AUTOMATA THEORY AND FORMAL LANGUAGES 2015-16

UNIT 5 – PART 2: REGULAR LANGUAGES

David Griol Barres
Universidad Carlos III de Madrid

Regular Expressions. Bibliography

- Enrique Alfonseca Cubero, Manuel Alfonseca Cubero, Roberto Moriyón Salomón. Teoría de Autómatas y Lenguajes Formales. McGraw-Hill (2007). Section 7.2.

- John E. Hopcroft, Rajeev Motwani, Jeffrey D.Ullman. Introduction to Automata Theory, Languages, and Computation (3rd edition). Ed, Pearson Addison Wesley. Unit 3.

- Manuel Alfonseca, Justo Sancho, Miguel Martínez Orga. Teoría de Lenguajes, Gramáticas y Autómatas. Publicaciones R.A.E.C. 1997. Unit 7

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

OUTLINE

Unit 5. Part 2: Regular Expressions

- **Definition of a Regular Expression (RE)**
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Definition of Regular Expression

***Kleene, 1956*:**

“Metalanguage for expressing the set of words accepted by a FA (i.e. to express Type-3 or regular languages)"

Example: given the alphabet Σ= {0,1}

0*10* is a word of the metalanguage representing the infinite words which consist of a 1, preceded and followed by none, one or infinite zeros

Definition of Regular Expression

- Regular expressions: rules that define exactly the set of words that are included in the language.
- Main operators:
  - **Concatenation:** xy
  - **Alternation:** x|y (x or y)
  - **Repetition:** x^* (x repeated 0 or more times)

    x^+ (x repeated 1 or more times)

Definition of Regular Expression

- Given an alphabet Σ, the rules that define regular expressions of Σ are:
  - $\forall a \in \Sigma$ is a regular expression.
  - λ is a regular expression.
  - Φ is a regular expression.
  - If **r** and **s** are regular expressions, then

    (r) r·s r|s r*

    are regular expressions.

$$r^* = \bigcup_{i=0}^{\infty} r^i$$

Definition of Regular Expression

- Valid RE are those obtained after applying the previous rules a finite number of times over symbols of Σ, Φ, λ

- The priority of the different operations is the following:
  - *, • ,+

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- **Regular Expressions and Regular Languages**
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Regular expressions and Regular Languages

Each RE describes a regular language

- Each RE α has a set of Σ^* associated, L(α), that is the RL described by α. This language is defined by:

  - If $\alpha = \Phi$, $L(\alpha) = \Phi$
  - If $\alpha = \lambda$, $L(\alpha) = \{\lambda\}$
  - If α = a, a $\in \Sigma$, $L(\alpha) = \{a\}$
  - If α and β are RE $\Rightarrow L(\alpha \mid \beta) = L(\alpha) \cup L(\beta)$

Regular Expressions. Examples

Write the regular languages described by the following RE:

1) Given Σ = {a,b,...,z} and α = (a|b|...|z)*, what is L(α)?

2) Given Σ = {0,1} and α = 0*10*, what is L(α)?

3) Given Σ = {0,1} and α = 01|000, what is L(α)?

4) Given Σ = {a,b,c} and α = a (a|b|c)*, what is L(α)?

5) Given Σ = {a,b,c} and α = a|bc|b^2a, what is L(α)?

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- **Equivalence of Regular Expressions**
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Equivalence of Regular Expressions

- Two RE are equivalent, **α = β**, if they describe the same regular language, L(α) = L(β). <u>Properties</u>:
  1) (α | β) | σ = α | (β | σ) (| is associative)
  2) α | β = β | α (| is commutative)
  3) (α • β) • σ = α • (β • σ) (• is associative)
  4) α •(β | σ) = (α • β) | (α • σ) (| is distributive
     (β | σ) • α = (β • α) | (σ • α) regarding •)
  5) α • λ = λ • α = α (• has a neutral element)
  6) α | Φ = Φ | α = α (| has a neutral element)

Equivalence of Regular Expressions

9) $\Phi^* = \lambda$
10) $\alpha^* \bullet \alpha^* = \alpha^*$
11) $\alpha \bullet \alpha^* = \alpha^* \bullet \alpha$
12) $(\alpha^*)^* = \alpha^*$ (IMPORTANT)
13) $\alpha^* = \lambda \mid \alpha \mid \alpha^2 \mid .. \mid \alpha^n \mid \alpha^{n+1} . \alpha^*$
14) $\alpha^* = \lambda \mid \alpha \bullet \alpha^*$ (13 with n=0) (IMPORTANT)
15) $\alpha^* = (\lambda \mid \alpha)^{n-1} \mid \alpha^n \bullet \alpha^*$ (from 14)
16) Given a function f, $f: E^n_\Sigma \rightarrow E_\Sigma$ then:

$f(\alpha, \beta, ..., \sigma) \mid (\alpha \mid \beta \mid ... \mid \sigma)^* = (\alpha \mid \beta \mid ... \mid \sigma)^*$

Equivalence of Regular Expressions

18) $(\alpha^* \mid \beta^*)^* = (\alpha^* \bullet \beta^*)^* = (\alpha \mid \beta)^*$ (IMPORTANT)
19) $(\alpha \bullet \beta)^* \bullet \alpha = \alpha \bullet (\beta \bullet \alpha)^*$
20) $(\alpha^* \bullet \beta)^* \bullet \alpha^* = (\alpha \mid \beta)^*$
21) $(\alpha^* \bullet \beta)^* = \lambda \mid (\alpha \mid \beta)^* \bullet \beta$ (from 14 with 20)
22) Inference Rules:

given three regular expressions R,T and S:

$$R = S^* \bullet T \Rightarrow R = S \bullet R \mid T$$

If $\lambda \notin S$, then:

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- **Analysis Theorem and Kleene's Synthesis Theorem**
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Analysis and Kleene's Synthesis Theorems

1) Analysis Theorem:

Every language accepted by a FA is a regular language.

Solution to the problem of analysis: To find the language associated to a specific FA: **"Given a FA, A, find a RE that describes L(A)".**

2) Synthesis Theorem:

Every regular language is a language accepted by a FA.

Solution to the problem of synthesis: To find a recognizer

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - **Solution of the Analysis Problem. Characteristic Equations**
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Solution of the Analysis Problem. Characteristic Equations

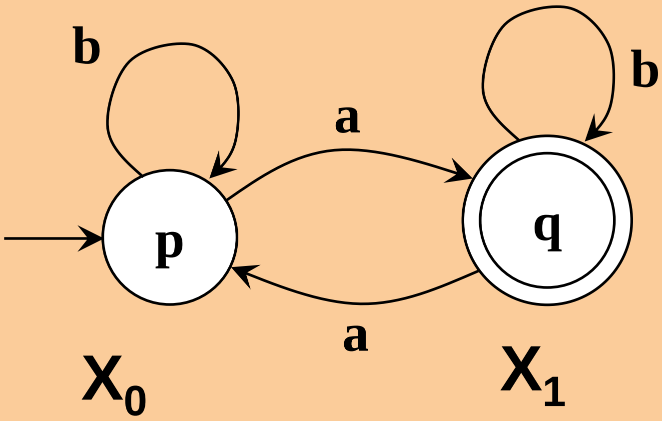
ANALYSIS PROBLEM (AF→RE): Given a FA, write the characteristic equations of each one of its states, solve them and obtain the requested RE.

- **CHARACTERISTIC EQUATIONS:** They describe all the strings that can be recognized from a given state:
  - An equation x_i is written for each state q_i
    - First member x_i;
    - The second member has a term for each branch from q_i
      - Branches has the format $a_{ij} \cdot x_j$ where a_{ij} is the label of the branch that joins qi with q_j, x_j is the variable corresponding to q_j
      - A term a_{ij} is added for each branch that joins q_i with a final state.

If it not a final state: $x_j = Φ$

Solution of the Analysis Problem. Characteristic Equations

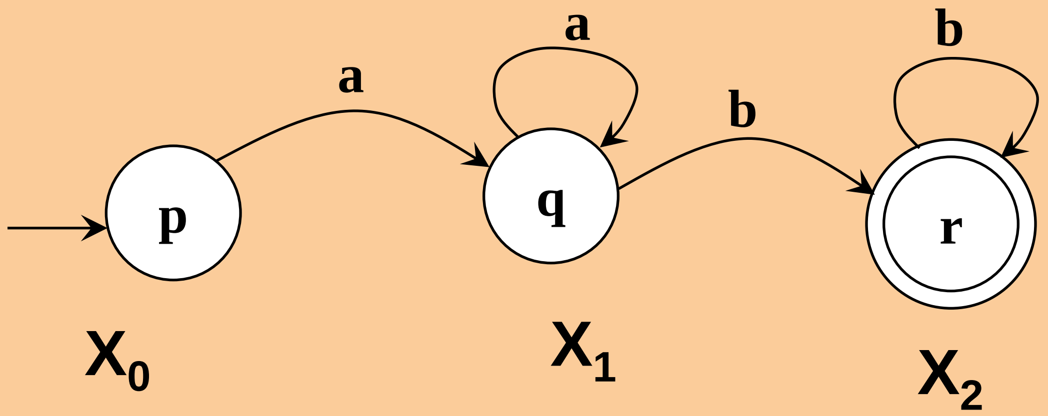
Exercise 1



$$\mathbf{X_0} = b\,X_0 + a\,X_1 + a$$

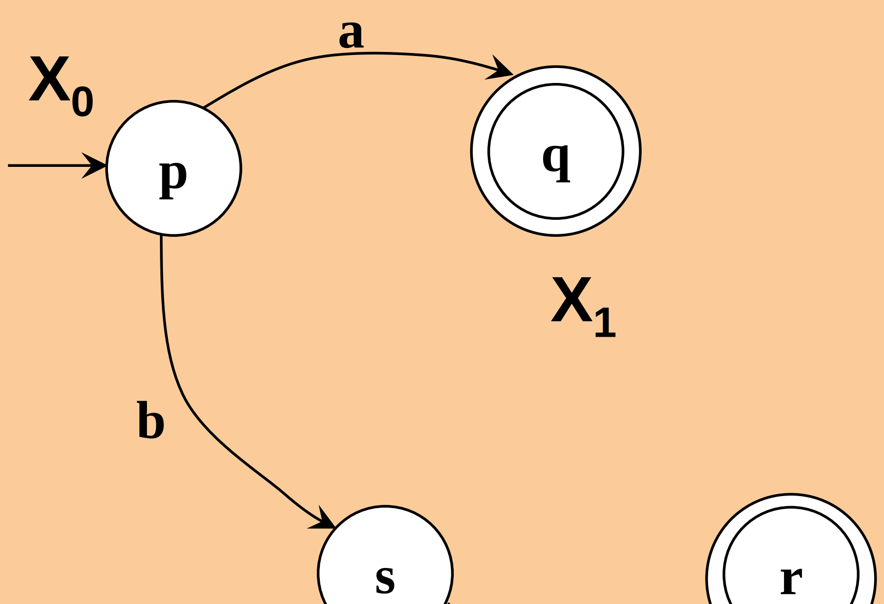
Solution of the Analysis Problem. Characteristic Equations

Exercise 2



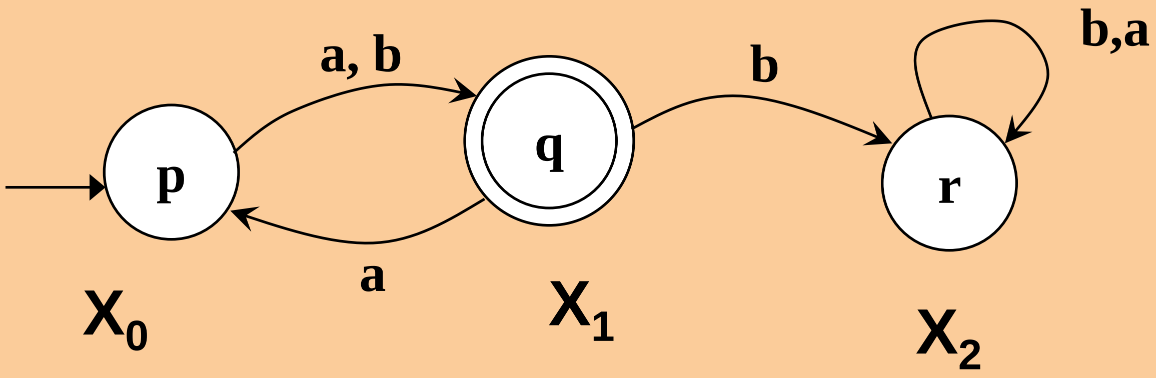
Solution of the Analysis Problem. Characteristic Equations

Exercise 3



Solution of the Analysis Problem. Characteristic Equations

Exercise 4



OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - **Solution of the Characteristic Equations**
    - Algorithm to Solve the Analysis Problem

Solution of the Characteristic Equations

They have the form: **X = AX + B**

where:

X: set of strings that allow transitting from q_i to $q_f \in F$

A: set of strings that allows reaching a state q from q.

B: set of strings that allows reaching a final state, without reaching again the leaving state q_i.

$\Downarrow$ (Arden solution or proof by contradiction)

OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - **Algorithm to Solve the Analysis Problem**

Solution of the Analysis Problem. Algorithm

1. Write the characteristic equations of the FA.

2. Resolve them.

3. If the initial state is q_0, X_0 gives us the set of strings that leads from q_0 to q_f and, therefore,

OUTLINE

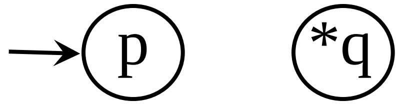
Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

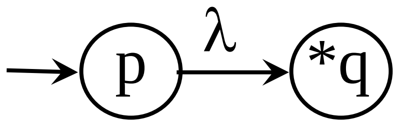
Synthesis Problem: Recursive Algorithm

SYNTHESIS PROBLEM (RE→FA): “Given an RE representing a regular language, build a FA that accepts that regular language.

- Given a regular expression α:
  - If $\alpha = \Phi$, the automaton is:

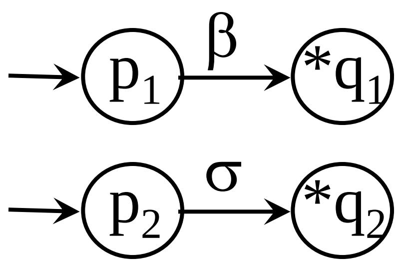


  - If $\alpha = \lambda$, the automaton is:

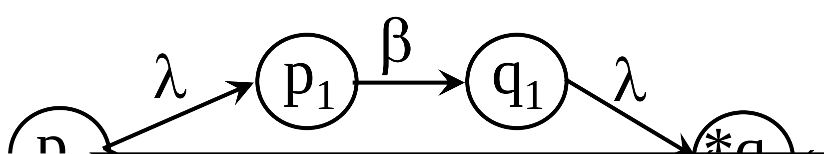


Synthesis Problem: Recursive Algorithm

- If $\alpha=\beta|\sigma$, using the automata β and σ

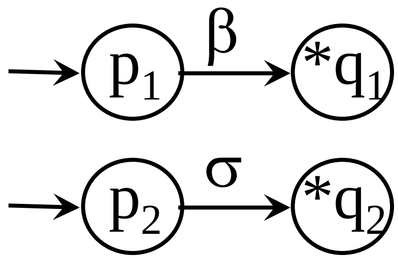


the result is:

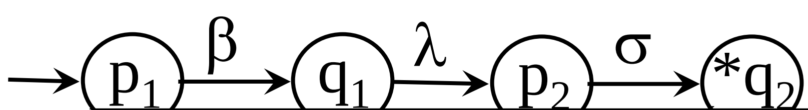


Synthesis Problem: Recursive Algorithm

- If $\alpha=\beta \bullet \sigma$, using the automata β and σ

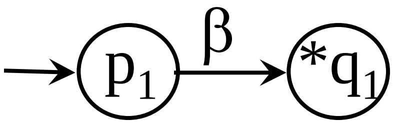


the result is:

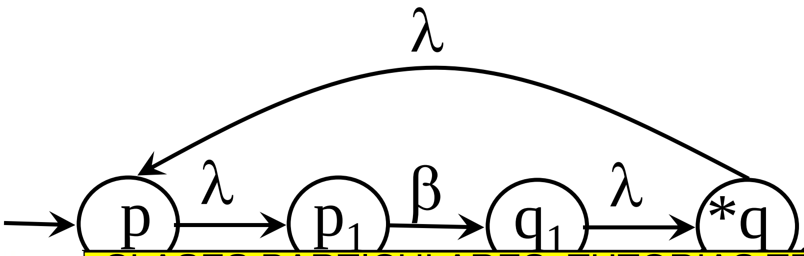


Synthesis Problem: Recursive Algorithm

- If $\alpha=\beta^*$, using the automata β



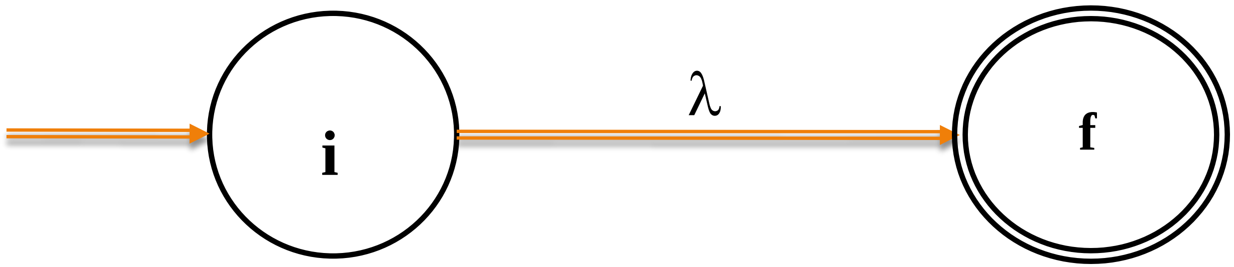
the result is:



Synthesis Problem: Recursive Algorithm

Summary

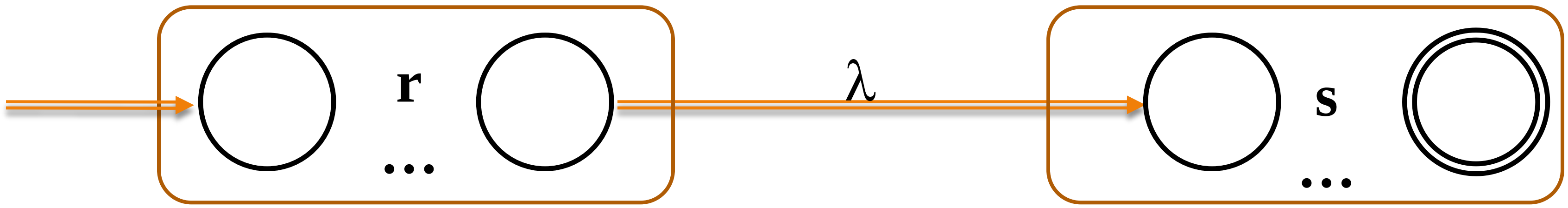
Basic Regular expressions (λ, a):



Synthesis Problem: Recursive Algorithm

Summary

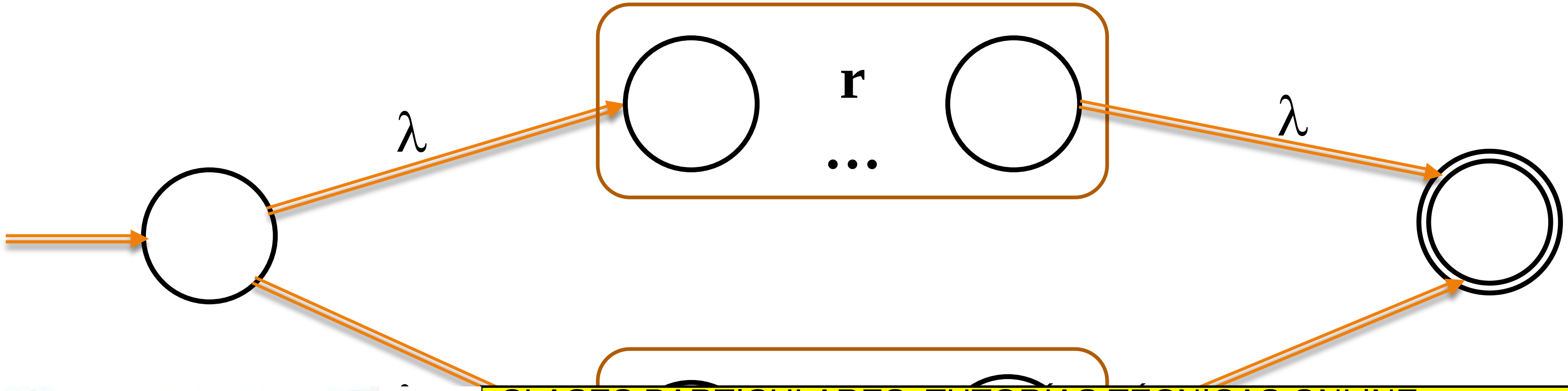
Concatenation rs:



Synthesis Problem: Recursive Algorithm

Summary

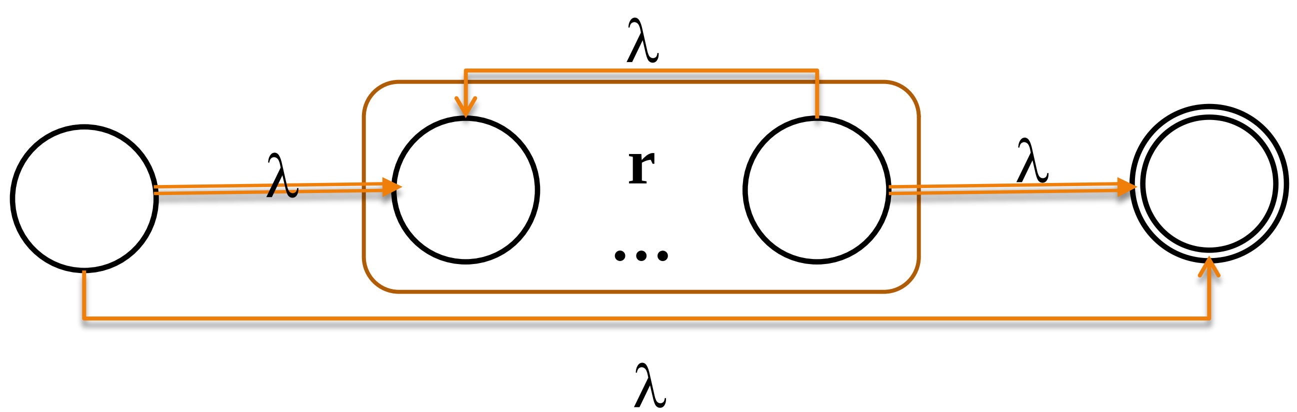
Selection r | s:



Synthesis Problem: Recursive Algorithm

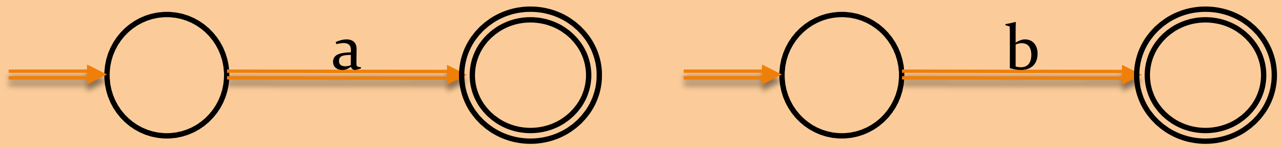
Summary

Repetition r^*:

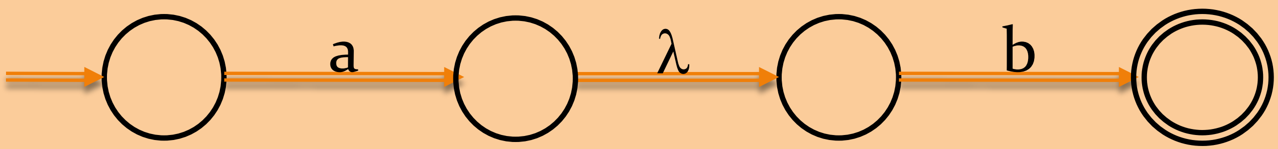


Synthesis Problem: Recursive Algorithm

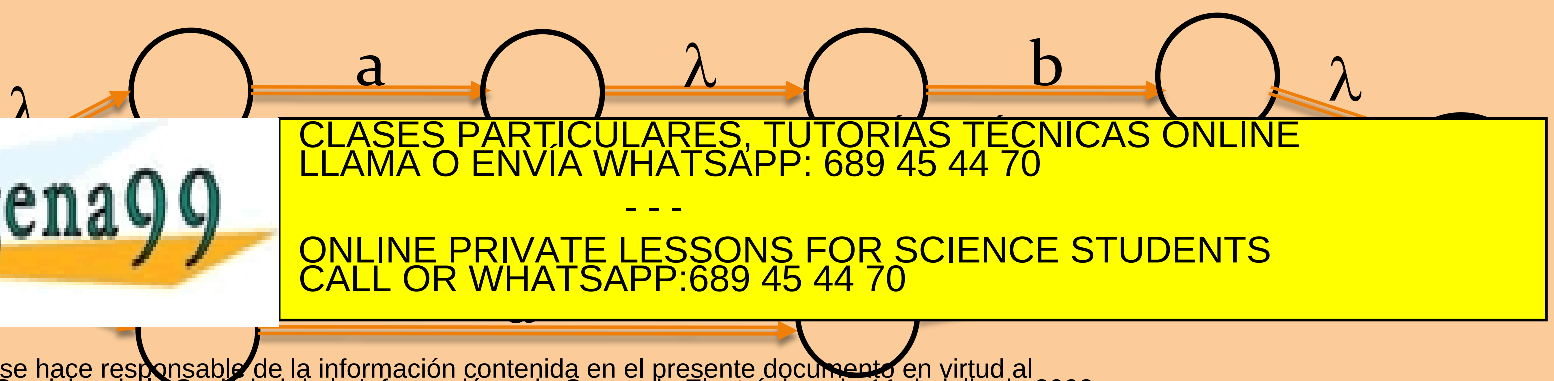
Example 1: ab | a



ab

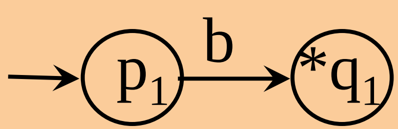


ab|a

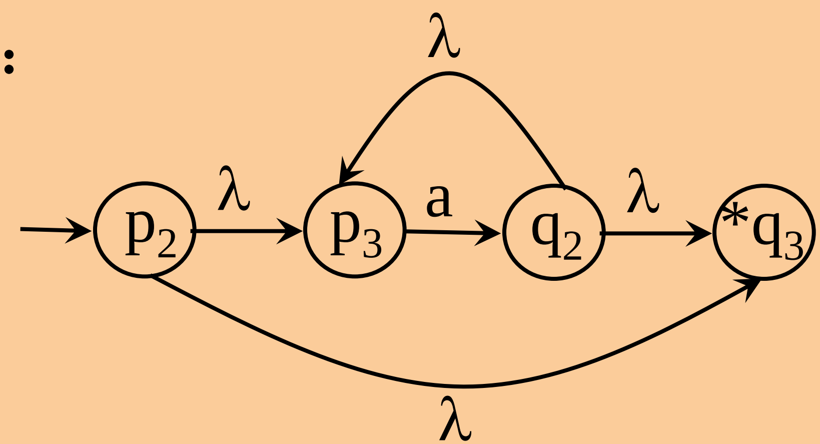


Synthesis Problem: Recursive Algorithm

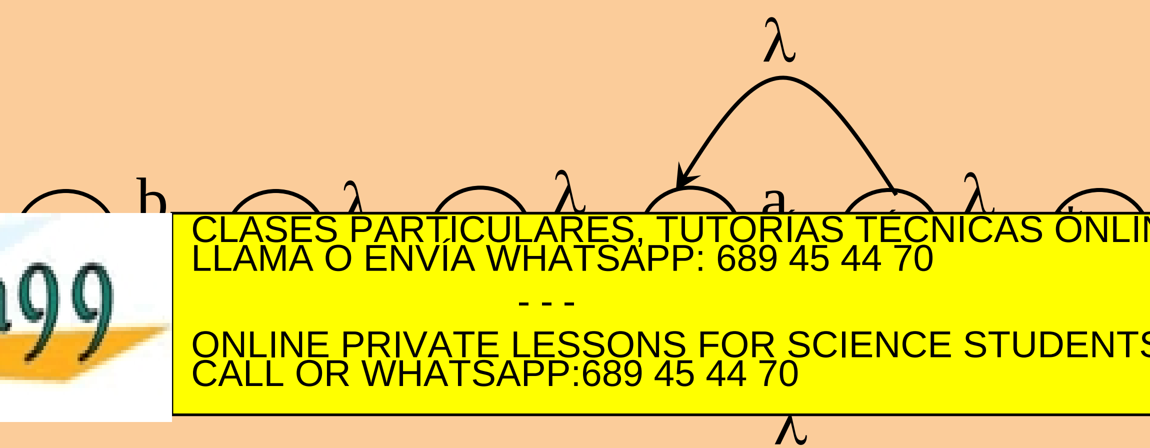
- Example $\alpha=(b \bullet a^*)^*$
  - b:



a*:

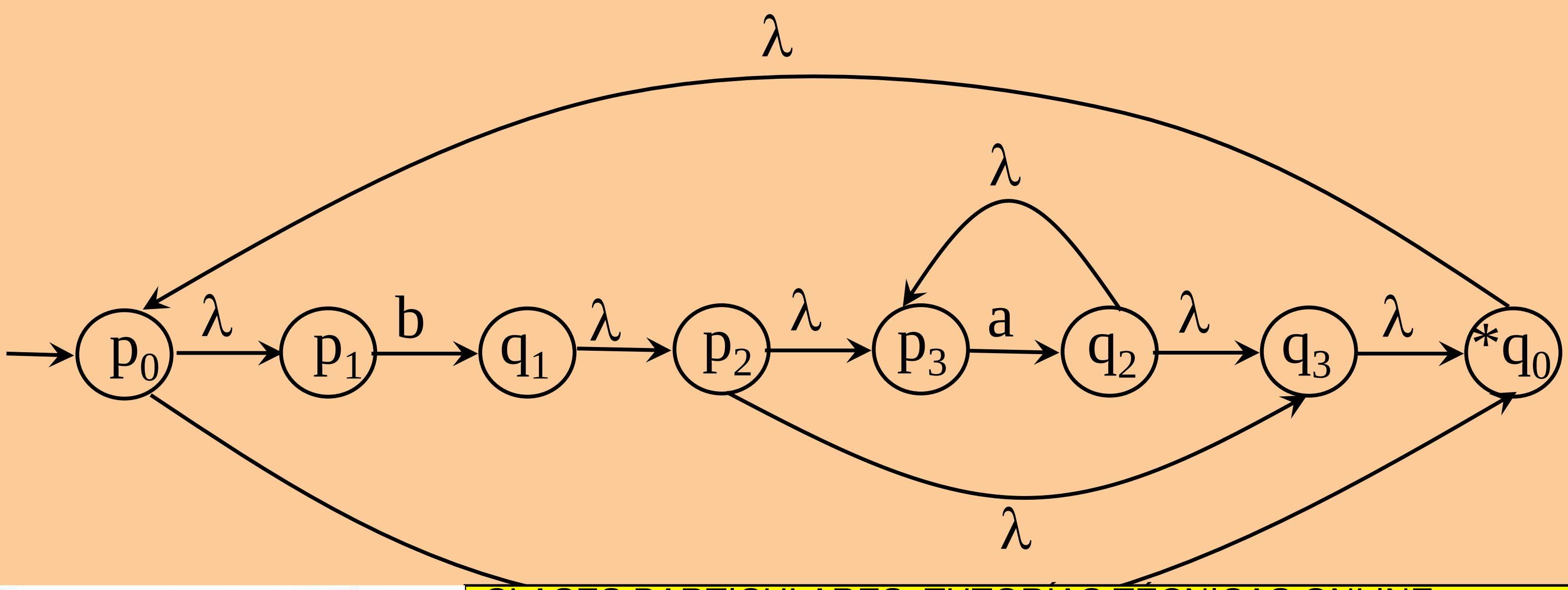


  - b • a*



Synthesis Problem: Recursive Algorithm

- (b • a*)*



OUTLINE

Unit 5. Part 2: Regular Expressions

- Definition of a Regular Expression (RE)
- Regular Expressions and Regular Languages
- Equivalence of Regular Expressions
- Analysis Theorem and Kleene's Synthesis Theorem
  - Solution of the Analysis Problem. Characteristic Equations
    - Solution of the Characteristic Equations
    - Algorithm to Solve the Analysis Problem

Solution to the synthesis problem: Derivatives of Regular Expressions

- Given a RE, construct a FA which recognizes the language that the RE describes.
  - Derive the RE and obtain a Right-Linear G3 and, from it, a FA.
    - Derivative of a RE?
- Derivative of a RE: $D_a(R) = \{ x \mid a \bullet x \in R \}$.
    - Derivative of a regular expression R with regard an input symbol

Solution to the synthesis problem: Derivatives of Regular Expressions

Given an RE → right-linear G3 grammar → FA which recognizes the language that describes the ER.

$$D_a(R) = \{ x \mid a.x \in R \}$$

Derivative of a RE: Recursive definition. $\forall$ a, b $\in \Sigma$ and R, S Reg. Exp.

- $D_a(\Phi) = \Phi$
- $D_a(\lambda) = \Phi$
- $D_a(a) = \lambda, \quad a \in \Sigma$
- $D_a(b) = \Phi, \quad \forall\, b \neq a, b \in \Sigma$
- $D_a(R+S) = D_a(R) + D_a(S)$
- $D_a(R \bullet S) = D_a(R) \bullet S + \delta(R) \bullet D_a(S), \quad \forall R$
- $D_a(R^*) = D_a(R) \bullet R^*$

Solution to the synthesis problem: Derivatives of Regular Expressions

- **Definition: $D_{ab}(R)=D_b(D_a(R))$**
- From a derivative of a RE, obtain the right-linear G3 grammar.
  - The number of different derivatives of a RE is finite.
  - Once all have been obtained, you can obtain the G3 grammar:
  - **Given $D_a(R) = S$, with $S \neq \Phi$**
    - $S \neq \lambda \Rightarrow R ::= aS \in P$
    - $S = \lambda \Rightarrow R ::= a \in P$
  - **Given $\delta(D_a(R)) = S$**
    - $\delta(D_a(R)) = \lambda \Rightarrow R ::= a \in P$
    - $\delta(D_a(R)) = \Phi \Rightarrow$ no rules included in P

- ΣN = letters which distinguish each one of the different derivatives.

Solution to the synthesis problem: Derivatives of Regular Expressions

Obtain the G3 RL grammars that are equivalent to the following RE:

- $R = a \bullet a^* \bullet b \bullet b^*,\ \Sigma=\{a,b\}$
  - $D_a(R) = D_a(a)\ a^*\ b\ b^* = a^*\ b\ b^*$
  - $D_b(R) = \Phi$
  - $D_{aa}(R) = D_a(a^*\ b\ b^*) = D_a(a^*)\ b\ b^* + \lambda\ D_a(b\ b^*) = a^*bb^* = D_a(R)$
  - $D_{ab}(R) = D_b(a^*\ b\ b^*) = D_b(a^*)\ b\ b^* + \lambda\ D_b(b\ b^*) = b^*$
  - $D_{aba}(R) = D_a(b^*) = \Phi$
  - $D_{abb}(R) = D_b(b^*) = D_b(b)\ b^* = b^* = D_{ab}(R)$

  - $D_a(R)= a^*bb^*$ $\delta(D_a(R))= \Phi$
  - $D_{abb}(R)$ ... $\delta(D_{abb}(R)) = \lambda$

Solution to the synthesis problem: Derivatives of Regular Expressions

- R_0=aa*bb* R_1=a*bb* R_2=b*
  - $D_a(R_0)=R_1$ $\delta(D_a(R_0))= \Phi$
  - $D_a(R_1)=R_1$ $\delta(D_a(R_1))= \Phi$
  - $D_b(R_1)=R_2$ $\delta(D_b(R_1))= \lambda$
  - $D_b(R_2)=R_2$ $\delta(D_b(R_2))= \lambda$
- $D_a(R)=S \Rightarrow R \rightarrow aS$ $\delta(D_a(R))= \lambda \Rightarrow R \rightarrow a$
  - $R_0 \rightarrow aR_1$ ----------
  - $R_1 \rightarrow aR_1$ ----------