MMU
MULTIMEDIA UNIVERSITY
Faculty of
COMPUTING & INFORMATICS

TIC 2151 – Theory of Computation

Lecture 4

Regular Grammars

Dr. Nabhan D. Salih

Lecture 4 - Outline

- What is a grammar?
- Converting FA to regular grammar
- Converting regular grammar to FA.
- In class exercises and/or demonstrations.

What is a grammar?

- Grammar is another way to describe the languages.
- We have studied three characterizations of the regular languages:
  1. Deterministic finite automata, DFA
  2. Nondeterministic finite automata, NFA
  3. Regular expressions, RE

In this class we will study another way to characterize the regular languages called **Regular Grammars**

- Every regular language can be expressed by a regular grammar.
- Regular grammar is more powerful way of describing languages than finite automata.

Formal Definition of a Grammar

Grammar G = (V, Σ, R, S) where

- **V** is variables, nonterminals (finite set, nonempty set)
- **Σ** is terminals, alphabet (finite set, nonempty set with $V \cap \Sigma = \varnothing$, disjoint from V)
- **R** is rules or productions (finite set), with each rule being a variable and a string of variables and terminals
- **S** $\in$ V is the start variable.

Example:
$G_1 = (\{S\}, \{a\}, \{[S,a], [S,aS]\}, S)$

Formal Definition of a Grammar

Example: consider the language represented by a^+, which is {a, aa, aaa, . . } . One can generate the strings of this language by the following procedure:

Given G_1 = ({S}, {a}, {[S,a], [S,aS]}, S)
the rules of the grammar are

$$\left.\begin{array}{l} S \to a \\ \\ S \to aS \end{array}\right\}$$

(or shorter: $S \to a \mid aS$) means S is rewritten as ***a*** or as ***aS***.

And we can derive words starting from the start variable S
and using the rules like:

$$\begin{tikzcd}[row sep=small]
S \arrow[r] \arrow[dr] & a & & \\
& aS \arrow[r] \arrow[dr] & aa & \\
& & aaS \arrow[r] \arrow[dr] & \cdots \\
& & & \cdots
\end{tikzcd}$$

So the words produced by this grammar are: a, aa, aaa, ...

Formal definition of regular grammar

❖ A grammar G = (V, Σ, R, S) is *regular* if every rule takes one of two forms:

$B \rightarrow aC$ (right regular)

$B \rightarrow a$

** *B* and *C* are variables (nonterminal) , and *a* is a sequence of terminals

Some more examples

G_2: $S \Rightarrow aS \mid T$
$T \Rightarrow bT \mid \varepsilon$

$L(G_2) = \{a^n b^m \mid n \geq 0, m \geq 0\}$

G_3: $S \Rightarrow a \mid aS \mid T$
$T \Rightarrow b \mid bT$

What's the difference between G_2 and G_3?

G_4: $S \Rightarrow T_1 \mid T_2$
$T_1 \Rightarrow bb \mid bbT_1$
2 $T_2 \Rightarrow bbb \mid bbbT_2$

$L(G_4) = \{b^{2n}\} \cup \{b^{3n}\}$, $n \geq 1$

G_5: $S \Rightarrow abS$
$S \Rightarrow a$

$L(G_5) = (ab)^*a$

G_6: $S \Rightarrow Aa$
$A \Rightarrow Aab \mid \varepsilon$

What's the difference between G_5 and G_6?

Some more examples

G_7: $S \Rightarrow Aab$
$A \Rightarrow Aab \mid B$
$B \Rightarrow a$

$L(G_7) = aab(ab)^*$

G_8: $S \Rightarrow aaA$
$A \Rightarrow bB$
$B \Rightarrow aA \mid \varepsilon$

What's the difference between G_7 and G_8?

Converting FA to regular grammar

NFA $\Rightarrow$ Regular Grammar (right regular)

step 1 – convert edges from V to W consuming a, to productions of the form $V \rightarrow aW$

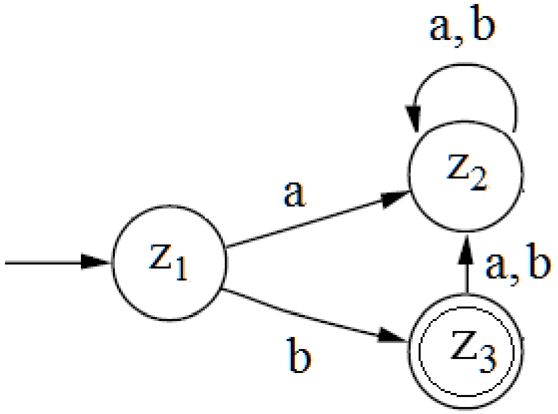
step 2 – convert null edges from V to W into productions of the form $V \rightarrow W$

step 3 – convert accept states (say, V) into rules of the form $V \rightarrow \varepsilon$

Converting FA to regular grammar

NFA ⇨ Regular Grammar (right regular)

Every regular language can be expressed by a regular grammar.



Transform

$Z_1 \Rightarrow aZ_2 \mid bZ_3$

$Z_2 \Rightarrow aZ_2 \mid bZ_2$

$Z_3 \Rightarrow aZ_2 \mid bZ_2 \mid \varepsilon$

(ε because Z_3 is an accept state)

Converting regular grammar to FA

NFA $\Longleftarrow$ Regular Grammar (right regular)

Step 1 – create states for each non-terminal

Step 2 – productions of the form $V \rightarrow aW$ result in a transition from V to W that consumes symbol a.

Step 3 – productions of the form $V \rightarrow ab\ldots cW$ result in a series of transitions starting at V and ending at W, that consume the symbols $a, b, \ldots, c$ and pass through intermediate series of states as necessary.

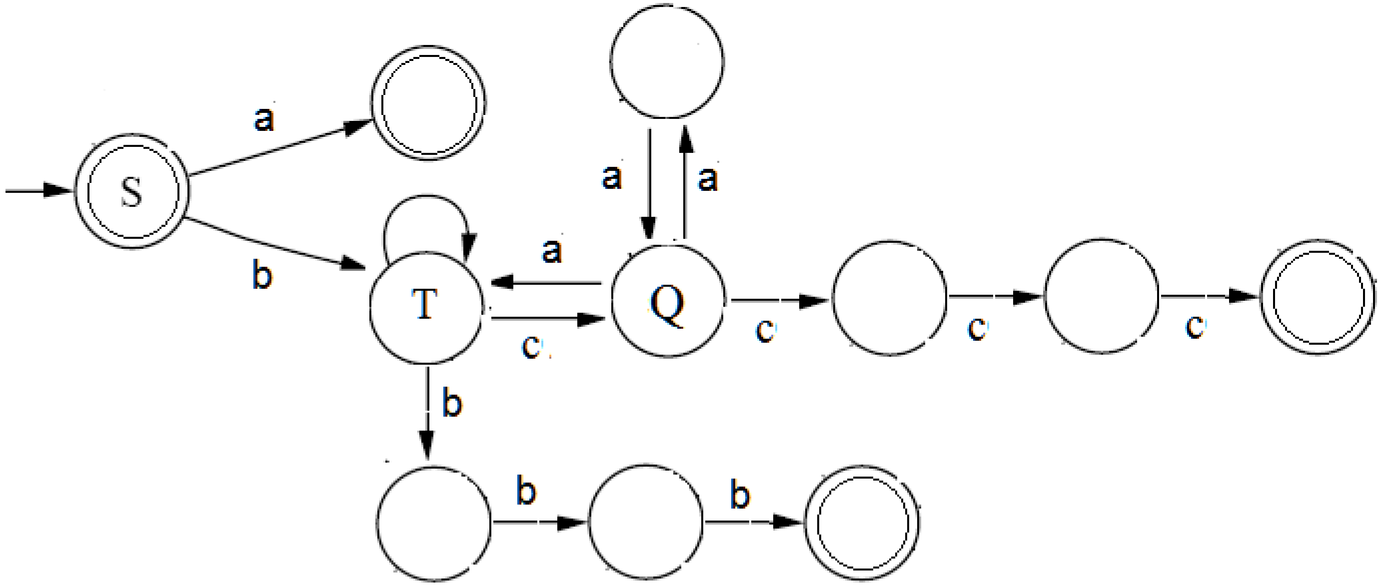
Step 4 – productions of the form $V \rightarrow ab\ldots c$ result in a series of transitions starting at V and ending in an accept state, passing through an intermediate series of states as necessary.

Converting regular grammar to FA

FA ⟸ Regular Grammar

Every regular grammar produces a regular language.

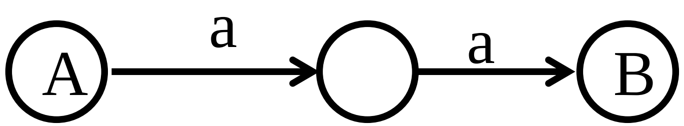
S => a | bT | ε
T => bT | cQ | bbb
Q => aaQ | ccc | aT



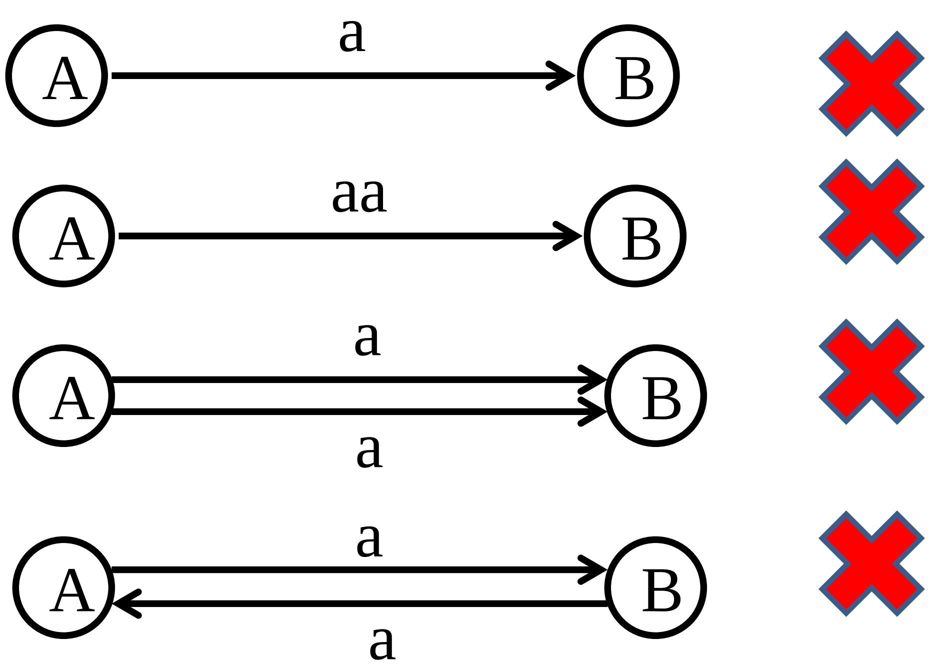
Converting regular grammar to FA

Take care! For example when we have $A => aaB$

Means



and not



In class exercises/demonstrations (1)

- Write a regular grammar for the languages over alphabet {0, 1}.

  $L1 = \{w \in \{0, 1\}^* \mid w$ begins with $01\}$

  $L_2 = \{w \in \{0, 1\}^* \mid w$ ends with $1\}$

  $L_3 = \{w \in \{0, 1\}^* \mid w$ begins with 01 and ends with $1\}$

  $L_4 = \{w \in \{0, 1\}^* \mid$ the length of w is at least three$\}$

In class exercises/demonstrations (2)

❖ construct the corresponding NFA for the following regular grammar

$$\Sigma = \{a, b\},\ V = \{ S \} \text{ and } R = \{ S \rightarrow aS,\ S \rightarrow bS,\ S \rightarrow a,\ S \rightarrow b \}$$

SOLUTION

R

$S \rightarrow aS \mid bS \mid a \mid b$

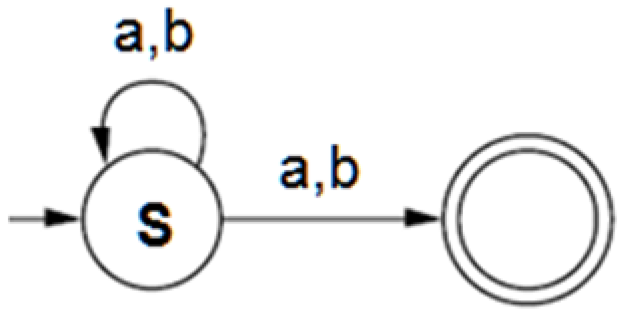
OR

$S \rightarrow aS$

$S \rightarrow bS$

$S \rightarrow a$

$S \rightarrow b$



The regular grammar generates the language $(a + b)^{+}$

In class exercises/demonstrations (3)

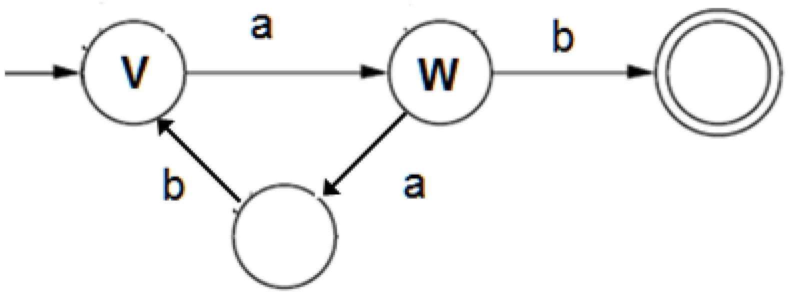
- construct the corresponding NFA for the following regular grammar ***M***

  V → aW

  W → abV | b

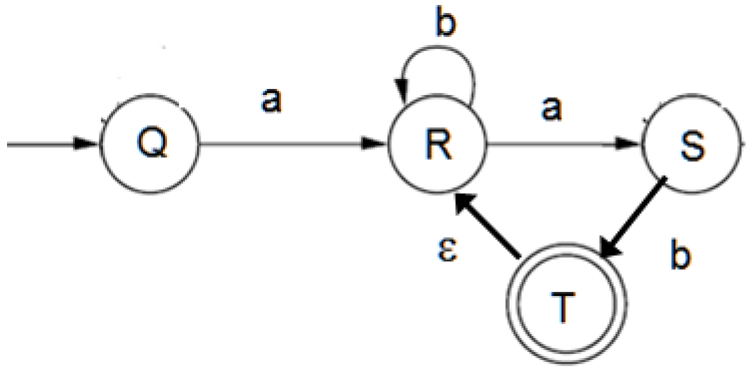
SOLUTION

M is clearly right-linear.



In class exercises/demonstrations (4)

- What is the regular grammar corresponding to the NFA given below



SOLUTION

$Q \rightarrow aR$

$R \rightarrow bR$

$R \rightarrow aS$

$S \rightarrow bT$

$T \rightarrow R \quad (null\ edges)$

$T \rightarrow \varepsilon \quad (\varepsilon\ because\ T\ is\ an\ accept\ state)$

In class exercises/demonstrations (5)

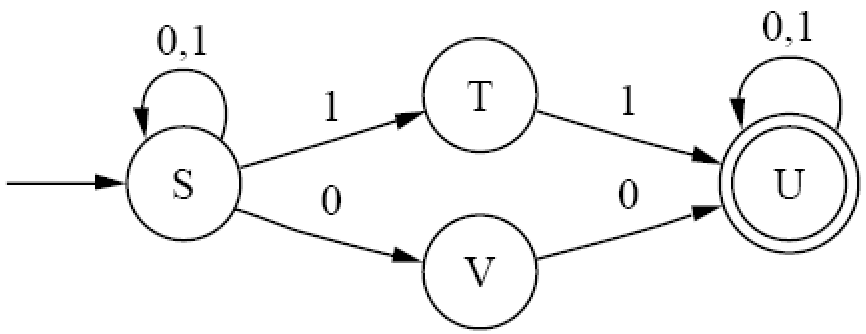
- Write a regular grammar for the following language over alphabet {0, 1}and construct the corresponding NFA's.

{w ∈ {0, 1}* w | containts the substring 00 or 11}

SOLUTION

G = (V, Σ, R, S) = ({S,T,V,U}, {0, 1}, {[S,0S], [S,1S] , [S,1T], [S,0V],[T,1U], [V,0U], [U,0U], [U,1U], [U,U]}, S)

$$\begin{aligned} S &\rightarrow 0S \mid 1S \mid 1T \mid 0V \\ T &\rightarrow 1U \\ V &\rightarrow 0U \\ U &\rightarrow 0U \mid 1U \mid \varepsilon \end{aligned}$$



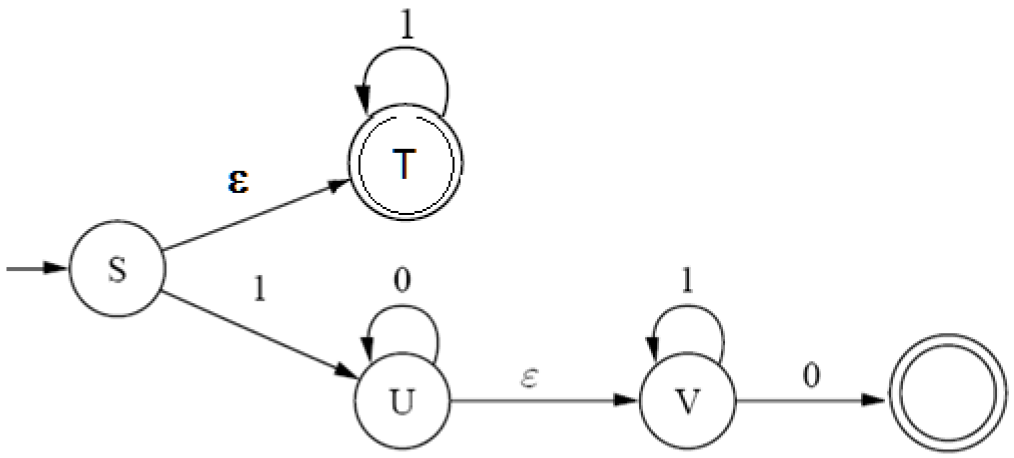
In class exercises/demonstrations (6)

❖ Write a regular grammar for the following language over alphabet {0, 1}and construct the corresponding NFA's.

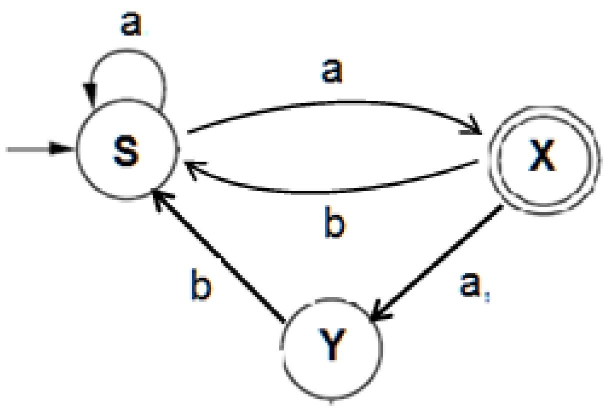
$1^* \cup 10^*1^*0$

SOLUTION

$$
\begin{aligned}
S &\rightarrow T \mid 1U \\
T &\rightarrow 1T \mid \varepsilon \\
U &\rightarrow 0U \mid V \\
V &\rightarrow 1V \mid 0
\end{aligned}
$$



- What is the regular grammar corresponding to the NFA given below



SOLUTION

G = (V, Σ, R, S) = ({S, X, Y }, {0, 1}, { S → aS, S → aX, X → bS, X→Ɛ, X→aY, Y→bS}, S)

Learning Outcomes

After completing this lecture, you should be able to:

- Define what a grammar is (in particular regular ones)
- Convert a regular grammar into a FA, and vice-versa.
- Produce grammar rules for a given description of a regular language.
- Distinguish between regular expressions and regular grammars.