

REGULAR LANGUAGES AND FINITE AUTOMATA (1)

Jie Jiang

October 12, 2009



Languages definition

- ▶ Σ^* denotes the set of all sequences of strings that are composed of zero or more symbols of Σ .
- ▶ A *language* is a subset of Σ^* .
- ▶ A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols.
 - ▶ Σ^* , \emptyset and Σ are languages.
 - ▶ $\{aba, czr, d, f\}$ is a language over $\{a, b, \dots, z\}$.
- ▶ The infinite languages are specified by the scheme:

$$L = \{w \in \Sigma^* : w \text{ has property } P\}$$

- ▶ $\{w \in \{0, 1\}^* : w \text{ has an equal number of 0's and 1's}\}$
- ▶ $\{w \in \Sigma^* : w = w^R\}$



Finite representation of languages

- ▶ How to represent languages by finite specifications? (A central issue in the theory of computation)
- ▶ Finite language: exhaustive enumeration of all the strings in the language.
 - ▶ $\{aba, czr, d, f\}$ is a language over $\{a, b, \dots, z\}$.
- ▶ What about infinite languages?
 - ▶ $\{w \in \{0, 1\}^* : w \text{ has an equal number of 0's and 1's}\}$
 - ▶ $\{w \in \Sigma^* : w = w^R\}$



Finite representation of languages

- ▶ What's "finite representation of a language."
 - ▶ any such representation must be a *string*, a finite sequence of symbols over some alphabet Σ
 - ▶ different languages should have different representations.
- ▶ Problems:
- ▶ Only a countable number of representations
 - ▶ the set Σ^* of strings over an alphabet Σ is countably infinite
 - ▶ The union of a countably infinite collection of countably infinite sets is countably infinite.
- ▶ An uncountable number of things to represent
 - ▶ the set of all possible languages over a given alphabet Σ is uncountably infinite.
 - ▶ power set of any countably infinite set is not countably infinite.
- ▶ Find finite representations, of one sort or another, for at least some of the more interesting languages.



Finite representation of languages

- ▶ No matter how powerful are the methods we use for representing languages, only countably many languages can be represented, so long as the representations themselves are finite.
- ▶ There being uncountably many languages in all, the vast majority of them will inevitably be missed under any finite representational scheme.
- ▶ Several ways of describing and representing languages:
 - ▶ in Chomsky hierarchy
 - ▶ all these finite representational methods are inevitably limited
- ▶ Regular languages, regular grammars and finite automata



Regular expressions

- ▶ Example for expressions (strings of symbols)
 - ▶ to describe how languages can be built up by using the operations described in the previous section.
- ▶ $L = \{w \in \{0, 1\}^* : w \text{ has two or three occurrences of } 1, \text{ the first and second of which are not consecutive}\}.$
- ▶ This language can be described using only singleton sets and the symbols \cup , \circ , and $*$ as

$$\{0\}^* \circ \{1\} \circ \{0\}^* \circ \{0\} \circ \{1\} \circ \{0\}^* \circ ((\{1\} \circ \{0\}^*) \cup \emptyset^*)$$
- ▶ The only symbols used are the braces $\{$ and $\}$, the parentheses $($ and $)$, \emptyset , 0 , 1 , $*$, \circ , and \cup .
- ▶ dispense with the braces and \circ :

$$L = 0^*10^*010^*(10^* \cup \emptyset^*)$$



Regular expressions

- ▶ Roughly speaking, a regular expression describes a language exclusively by means of single symbols and \emptyset , combined perhaps with the symbols \cup and $*$, possibly with the aid of parentheses.
- ▶ Definition: **regular expression**
 - ▶ Basis: \emptyset , ϵ , and a are regular expressions for all $a \in \Sigma$.
 - ▶ Induction: If R and S are regular expressions, then the following expressions are also regular: (R) , $R|S$, RS , and R^* .
- ▶ Kleene star (or Kleene operator or Kleene closure): R^*
- ▶ Kleene plus : R^+
- ▶ Note: different from the notations above (for sets etc.), for example: replace \cup with $|$, ...

Regular expressions

- ▶ **Order of precedence: parentheses > Kleene closure > concatenation > alternation.**
- ▶ Some properties:
 - ▶ $R^* = R^*R^* = (R^*)^* = R|R^*$.
 - ▶ $R(SR)^* = (RS)^*R$.
 - ▶ $(R^*S)^* = \epsilon|(R|S)^*S$.
 - ▶ $(RS^*)^* = \epsilon|R(R|S)^*$.



Regular languages

- ▶ Every regular expression represents a regular language, and every regular language is represented by a regular expression.
- ▶ Definition: **regular language**
 - ▶ Basis: \emptyset , $\{\epsilon\}$, and $\{a\}$ are regular languages for all $a \in \Sigma$.
 - ▶ Induction: If L and M are regular languages, then the following languages are also regular: $L \cup M$, LM and L^* .

Regular languages

- ▶ Given regular expression R , $L(R)$ stands for the language represented by R .
 - ▶ the relation between regular expressions and their corresponding languages is established by a function L , which is a function from strings to languages.
- ▶ relations between regular expressions and regular languages by definition:
 - ▶ basis: $L(\emptyset) = \emptyset$, $L(\epsilon) = \{\epsilon\}$, $L(a) = \{a\}$ for each $a \in \Sigma$
 - ▶ induction: $L(RS) = L(R)L(S)$, $L(R|S) = L(R) \cup L(S)$, $L(R^*) = L(R)^*$
- ▶ The class of **regular languages** over an alphabet Σ is defined to consist of all languages L such that $L = L(a)$ for some regular expression a over Σ . That is, regular languages are all languages that can be described by regular expressions.



Examples

- ▶ Let $\Sigma = \{a, b\}$
 - ▶ $a|b$ denotes $\{a, b\}$
 - ▶ $(a|b)(a|b)$ denotes $\{aa, ab, ba, bb\}$
i.e., $(a|b)(a|b) = aa|ab|ba|bb$
 - ▶ a^* denotes $\{\epsilon, a, aa, aaa, \dots\}$
 - ▶ $(a|b)^*$ denotes the set of all strings of a 's and b 's (including ϵ)
i.e., $(a|b)^* = (a^*b^*)^*$
 - ▶ $a|a^*b$ denotes $\{a, b, ab, aab, aaab, \dots\}$

Examples

- ▶ Regular expressions are an inadequate specification method in general.
- ▶ $\{0^n 1^n : n \geq 0\}$ is not regular.

Language recognition device

- ▶ An algorithm that is specifically designed, for some language L , to answer questions of the form *Is string w a member of L ?* will be called a **language recognition device**.
- ▶ For example, a device for recognizing the language:
$$L = \{w \in \{0, 1\}^* : w \text{ does not have } 111 \text{ as a substring}\}$$
- ▶ by reading strings, a symbol at a time, from left to right, it might operate like this:
 - ▶ Keep a count, which starts at zero and is set back to zero every time a 0 is encountered in the input;
 - ▶ add one every time a 1 is encountered in the input;
 - ▶ stop with a No answer if the count ever reaches three
 - ▶ stop with a Yes answer if the whole string is read without the count reaching three.



Language generator

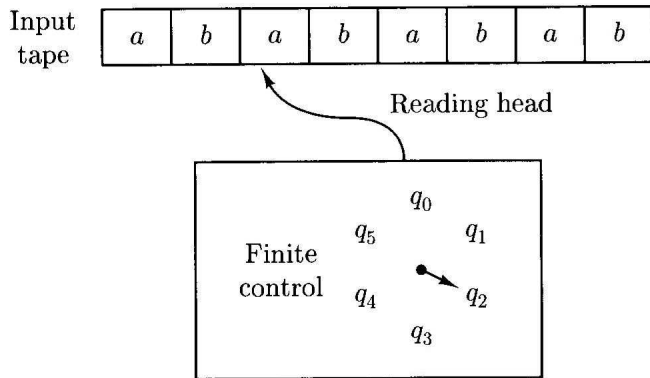
- ▶ To describe how a generic specimen in the language is produced.
- ▶ For example, regular expression such as $(\epsilon|b|bb)(a|ab|abb)^*$ maybe viewed as a way of generating members of a language:
 - ▶ To produce a member of L , first write down either nothing, or b , or bb
 - ▶ then write down a or ab , or abb
 - ▶ and do this any number of times, including zero
 - ▶ all and only members of L can be produced in this way.
- ▶ Such language generators are not algorithms, but they are important and useful means of representing languages all the same.



Finite automata

- ▶ Finite automaton (finite state machine)
 - ▶ No *stored program* concept – the machine is the computation.
 - ▶ No auxiliary memory – the automaton fixes memory by its definition.
 - ▶ Input: a string on a *tape*
 - ▶ Read head moves over string (left to right).
 - ▶ Output: *Accept* or *Not Accept*

Finite automata





Finite automata

- ▶ **input tape**
 - ▶ made of squares, one symbol per square
- ▶ **reading head**
 - ▶ After reading an input symbol, moves one square to the right
- ▶ **finite control**
 - ▶ CPU is a finite collection of **states**
 - ▶ **initial state**
 - ▶ a set of **final states**



Finite automata

- ▶ CPU is a finite collection of **states**
 - ▶ **initial state**, a set of **final states**
 - ▶ At any instant, finite state automaton is *in* some state
 - ▶ As symbols are read, finite state automaton may change to another state
 - ▶ **deterministic** finite automaton : new state depends only on the current state and the symbol just read
 - ▶ **nondeterministic** finite automaton
- ▶ repeat till the reading head reaches the end of the input string
 - ▶ **accepted** : it winds up in one of a set of final states
 - ▶ The language accepted by the machine is the set of strings it accepts



Finite automata

- ▶ A severely restricted model of an actual computer
 - ▶ complete absence of memory outside its fixed central processor
 - ▶ has a memory capacity that is fixed *at the factory* and cannot thereafter be expanded
- ▶ Why do we use finite automaton:
 - ▶ first be sure that the theory computers with limited memory is well understood
 - ▶ could be used to design several common types of computer algorithms and programs
 - ▶ lexical analysis phase of a compiler
 - ▶ the problem of finding an occurrence of a string within another string
- ▶ **Kleene's theorem:** A language is regular **iff** it is accepted by a finite state automaton.

Exercise 1

Try to write down the regular expressions that represent the following regular languages:

- ▶ $\{0, 1\}^*$
 - ▶ $(0|1)^*$
- ▶ $\{0, 1\}^+$
 - ▶ $(0|1)^+$
- ▶ $\{w \mid w \in \{0, 1\}^+ \text{ and } w \text{ has the substring } 10110\}$
 - ▶ $(0|1)^*10110(0|1)^*$
- ▶ $\{w \mid w \in \{0, 1\}^+ \text{ and } w(10) = 1\}$
 - ▶ $(0|1)^91(0|1)^*$

Exercise 1

- ▶ $\{w \mid w \in \{0, 1\}^+ \text{ and } w \text{ starts with } 0, \text{ ends with } 1\}$
 - ▶ $0(0|1)^*1$
- ▶ $\{w \mid w \in \{0, 1\}^+ \text{ and if } w \text{ has at least two symbols of } 1\}$
 - ▶ $(0|1)^*1(0|1)^*1(0|1)^*$
- ▶ $\{w \mid w \in \{0, 1\}^* \text{ and if } w \text{ ends with } 1, \text{ then its length is an even number; if } w \text{ ends with } 0, \text{ then its length is an odd number}\}$
 - ▶ $(0|1)^{2n+1}1|(0|1)^{2n}0 \ (n \in \mathbb{N})$



Exercise 2

Try to explain the regular languages represented by the given regular expressions:

- ▶ $(00|11)^+$
 - ▶ $\{w | w \in \{0, 1\}^* \text{ and } w \text{ consists of double 0 and double 1}\}$
- ▶ $(0|1)^*0100^*$
 - ▶ $\{w | w \in \{0, 1\}^* \text{ and } w \text{ ends with } 010 \text{ plus consecutive 0s}\}$
- ▶ $(1|01|001)^*(\epsilon|0|00)$
 - ▶ $\{w | w \in \{0, 1\}^* \text{ and } w \text{ does not have three consecutive 0s}\}$
- ▶ $((0|1)(0|1))^* | ((0|1)(0|1)(0|1))^*$
 - ▶ $\{w | w \in \{0, 1\}^* \text{ and the length of } w \text{ is } 3n \text{ or } 2m \text{ (} n \in \mathbb{N}, m \in \mathbb{N}\text{)}\}$
- ▶ $((0|1)(0|1))^* ((0|1)(0|1)(0|1))^*$
 - ▶ $\{w | w \in \{0, 1\}^* \text{ and the length of } w \text{ is } 3n + 2m \text{ (} n \in \mathbb{N}, m \in \mathbb{N}\text{)}\}$

Regular expressions and languages

Finite representations

Regular expression

Regular languages

Example

Examples of regular expression

Finite Automata

Language recognizer and generator

Finite Automata

Exercise

Exercises for regular expression and language