San José State University
Department of Computer Science

Ahmad Yazdankhah
ahmad.yazdankhah@sjsu.edu
www.cs.sjsu.edu/~yazdankhah

Regular Expressions

(Part 2)

Lecture 20-1
Day 22/31

CS 154
Formal Languages and Computability
Fall 2019

Agenda of Day 22

- Collecting Quiz ++
- Summary of Lecture 19
- Lecture 20: Teaching ...
  - Regular Expressions (Part 2)
  - Grammars (Part 1)

Summary of Lecture 19: We learned ...

Regular Expressions (REGEXs)

- REGEXs are another way to represent formal languages.
- We like REGEXs because ...
  - ... they represent formal languages in a more compact way.
  - They are shorthand for some formal languages.
  - They are easier to be implemented in computer.
  - They have practical applications in OS's and programming languages.
- This course introduces the mathematical base of them.

- The elements of REGEXs are:
  - Σ , ϕ , λ
  - ( )
  - Operators:

    + (union)

    . (dot or concatenation)

    * (star-closure)

Any Question?

Summary of Lecture 19: We learned ...

REGEXs

- We defined REGEXs formally as:

1. ϕ, λ, and $a \in \Sigma$ are all REGEXs.
2. If r_1 and r_2 are REGEXs, then the following expressions are REGEXs too:

   $r_1 + r_2$

   $r_1 . r_2$

   r_1^*

   (r_1)
3. A string is REGEX if it can be derived recursively from the primitive REGEXs by a finite number of applications of the above rules.

- Between REGEXs and languages, there are the following correspondence:

1. $L(\phi) = \{\ \}$
2. $L(\lambda) = \{\lambda\}$
3. $L(a) = \{a\}$ for all $a \in \Sigma$
4. $L(r_1 + r_2) = L(r_1) \cup L(r_2)$
5. $L(r_1 . r_2) = L(r_1)\, L(r_2)$
6. $L((r_1)) = L(r_1)$
7. $L(r_1^*) = (L(r_1))^*$

- We learned how to calculate the language represented by a REGEX by using the above correspondences.

Any Question?

Summary of Lecture 19: We learned ...

REGEXs

- Two regular expressions r_1 and r_2 are equivalent if ...
- ... both represent the same language.

$$L(r_1) = L(r_2) \rightarrow r_1 \equiv r_2$$

Identities

- If r, s, and t are REGEXs, and a, b $\in \Sigma$, then:

1. $r(s + t) = rs + rt$
2. $(s + t)r = sr + tr$
3. $(a^*)^* = a^*$
4. $(a \ldots a)^* a = a (a \ldots a)^*$
5. $a^* (a + b)^* = (a + b)^* a^* = (a + b)^*$

- Associated language to a REGEX is ...
  - ... the language that it represents.

Any Question?

Language → REGEX Examples

Language → REGEX

Example 24

- Given $L(r) = \{w \in \Sigma^* : w \text{ has exactly one a}\}$ over $\Sigma = \{a, b\}$

  $r = ?$

Solution

Language → REGEX

Example 25

- Given $L(r) = \{w \in \Sigma^* : w \text{ has at least one pair of consecutive a's}\}$ over $\Sigma = \{a, b\}$

  r = ?

Solution

Language → REGEX

Example 26

- Given $L(r) = \{a^n b^m : n \geq 3, m \text{ is even}\}$ over $\Sigma = \{a, b\}$

  $r = ?$

Solution

Language → REGEX

Example 27

- Given L(r) = {w : |w| ≥ 3, 3rd symbol of w is 'a'} over Σ = {a, b}

  r = ?

Solution

Language → REGEX

Example 28

- Given $L(r) = \{a^n b^m : n + m \text{ is even}\}$ over $\Sigma = \{a, b\}$

  $r = ?$

Solution

Homework

- Find a REGEX for the following languages.
  1. $L(r) = \{w \in \{a, b\}^* : w \text{ contains no a}\}$
  2. $L(r) = \{w \in \{a, b\}^* : w \text{ contains exactly two a's}\}$
  3. $L(r) = \{a^{2n} : n \geq 0\}$ over $\Sigma = \{a\}$
  4. $L(r) = \{a^{2n+1} : n \geq 0\}$ over $\Sigma = \{a\}$
  5. $L(r) = \{w \in \{a, b\}^* : w \text{ contains at least two a's}\}$
  6. $L(r) = \{w \in \{a, b\}^* : w \text{ begins with an 'a' and ends with a 'b'}\}$
  7. $L(r) = \{w \in \{a, b\}^* : w \text{ begins and ends with the same symbol}\}$
  8. $L(r) = \{w \in \{a, b\}^* : w \text{ contains exactly one occurrence of aa}\}$

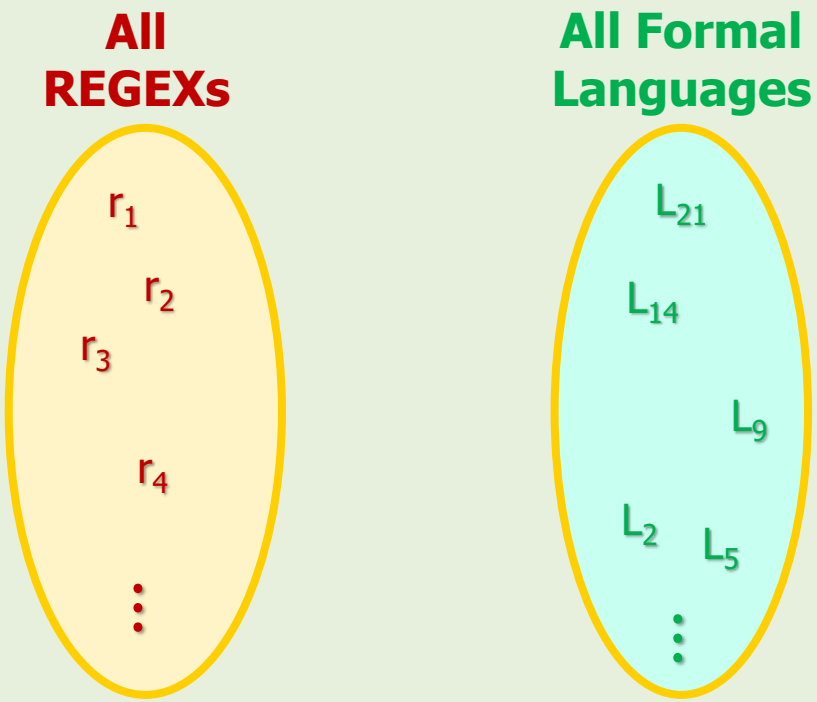
REGEXs and Languages Association

REGEXs and Languages Association

- What is the relationship between:

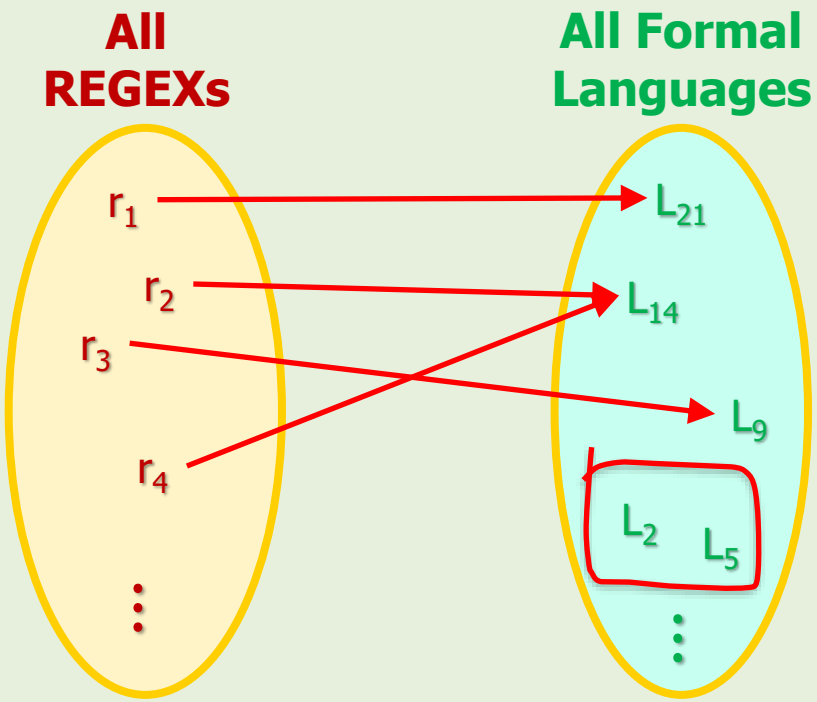
  the set of REGEXs, and

  the set of all formal languages?



REGEXs and Languages Association

- You agree that "every REGEX represents a language".
- BUT we don't know yet whether we can represent every language by a REGEX or not!
  - Our knowledge is not enough yet.



REGEX for More Complex Languages

- Find a REGEX for each of the following languages:

1. $L = \{a^n b^n : n \geq 0\}$ over $\Sigma = \{a, b\}$
2. $L = \{ww^R : w \in \Sigma^*\}$ over $\Sigma = \{a, b\}$

Solution

- Struggling?!

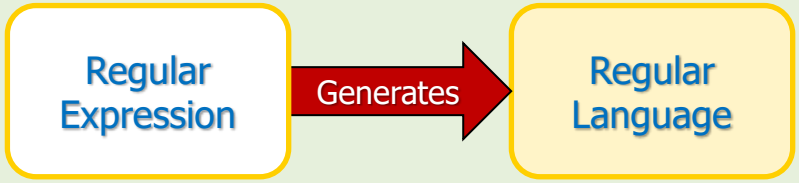
- After some struggling, you realize that you cannot find any REGEX for these languages! Why?

- Look at the theorems in the next slide!

REGEXs and Regular Languages

Theorem

- If r is a REGEX, then L(r) is a regular language over Σ.



Theorem

- Let L be a regular language over Σ.
  Then there exists a REGEX r such that L = L(r).

REGEXs and Regular Languages

- The following definition shows that REGEXs are another way to represent regular languages.

Definition

- A language is regular if a REGEX represents it.

Set of Languages Represented by REGEXs = Set of Regular Languages

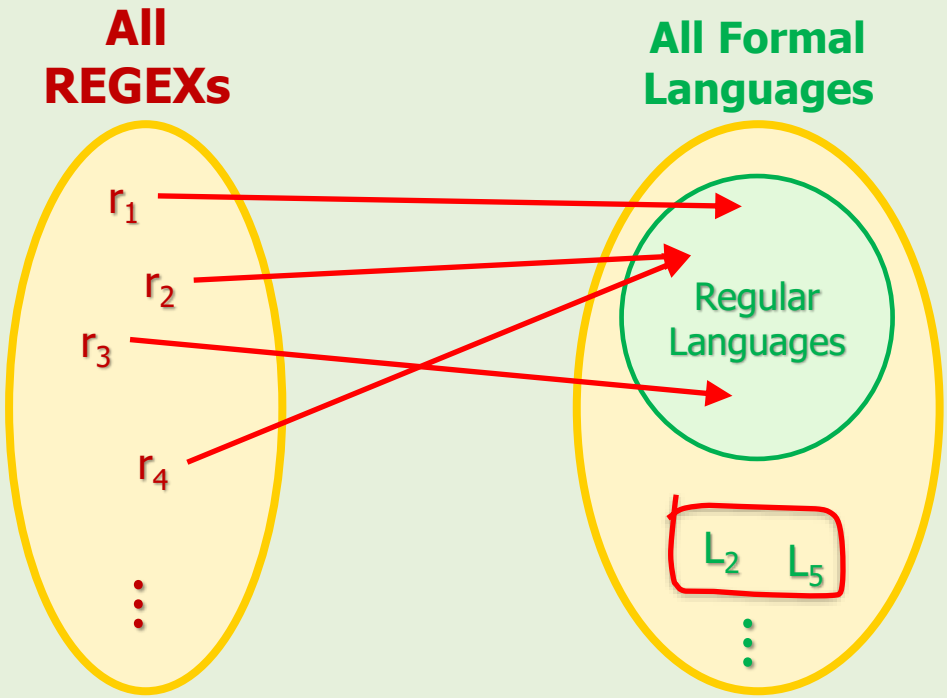
REGEXs and Languages Association

Revisited

- We've already agreed that "every REGEX represents a language".
- Now we know that:

  Those languages are regular.

  And there is no association between non-regular Languages and REGEXs.



What is the Next Step?

- We started this topic to look for a compact way to represent formal languages.
- We introduced REGEXs and experienced their usefulness.
- But the theorems showed their limitations.
  - REGEXs represent only regular languages.

- So, the next step would be looking for ...
  a practical compact way to represent non-regular languages.

Last Question: Do We Have a Standard REGEX?

- In computer science, we do NOT have a standard REGEX!
- Every OS and every programming language has its own REGEX.

- Of course, there are some common elements and rules between all of them.
  - So, you should learn each one based on their elements and rules.

- But the basic idea is the same.
  - In fact, they have implemented their REGEXs based on the REGEX we introduced here.

Homework

- Fill out the following tables.

+	ϕ	λ	a
ϕ			a
λ			
a	a		a

Examples

$\phi + a = \phi + a = a$

$a \,.\, a = aa$

.	ϕ	λ	a
ϕ			
λ			
a			aa

- Note that '+' and '.' are binary operators and need two operands but '*' is unary operator and needs one operand.

ϕ*	
λ*	
a*	a*

References

1. Linz, Peter, "An Introduction to Formal Languages and Automata, 5th ed.," Jones & Bartlett Learning, LLC, Canada, 2012
2. Michael Sipser, "Introduction to the Theory of Computation, 3rd ed.," CENGAGE Learning, United States, 2013
   ISBN-13: 978-1133187790