AUTOMATA THEORY AND FORMAL LANGUAGES 2015-16

UNIT 5 – PART 1: REGULAR LANGUAGES

Universidad Carlos III de Madrid

Regular languages. Bibliography

- Enrique Alfonseca Cubero, Manuel Alfonseca Cubero, Roberto Moriyón Salomón. Teoría de Autómatas y Lenguajes Formales. McGraw-Hill (2007). Chapters 3 and 7.

- John E. Hopcroft, Rajeev Motwani, Jeffrey D.Ullman. Introduction to Automata Theory, Languages, and Computation (3rd edition). Ed, Pearson Addison Wesley. Sects. 2.1-2.2; Sects. 2.3-2.8; Chap. 4; Sects. 3-1-3.7

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

OUTLINE

PART 1:

- **Finite Automata and Type-3 Grammars**
  - **Finite Automata associated to a Type-3 grammar (G3→FA)**
  - **Type-3 Grammar associated to a FA (FA→G3)**

PART 2:

- Regular expressions and Regular Languages

From FA to Type-3 grammar

1 **From FA → G3:**

Given the FA, A = (Σ, Q, qo, f, F), there is a right-linear grammar that fulfills

L(G3RL) = L(A)

That it is to say, the language generated by the grammar is the same that the recognized by the automaton

<u>Following:</u> How to obtain the grammar G={Σ_T, Σ_N, S, P}

from the FA= {Q, Σ, q, f, F}

From FA to Type-3 grammar

1 **From FA → G3:**

Process:

- $\Sigma_T = \Sigma$; Σ_N **= Q,** S **= qo**
- P= {…}

1. Transition f(p,a) =q → if q' is not a final state → p::= aq
2. q∈F and f(p,a) =q → p::= a and p::= aq
3. p0 ∈ F → p0 ::= λ
4. If f(p, λ) = q→ p::= q
5. q ∈ F and f(p, λ) = q → p ::= q and q::= λ

From FA to Type-3 grammar

1 **From FA→ G3: Example**

Given the FA described by the following table, calculate the right-linear G3 grammar that generates the language described by it. Verify that both languages are the same.

	0	**1**
→A	A	C
B	A	C
*C	C	B

From Type-3 grammar to FA

2 **From G3 → FA:**

Given a right-linear G3, G = (Σ_T, Σ_N S, P), there is a FA, A, that fulfills: L(G3LD) = L(A)

<u>Process</u>:

- $\Sigma = \Sigma_T$
- $Q = \Sigma_N \cup \{F\}$, with $F \notin \Sigma_N$
- qo = S
- F = {F}
- f:
  - If A ::= aB → f(A,a) = B

From Type-3 grammar to FA

2 **From G3 → FA : Example**

Given the following right-linear G3 right-linear grammar, calculate the equivalent FA.

G = ({d,c}, {A,S,T}, A, {A ::= cS, S::= d/cS/dT, T::= dT/d})

From Type-3 grammar to FA

- We have seen the procedure to obtain a FA that accepts the language described by a G3 left-linear grammar, however, this procedure does not always lead to an DFA, typically:

$$G3 \rightarrow NFA \rightarrow DFA$$

- **<u>Exercise 1</u>**: Given the left-linear grammar: G= ({0,1}, {S,U}, S,{S ::= U0, U ::= U0 | S1 | 0}) Calculate the corresponding DFA.

- **Exercise 2**: Given the left-linear grammar: G= ({0,1}, {S,U}, S,{S ::= U0

From Type-3 grammar to FA

Given the left-regular grammar G3: $G = (\Sigma_T, \Sigma_N, S, P)$

From it, we build the FA: $A= (\Sigma_T, \Sigma_N \cup \{p,q\}, f, p, \{S\})$

$\{p,q\}$: Q

where: $p, q \notin \Sigma_T$ and/or Σ_N

f is defined by:

1) $f(U,t) = V$ si $V ::= U\,t \in P$

2) $f(p,t) = V$ si $V ::= t \in P$

3) $f(U,t) = q \quad \forall t \in \Sigma_T \,/\, V ::= U\,t \notin P$

From Type-3 grammar to FA

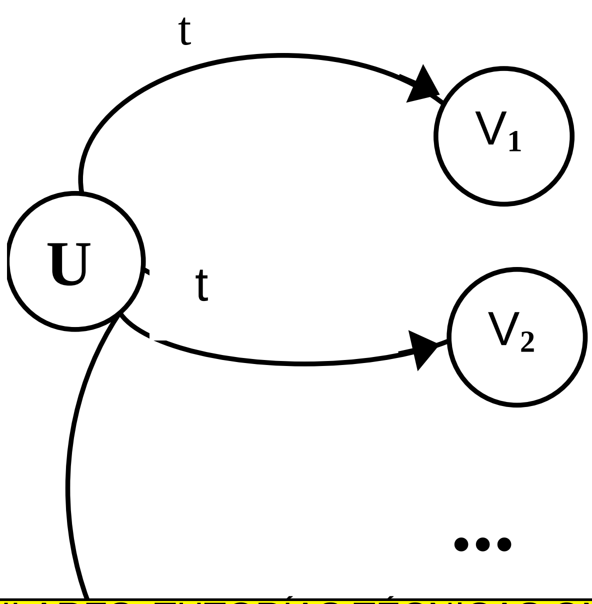
- This definition does not ensure a deterministic FA since it is possible:

  V1 ::= Ut

  V2 ::= Ut

  ...

  V3 ::= Ut



From Type-3 grammar to FA

Given the G3 left-linear grammar:

G= ({0,1}, {S,U,V}, S, P)

Where P = {S ::= U0 / V1

U ::= S1 / 1

V ::= S0 /0}

Calculate the minimum DFA that recognizes the language generated by G.

Steps: 1) Calculate the FA (Determinist in this case)

2) Minimize it.

Additional Issues

And if we want to obtain a FA from a left-linear G3?

G3 left-linear → G3 right-linear → FA

And if we want to obtain a left-linear G3 from a FA?

FA → G3 right-linear → G3 left-linear