

Finite Representation of Languages: Regular Expressions

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#)



Finite Representation of Languages: Regular Expressions

Language Representations

Finite languages can be **represented** merely by listing them.

How, then, can we represent an infinite language? We can even ask "what does it mean to represent a language?"

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#)

Finite Representation of Languages: Regular Expressions

Language Representations

Consider the language $L = \Sigma^*$. This is clearly an infinite language that is countably infinite (a one-to-one correspondence exists with Σ^* as the domain and the integers as the co-domain).

The set of all possible languages over Σ is the power set of Σ^* , and this set is uncountably infinite.

This is big news! It tells us that, given a system of representation using a finite set of symbols, it will not be possible to represent every possible language.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#)

Finite Representation of Languages: Regular Expressions

Language Representations

Let us not be daunted by this, however. There are still plenty of languages that we can write representations for.

What exactly do we mean by a representation? For our purposes, a **representation of a language** is an expression formed using a set of symbols (possibly different from Σ) that describes the set of all strings in a language, and only those strings.

Consider the set $\{a\}^*$. This is the set of all strings that can be formed with the symbol a ; thus it is all strings with zero or more a 's. Since, viewed in this light, $\{a\}^*$ defines a set of strings, it also represents a language.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#)

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

Now we will develop a powerful system of language representation, one that, while it can be used to represent only a limited number of languages, is very useful nonetheless and plays an important role in computer science theory.

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

The following are examples of regular expressions over the alphabet $\Sigma = \{a, b, c\}$:

- a
- $(a+b)^*$
- $a^*(bc+c)b^*$
- $(a+\Lambda)b^*$
- $(a(b+c))^*$

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

A **regular expression** over an alphabet Σ is a string over the alphabet

$$\Sigma \cup \{ (,), \emptyset, +, * \}$$

and is defined inductively as

1. \emptyset and each symbol in Σ are regular expressions.
2. If \mathbf{u} and \mathbf{v} are regular expressions then so is (\mathbf{uv}) (concatenation).
3. If \mathbf{u} and \mathbf{v} are regular expressions then so is $(\mathbf{u+v})$.
4. If \mathbf{u} is a regular expression then so is \mathbf{u}^* .
5. Nothing else is a regular expression.

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

Theorem: Let \mathbf{r} , \mathbf{s} , and \mathbf{t} be regular expressions over the same alphabet Σ . Then:

1. $\mathbf{r + s = s + r}$
2. $\mathbf{r + \Lambda = \Lambda + r = r}$
3. $\mathbf{r + r = r}$
4. $\mathbf{(r + s) + t = r + (s + t)}$
5. $\mathbf{r \Lambda = \Lambda r = r}$

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

Theorem (continued)

6. $(rs)t = r(st)$
7. $r(s + t) = rs + rt$ and $(r + s)t = rt + st$
8. $r^* = r^{**} = r^*r^* = (\Lambda + r)^* = r^*(r + \Lambda) = (r + \Lambda)r^* = \Lambda + rr^*$
9. $(r + s)^* = (r^* + s^*)^* = (r^*s^*)^* = (r^*s)^*r^* = r^*(sr^*)^*$
10. $r(sr)^* = (rs)^*r$

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Expressions

Theorem (continued)

11. $(r^*s)^* = \Lambda + (r + s)^*s$
12. $(rs^*)^* = \Lambda + r(r + s)^*$
13. $s(r + \Lambda)^* (r + \Lambda) + s = sr^*$
14. $rr^* = r^*r$

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Languages

If \mathbf{u} is a regular expression, then $L(\mathbf{u})$ is the language represented the regular expression \mathbf{u} . Thus L is a function whose domain is the set of all regular expressions and whose codomain is the set of languages over an alphabet Σ .

The following properties are consequences of the definitions we've just seen:

- $L(\emptyset) = \emptyset$ and $L(a) = \{a\}$ for each a in Σ .
- If \mathbf{u} and \mathbf{v} are regular expressions then $L(\mathbf{uv}) = L(\mathbf{u})L(\mathbf{v})$.
- If \mathbf{u} and \mathbf{v} are regular expressions then $L(\mathbf{u+v}) = L(\mathbf{u}) \cup L(\mathbf{v})$.
- If \mathbf{u} is a regular expression then $L(\mathbf{u}^*) = L(\mathbf{u})^*$.

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Languages

Any language that can be represented by a regular expression is a **regular language**.

Assume $\Sigma = \{a, b\}$. What is a regular expression that describes the regular language

$$L = \{x \in \Sigma^* \mid x \text{ contains } aba \text{ or } bab \text{ as a substring}\}$$

Here is a [possible solution](#).

1 2 3 4 5 6 7 8 9 10 11 12 13

Finite Representation of Languages: Regular Expressions

[prev](#) | [slides](#) | [next](#)

Regular Languages

Theorem: All finite languages are regular.

Proof: Consider a finite language L over the alphabet Σ . Every string in L can be formed using the concatenation operation on individual symbols from Σ and possibly Λ .

Since Λ and the symbols in Σ are regular expressions, and since the concatenations of regular expressions are regular expressions, we conclude that each string in L is a regular expression.

If we form the union of all of the strings we obtain a regular expression that represents L , thus L is a regular language.

1 2 3 4 5 6 7 8 9 10 11 12 13