Chapter 3

Regular Expressions, Regular Languages, Nondeterminism, and Kleene's Theorem

Regular Languages and Regular Expressions

- Many simple languages can be expressed by a formula involving languages containing a single string of length 1 and the operations of union, concatenation and Kleene star. Here are three examples
  - Strings ending in *aa*: $\{a, b\}^* \{aa\}$
  - (This is a simplification of $(\{a\} \cup \{b\})^*\{a\}\{a\}$)
  - Strings containing *ab* or *bba*: $\{a, b\}^* \{ab, bba\} \{a, b\}^*$
- These are called *regular* languages

Regular Languages and Regular Expressions (cont'd.)

- Definition: If Σ is an alphabet**, the set *R* of regular languages** over Σ is defined as follows:
  - The language $\varnothing$ is an element of ***R***, and for every $\sigma \in \Sigma$, the language $\{\sigma\}$ is in ***R***
  - For every two languages L_1 and L_2 in ***R***, the three languages $L_1 \cup L_2$, L_1L_2, and $L_1{}^*$ are elements of ***R***
- Examples:
  - $\{\Lambda\}$, because $\varnothing^* = \{\Lambda\}$
  - $\{a, b\}^*\{aa\} = (\{a\} \cup \{b\})^*(\{a\}\{a\})$

2. Then with these

3. Kleene's star

4. Last concatenation

1. We start with these

Regular Languages and Regular Expressions (cont'd.)

- A *regular expression* for a language is a slightly more user-friendly formula which is similar to algebraic expressions
  - Parentheses replace curly braces, and are used only when needed, and the union symbol is replaced by +

Regular language	***Regular Expression***
$\varnothing$	$\varnothing$
$\{\Lambda\}$	Λ
$\{a,b\}^*$	$(a+b)^*$
$\{aab\}^*\{a,ab\}$	$(aab)^*(a+ab)$
$(\{aa, bb\} \cup \{ab, ba\}\{aa, bb\}^*\{ab, ba\})^*$	$(aa + bb + (ab + ba)(aa + bb)^*(ab + ba))^*$

Regular Languages and Regular Expressions (cont'd.)

- A regular expression describes a regular language, and a regular language can be described by a regular expression.
- Two regular expressions are equal if the languages they describe are equal. For example,
  - $(a^*b^*)^* = (a+b)^*$
  - $(a+b)^*ab(a+b)^*+b^*a^* = (a+b)^*$
    - The first half of the left-hand expression describes the strings that contain the substring *ab* and the second half describes those that don't

Regular Languages and Regular Expressions (cont'd.)

- The language in $\{a, b\}^*$ with an odd number of a's
- A string with an odd number of *a*'s has at least one *a*, and the additional *a*'s can be grouped into pairs. There can be arbitrarily many *b*'s before the first *a*, between any two consecutive *a*'s, and after the last *a*.

  - $b^*ab^*(ab^*ab^*)^*$
  - $b^*a(b^*ab^*a)^*b^*$
  - $b^*a(b+ab^*a)^*$
  - $(b+ab^*a)^*ab^*$

See more examples in the textbook!

Regular Languages and Regular Expressions (cont'd.)

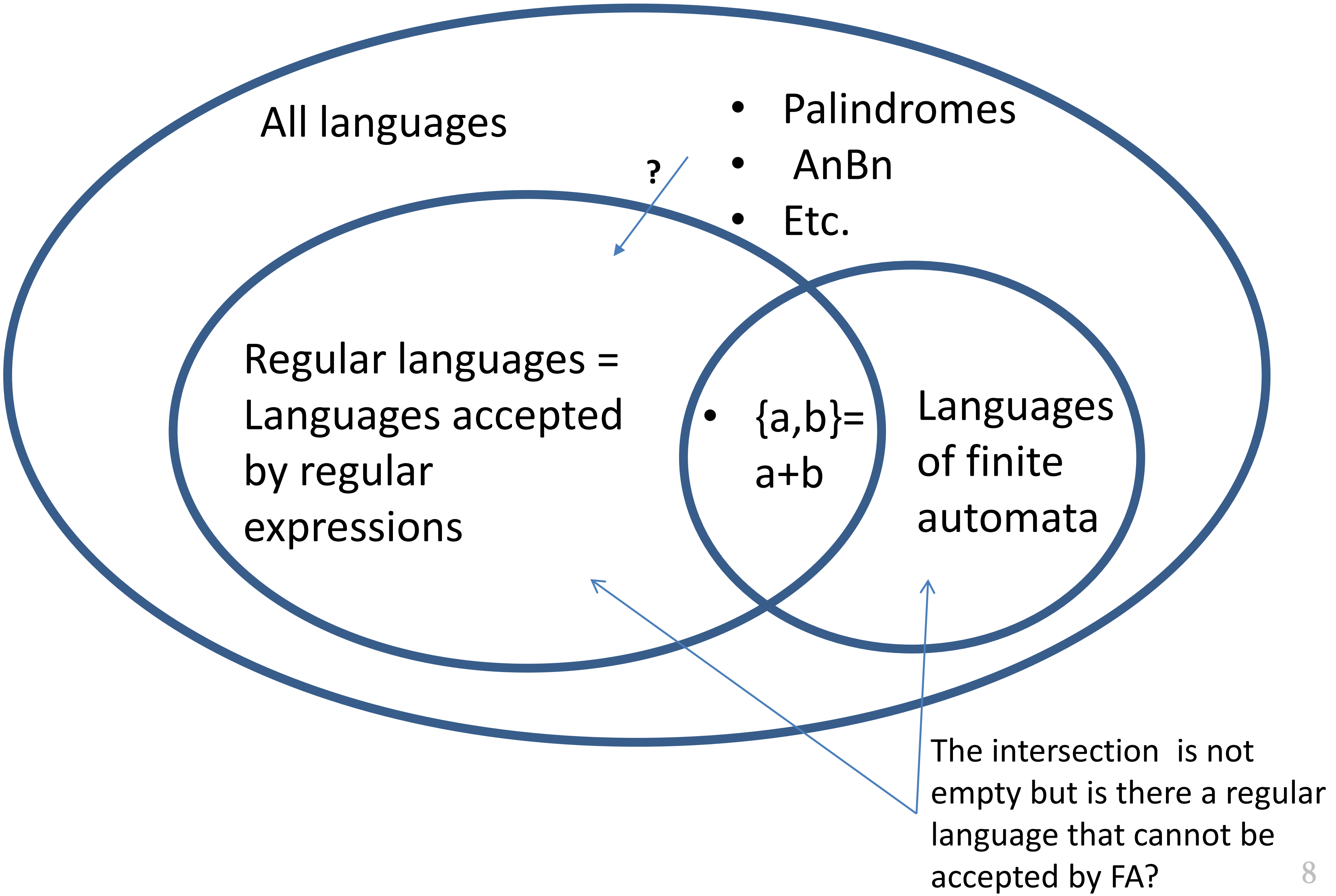
- An identifier in C is a string of length 1 or more that contains only letters, digits, and underscores ("_") and does not begin with a digit.

digit

$(l+\_)(l+d+\_)^*$

Letter, i.e., *a+b+c+…+A+B+…+Z*

This is what we know about languages …



For the alphabet {0, 1} find regular expressions for languages

- All binary strings

  (0+1)* = (1+0)*

- All binary strings of even length

  ((0+1)(0+1))*

- All binary strings containing the substring 001

  (0+1)*001(0+1)*

- All binary strings with #1s = 0 mod 3

  0* + (0*10*10*10*)*

- All binary strings without two consecutive 0s

  (01+1)*(0+Λ)

- All binary strings with either 001 or 100 occurring somewhere

  (0+1)*001(0+1)* + (0+1)*100(0+1)*