



## Chapter 9: Regular Languages

A **regular language** is a language that can be defined by a regular expression.

We study some properties of the class of regular languages.

Dr. Nejib Zagula CSI3104-W11

1



**Example: Every finite language is regular**

**Theorem:** Let  $L_1$  and  $L_2$  be two regular languages. The languages  $L_1+L_2$ ,  $L_1L_2$ , and  $L_1^*$  are regular languages.

**Proof 1:** If  $L_1$  and  $L_2$  are regular languages, then there exist regular expressions  $r_1$  and  $r_2$  that define them.

- The language associated with  $r_1+r_2$  is  $L_1+L_2$ .
- The language associated with  $r_1r_2$  is  $L_1L_2$ .
- The language associated with  $r_1^*$  is  $L_1^*$ .

$L_1+L_2$ ,  $L_1L_2$ , and  $L_1^*$  can be defined by regular expressions  
Thus, by definition, they are regular languages.

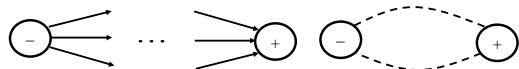
Dr. Nejib Zagula CSI3104-W11

2



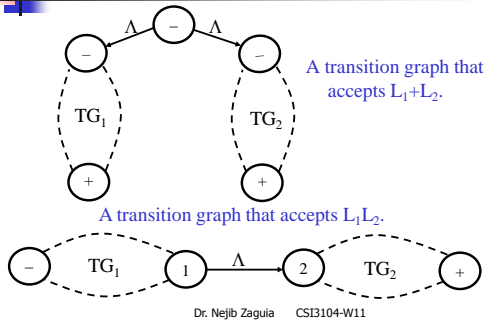
**Proof 2: using Kleene's theorem**

Because  $L_1$  and  $L_2$  are regular languages, there are regular expressions  $r_1$  et  $r_2$  that define them. By Kleene's theorem, there are transition graphs that accept them. We can transform these transition graphs into transition graphs with one start state and one final state. Let  $TG_1$  and  $TG_2$  be two transition graphs of this form that accept  $L_1$  et  $L_2$ .



Dr. Nejib Zagula CSI3104-W11

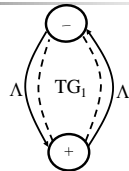
3



Dr. Nejib Zagula CSI3104-W11

4

A transition graph that accepts  $L_1^*$

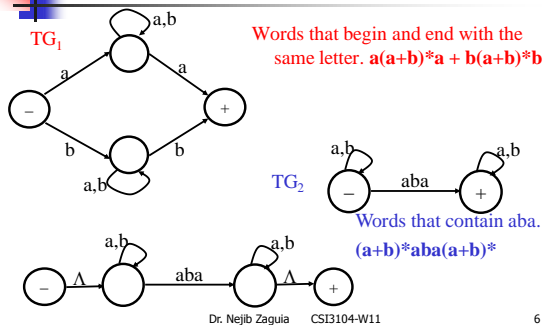


There exist transition graphs that accept  $L_1 + L_2$ ,  $L_1 L_2$ , and  $L_1^*$ . Thus, there are regular expressions that define them (by Kleene's theorem). Therefore

$L_1 + L_2$ ,  $L_1 L_2$  and  $L_1^*$  are regular languages (by definition).

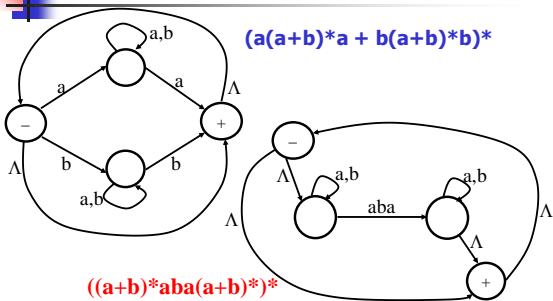
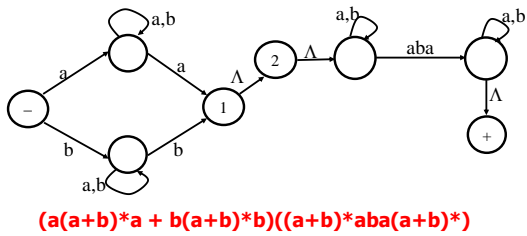
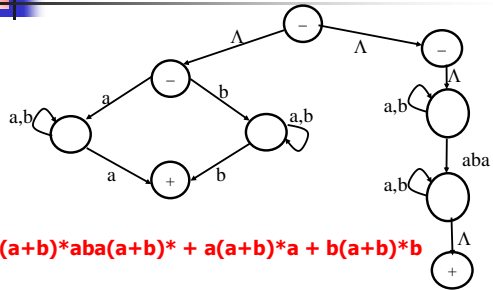
Dr. Nejib Zagula CSI3104-W11

5



Dr. Nejib Zagula CSI3104-W11

6





**Definition:** If  $L$  is a language over the alphabet  $\Sigma$ , the **complement of  $L$** , written  $L'$  is the language of all words on  $\Sigma$  that are not words in  $L$ . ( $L' = \Sigma^* - L$ ).

**Example:**

$S = \{a, b\}$

$L =$  all words containing  $aa$ .

$L' = ?$

$b^*(abb^*)^*(a+\Lambda)$

**Remark:**  $(L')' = L$ .

Dr. Nejib Zagula CSI3104-W11

10



**Theorem:** If  $L$  is a regular language, then  $L'$  is a regular language.

**Proof:** There exists a finite automaton that accepts  $L$  (by Kleene's theorem). All words accepted by this FA end in a final state. All words that are not accepted end in a state that is not a final state.

We reverse the final status of each state: all final states become non-final states, and all non-final states become final states.

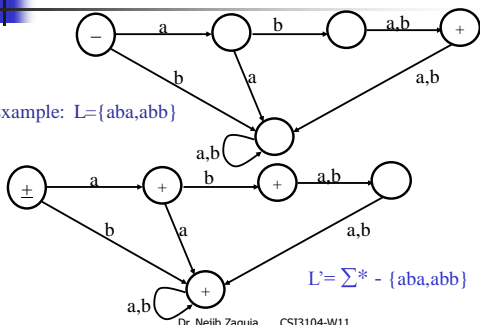
The new finite automaton accepts exactly those words that are not in  $L$ . By Kleene's theorem,  $L'$  is regular.

Dr. Nejib Zagula CSI3104-W11

11



Example:  $L = \{aba, abb\}$



Dr. Nejib Zagula CSI3104-W11

12



**Theorem:** Let  $L_1$  and  $L_2$  be two regular languages. Then  $L_1 \cap L_2$  is a regular language.

**Proof:**

$$L_1 \cap L_2 = (L_1' + L_2)'$$

If  $L_1$  and  $L_2$  are regular, then  $L_1'$  et  $L_2'$  are also regular.

If  $L_1'$  and  $L_2'$  are regular, then  $L_1' + L_2'$  is regular.

If  $L_1' + L_2'$  is regular, then  $(L_1' + L_2)'$  is regular.

Thus,  $L_1 \cap L_2$  is a regular language.

---

---

---

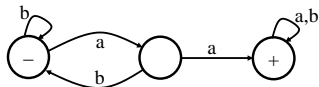
---

---

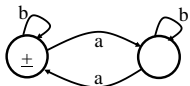
---

---

---



$L_1$ : words with double a



$L_2$ : words with an even number of a's

$$aaa \in L_1 \quad aba \in L_2$$

---

---

---

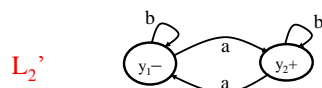
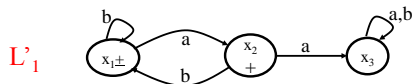
---

---

---

---

---




---

---

---

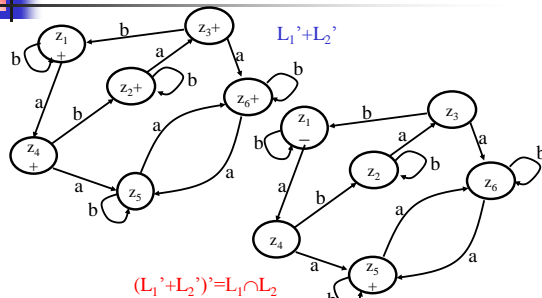
---

---

---

---

---



**Theorem:** Let  $L_1$  and  $L_2$  be two regular languages. Then  $L_1 \cap L_2$  is also a regular language.

**Proof 2:** By constructive algorithm. There exist finite automata that accept  $L_1$  and  $L_2$  (by Kleene's theorem). Recall the constructive algorithm for building a finite automaton for  $L_1 + L_2$  from the finite automata for  $L_1$  and  $L_2$ . We build the same finite automaton except for the final states. Each state in the new automaton represents a pair of states, one from each of the original finite automata,  $\{x_i, y_j\}$ . A state in the new machine is final if **both** of these states are final states in the original machines.