by three. Examples include 1, 00001, 00010, 00100, and 010000.

   (e) $L_1 \oplus L_2$ is the set of all binary words that either have an even number of 0's and an odd number of 1's or (exclusive) have a length that is divisible by three. Examples include 1, 00001, 00010, 00100, and 010000.

2. We have $01001001 \in L_1L_2 \cap L_2L_1$ since we may write it both as (01)(001001) and as (010010)(01). However, $0111 \in L_1L_2 - L_2L_1$ since (01)(11) is the only valid parsing.

3. We have the following results.

   (a) 10100011011101: no. There must be some prefix of this word that is a member of L. But 10100 cannot be a prefix of *any* word in L since it has three zeros and only two ones.

   (b) ε: yes. $\varepsilon \in L^*$ for any language L.

   (c) 01011110111000111: yes.

   $$01011110111000111 = 0101111 \cdot 01110 \cdot 00111,$$

   all three of which are words in L.

   (d) 110010101011100: no. If this word were in L^*, then it would have to be a concatenation of words from L, where the first word would have to be 11001, since the next bit is a zero, which would mean too many zeros. But this is followed by 01010 and no prefix of this word can be in L. Nor can this word be a prefix of some word in L.

4. We have the following.

   (a) `abaababbaa`: yes.

   (b) `ababbaa`: yes.

   (c) `bbaababbaba`: no. The first word in the concatenation would have to be `bbaa`. But then there is no word in L that begins with `ba`, which is what comes next.

(d) `bbaabbaaabaababbaa`: yes.

$$\texttt{bbaabbaaabaababbaa} = \texttt{bbaa} \cdot \texttt{bbaa} \cdot \texttt{aba} \cdot \texttt{aba} \cdot \texttt{bbaa}.$$

5. We have the following regular expressions.

(a) $1\{0,1\}^*0$

(b) $\{0,1\}^*1\{0,1\}^*1\{0,1\}^*1\{0,1\}^*$

(c) $\{0,1\}^*0101\{0,1\}^*$

(d) $\{0,1\}\{0,1\}0\{0,1\}^*$

(e) $1\{00,01,10,11\}^* \cup 0\{00,01,10,11\}^*\{0,1\}$

(f) Notice that once a word has two consecutive 1's, then all subsequent bits must equal 1. Thus, there are two cases: w contains only 0's or is empty, or w contains 1 or more isolated 1's, and ends with zero or more 1's. Therefore, we have

$$0^* \cup \{\varepsilon, 1\}(0^+1)^*0^*1^*$$

(g) $\bigcup_{i=0}^{5} \{0,1\}^i$, where, e.g., $\{0,1\}^3$ is shorthand for $\{0,1\}\{0,1\}\{0,1\}$ and $\{0,1\}^0 = \{\varepsilon\}$.

(h) We have

$$\{\varepsilon, 0, 1, 00, 01, 10, 000, 001, 010, 011, 100, 101, 110\} \cup \{0,1\}\{0,1\}\{0,1\}\{0,1\}\{0,1\}^*,$$

where the first expression (to the left of the union) includes all words, except 11 and 111, having length less than 4, while the second expression represents all binary words of length at least 4.

(i) There are three cases depending on whether w has even or odd length:

$$\{10,11\}^* \cup 1\{01,11\}^*.$$

(j) $000^* \cup 000^*10^* \cup 0^*1000^* \cup 00^*100^*$

(k) $\{\varepsilon, 0\}$ assuming the convention that the former is the same as $\varepsilon \cup 0$

(l) $(1^*01^*0)^*1^* \cup 0^*10^*10^*$

(m) $\emptyset$

(n) $\{0,1\}^+$

6. To Appear

7. To Appear