

Theory of Computer Science

B3. Regular Languages

Gabriele Röger

University of Basel

March 15, 2021

Introduction

Repetition: Regular Grammars

Definition (Regular Grammars)

A **regular grammar** is a 4-tuple $\langle V, \Sigma, R, S \rangle$ with

- V finite set of variables (nonterminal symbols)
- Σ finite alphabet of terminal symbols with $V \cap \Sigma = \emptyset$
- $R \subseteq (V \times (\Sigma \cup \Sigma V)) \cup \{ \langle S, \varepsilon \rangle \}$ finite set of rules
- if $S \rightarrow \varepsilon \in R$, there is no $X \in V, y \in \Sigma$ with $X \rightarrow yS \in R$
- $S \in V$ start variable.

Repetition: Regular Grammars

Definition (Regular Grammars)

A **regular grammar** is a 4-tuple $\langle V, \Sigma, R, S \rangle$ with

- V finite set of variables (nonterminal symbols)
- Σ finite alphabet of terminal symbols with $V \cap \Sigma = \emptyset$
- $R \subseteq (V \times (\Sigma \cup \Sigma V)) \cup \{ \langle S, \varepsilon \rangle \}$ finite set of rules
- if $S \rightarrow \varepsilon \in R$, there is no $X \in V, y \in \Sigma$ with $X \rightarrow yS \in R$
- $S \in V$ start variable.

Rule $X \rightarrow \varepsilon$ is only allowed if $X = S$ and
 S never occurs in the right-hand side of a rule.

Question (Slido)

With a regular grammar, how many steps does it take to derive a non-empty word (over Σ) from the start variable?



Repetition: Regular Languages

A language is regular if it is generated by some regular grammar.

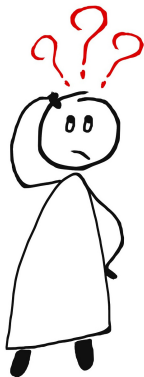
Definition (Regular Language)

A language $L \subseteq \Sigma^*$ is **regular** if there exists a regular grammar G with $\mathcal{L}(G) = L$.

Questions

- How restrictive is the requirement on ϵ rules?
If we don't restrict the usage of ϵ as right-hand side of a rule, what does this change?
- How do regular languages relate to finite automata?
Can all regular languages be recognized by a finite automaton? And vice versa?
- With what operations can we “combine” regular languages and the result is again a regular language?
E.g. is the intersection of two regular languages regular?

Questions



Questions?

Epsilon Rules

Repetition: Regular Grammars

Definition (Regular Grammars)

A **regular grammar** is a 4-tuple $\langle V, \Sigma, R, S \rangle$ with

- V finite set of variables (nonterminal symbols)
- Σ finite alphabet of terminal symbols with $V \cap \Sigma = \emptyset$
- $R \subseteq (V \times (\Sigma \cup \Sigma V)) \cup \{\langle S, \varepsilon \rangle\}$ finite set of rules
- if $S \rightarrow \varepsilon \in R$, there is no $X \in V, y \in \Sigma$ with $X \rightarrow yS \in R$
- $S \in V$ start variable.

Rule $X \rightarrow \varepsilon$ is only allowed if $X = S$ and
 S never occurs in the right-hand side of a rule.

Repetition: Regular Grammars

Definition (Regular Grammars)

A **regular grammar** is a 4-tuple $\langle V, \Sigma, R, S \rangle$ with

- V finite set of variables (nonterminal symbols)
- Σ finite alphabet of terminal symbols with $V \cap \Sigma = \emptyset$
- $R \subseteq (V \times (\Sigma \cup \Sigma V)) \cup \{\langle S, \varepsilon \rangle\}$ finite set of rules
- if $S \rightarrow \varepsilon \in R$, there is no $X \in V, y \in \Sigma$ with $X \rightarrow yS \in R$
- $S \in V$ start variable.

Rule $X \rightarrow \varepsilon$ is only allowed if $X = S$ and
 S never occurs in the right-hand side of a rule.

How restrictive is this?

Our Plan

We are going to show that every grammar with rules

$$R \subseteq V \times (\Sigma \cup \Sigma V \cup \varepsilon)$$

generates a regular language.

Question



This is much simpler!
Why don't we define
regular languages
via such grammars?

Question

Both variants (restricting the occurrence of ε on the right-hand side of rules or not) characterize exactly the regular languages.



In the following situations, which variant would you prefer?

- You want to prove something for all regular languages.
- You want to specify a grammar to establish that a certain language is regular.
- You want to write an algorithm that takes a grammar for a regular language as input.

Our Plan

We are going to show that every grammar with rules

$$R \subseteq V \times (\Sigma \cup \Sigma V \cup \varepsilon)$$

generates a regular language.

- The proof will be **constructive**, i. e. it will tell us how to construct a regular grammar for a language that is given by such a more general grammar.
- Two steps:
 - ① Eliminate the start variable from the right-hand side of rules.
 - ② Eliminate forbidden occurrences of ε .

Start Variable in Right-Hand Side of Rules

For every type-0 language L there is a grammar where the start variable does not occur on the right-hand side of any rule.

Theorem

For every grammar $G = \langle V, \Sigma, R, S \rangle$ there is a grammar

$G' = \langle V', \Sigma, R', S \rangle$ with rules

$R' \subseteq (V' \cup \Sigma)^ V' (V' \cup \Sigma)^* \times (V' \setminus \{S\} \cup \Sigma)^*$ such that*

$\mathcal{L}(G) = \mathcal{L}(G')$.

Note: this theorem is true for **all** grammars.

Start Variable in Right-Hand Side of Rules: Example

Before we prove the theorem, let's illustrate its idea.

Consider $G = \langle \{S, X\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$bS \rightarrow \varepsilon$$

$$S \rightarrow XabS$$

$$bX \rightarrow aSa$$

$$X \rightarrow abc$$

Start Variable in Right-Hand Side of Rules: Example

Before we prove the theorem, let's illustrate its idea.

Consider $G = \langle \{S, X\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$bS \rightarrow \varepsilon$$

$$S \rightarrow XabS$$

$$bX \rightarrow aSa$$

$$X \rightarrow abc$$

The new grammar has all original rules except that S is replaced with a new variable S' (allowing to derive everything from S' that could originally be derived from the start variable S):

$$bS' \rightarrow \varepsilon$$

$$S' \rightarrow XabS'$$

$$bX \rightarrow aS'a$$

$$X \rightarrow abc$$

Start Variable in Right-Hand Side of Rules: Example

Before we prove the theorem, let's illustrate its idea.

Consider $G = \langle \{S, X\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$bS \rightarrow \varepsilon$$

$$S \rightarrow XabS$$

$$bX \rightarrow aSa$$

$$X \rightarrow abc$$

The new grammar has all original rules except that S is replaced with a new variable S' (allowing to derive everything from S' that could originally be derived from the start variable S):

$$bS' \rightarrow \varepsilon$$

$$S' \rightarrow XabS'$$

$$bX \rightarrow aS'a$$

$$X \rightarrow abc$$

In addition, it has rules that allow to start from the original start variable but switch to S' after the first rule application:

$$S \rightarrow XabS'$$

Start Variable in Right-Hand Side of Rules: Proof

Proof.

Let $G = \langle V, \Sigma, R, S \rangle$ be a grammar and $S' \notin V$ be a new variable. Construct rule set R' from R as follows:

- for every rule $r \in R$, add a rule r' to R' , where r' is the result of replacing all occurrences of S in r with S' .
- for every rule $S \rightarrow w \in R$, add a rule $S \rightarrow w'$ to R' , where w' is the result of replacing all occurrences of S in w with S' .

Then $\mathcal{L}(G) = \mathcal{L}(\langle V \cup \{S'\}, \Sigma, R', S \rangle)$. □

Start Variable in Right-Hand Side of Rules: Proof

Proof.

Let $G = \langle V, \Sigma, R, S \rangle$ be a grammar and $S' \notin V$ be a new variable. Construct rule set R' from R as follows:

- for every rule $r \in R$, add a rule r' to R' , where r' is the result of replacing all occurrences of S in r with S' .
- for every rule $S \rightarrow w \in R$, add a rule $S \rightarrow w'$ to R' , where w' is the result of replacing all occurrences of S in w with S' .

Then $\mathcal{L}(G) = \mathcal{L}(\langle V \cup \{S'\}, \Sigma, R', S \rangle)$. □

Note that the rules in R' are not fundamentally different from the rules in R . In particular:

- If $R \subseteq V \times (\Sigma \cup \Sigma V \cup \{\varepsilon\})$ then $R' \subseteq V' \times (\Sigma \cup \Sigma V' \cup \{\varepsilon\})$.
- If $R \subseteq V \times (V \cup \Sigma)^*$ then $R' \subseteq V' \times (V' \cup \Sigma)^*$.

Epsilon Rules

Theorem

For every grammar G with rules $R \subseteq V \times (\Sigma \cup \Sigma V \cup \{\epsilon\})$ there is a regular grammar G' with $\mathcal{L}(G) = \mathcal{L}(G')$.

Epsilon Rules: Example

Let's again first illustrate the idea.

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \quad S \rightarrow aX \quad X \rightarrow aX \quad X \rightarrow aY \quad Y \rightarrow bY \quad Y \rightarrow \varepsilon$$

Epsilon Rules: Example

Let's again first illustrate the idea.

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \quad S \rightarrow aX \quad X \rightarrow aX \quad X \rightarrow aY \quad Y \rightarrow bY \quad Y \rightarrow \varepsilon$$

- ❶ The start variable does not occur on a right-hand side. ✓

Epsilon Rules: Example

Let's again first illustrate the idea.

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \quad S \rightarrow aX \quad X \rightarrow aX \quad X \rightarrow aY \quad Y \rightarrow bY \quad Y \rightarrow \varepsilon$$

- ① The start variable does not occur on a right-hand side. ✓
- ② Determine the set of variables that can be replaced with the empty word: $V_\varepsilon = \{S, Y\}$.

Epsilon Rules: Example

Let's again first illustrate the idea.

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \quad S \rightarrow aX \quad X \rightarrow aX \quad X \rightarrow aY \quad Y \rightarrow bY \quad Y \rightarrow \varepsilon$$

- ① The start variable does not occur on a right-hand side. ✓
- ② Determine the set of variables that can be replaced with the empty word: $V_\varepsilon = \{S, Y\}$.
- ③ Eliminate forbidden rules: ~~$Y \rightarrow aY$~~ ~~$Y \rightarrow bY$~~ ~~$Y \rightarrow \varepsilon$~~

Epsilon Rules: Example

Let's again first illustrate the idea.

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \quad S \rightarrow aX \quad X \rightarrow aX \quad X \rightarrow aY \quad Y \rightarrow bY \quad Y \rightarrow \varepsilon$$

- ① The start variable does not occur on a right-hand side. ✓
- ② Determine the set of variables that can be replaced with the empty word: $V_\varepsilon = \{S, Y\}$.
- ③ Eliminate forbidden rules: ~~$Y \rightarrow aY$~~ ~~$Y \rightarrow bY$~~ ~~$Y \rightarrow \varepsilon$~~
- ④ If a variable from V_ε occurs in the right-hand side, add another rule that directly emulates a subsequent replacement with the empty word: $X \rightarrow a$ and $Y \rightarrow b$

Epsilon Rules

Theorem

For every grammar G with rules $R \subseteq V \times (\Sigma \cup \Sigma V \cup \{\varepsilon\})$ there is a regular grammar G' with $\mathcal{L}(G) = \mathcal{L}(G')$.

Proof.

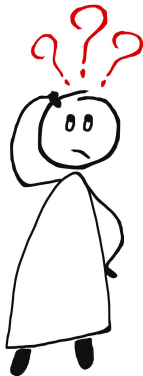
Let $G = \langle V, \Sigma, R, S \rangle$ be a grammar s.t. $R \subseteq V \times (\Sigma \cup \Sigma V \cup \{\varepsilon\})$. Use the previous proof to construct grammar $G' = \langle V', \Sigma, R', S \rangle$ s.t. $R' \subseteq V' \times (\Sigma \cup \Sigma(V' \setminus \{S\}) \cup \{\varepsilon\})$ and $\mathcal{L}(G') = \mathcal{L}(G)$.

Let $V_\varepsilon = \{A \mid A \rightarrow \varepsilon \in R'\}$.

Let R'' be the rule set that is created from R' by removing all rules of the form $A \rightarrow \varepsilon$ ($A \neq S$). Additionally, for every rule of the form $B \rightarrow xA$ with $A \in V_\varepsilon$, $B \in V'$, $x \in \Sigma$ we add a rule $B \rightarrow x$ to R'' .

Then $G'' = \langle V', \Sigma, R'', S \rangle$ is regular and $\mathcal{L}(G) = \mathcal{L}(G'')$. □

Questions



Questions?

Exercise (Slido)

Consider $G = \langle \{S, X, Y\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$S \rightarrow \varepsilon \qquad S \rightarrow aX$$

$$X \rightarrow aX \qquad X \rightarrow aY$$

$$Y \rightarrow bY \qquad Y \rightarrow \varepsilon$$

- Is G a regular grammar?
- Is $\mathcal{L}(G)$ regular?
- What is $\mathcal{L}(G)$?



Finite Automata

Languages Recognized by DFAs are Regular

Theorem

Every language recognized by a DFA is regular (type 3).

Languages Recognized by DFAs are Regular

Theorem

Every language recognized by a DFA is regular (type 3).

Proof.

Let $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ be a DFA.

We define a regular grammar G with $\mathcal{L}(G) = \mathcal{L}(M)$.

Define $G = \langle Q, \Sigma, R, q_0 \rangle$ where R contains

- a rule $q \rightarrow aq'$ for every $\delta(q, a) = q'$, and
- a rule $q \rightarrow \varepsilon$ for every $q \in F$.

(We can eliminate forbidden epsilon rules as described at the start of the chapter.)

...

Languages Recognized by DFAs are Regular

Theorem

Every language recognized by a DFA is regular (type 3).

Proof (continued).

For every $w = a_1 a_2 \dots a_n \in \Sigma^*$:

$w \in \mathcal{L}(M)$

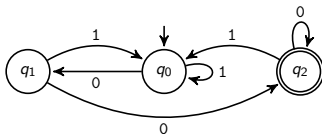
iff there is a sequence of states q'_0, q'_1, \dots, q'_n with
 $q'_0 = q_0$, $q'_n \in F$ and $\delta(q'_{i-1}, a_i) = q'_i$ for all $i \in \{1, \dots, n\}$

iff there is a sequence of variables q'_0, q'_1, \dots, q'_n with
 q'_0 is start variable and we have $q'_0 \Rightarrow a_1 q'_1 \Rightarrow a_1 a_2 q'_2 \Rightarrow$
 $\dots \Rightarrow a_1 a_2 \dots a_n q'_n \Rightarrow a_1 a_2 \dots a_n$.

iff $w \in \mathcal{L}(G)$

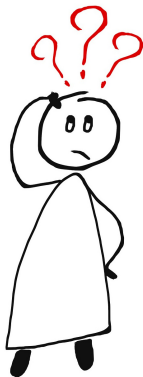


Exercise (Slido)



Specify a regular grammar that generates the language recognized by this DFA.

Questions



Questions?

Question



Is the inverse true as well:
for every regular language, is there a
DFA that recognizes it? That is, are the
languages recognized by DFAs **exactly**
the regular languages?

Question



Is the inverse true as well:
for every regular language, is there a
DFA that recognizes it? That is, are the
languages recognized by DFAs **exactly**
the regular languages?

Yes!

We will prove this via a detour.

Regular Grammars are No More Powerful than NFAs

Theorem

For every regular grammar G there is an NFA M with $\mathcal{L}(G) = \mathcal{L}(M)$.

Proof illustration:

Consider $G = \langle \{S, A, B\}, \{a, b\}, R, S \rangle$ with the following rules in R :

$$\begin{array}{llll} S \rightarrow \varepsilon & S \rightarrow aA & A \rightarrow aA & A \rightarrow aB \\ A \rightarrow a & B \rightarrow bB & B \rightarrow b & \end{array}$$

Regular Grammars are No More Powerful than NFAs

Theorem

For every regular grammar G there is an NFA M with $\mathcal{L}(G) = \mathcal{L}(M)$.

Proof.

Let $G = \langle V, \Sigma, R, S \rangle$ be a regular grammar.

Define NFA $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ with

$$Q = V \cup \{X\}, \quad X \notin V$$

$$q_0 = S$$

$$F = \begin{cases} \{S, X\} & \text{if } S \rightarrow \varepsilon \in R \\ \{X\} & \text{if } S \rightarrow \varepsilon \notin R \end{cases}$$

$$B \in \delta(A, a) \text{ if } A \rightarrow aB \in R$$

$$X \in \delta(A, a) \text{ if } A \rightarrow a \in R$$

Regular Grammars are No More Powerful than NFAs

Theorem

For every regular grammar G there is an NFA M with $\mathcal{L}(G) = \mathcal{L}(M)$.

Proof (continued).

For every $w = a_1 a_2 \dots a_n \in \Sigma^*$ with $n \geq 1$:

$w \in \mathcal{L}(G)$

iff there is a sequence on variables A_1, A_2, \dots, A_{n-1} with

$$S \Rightarrow a_1 A_1 \Rightarrow a_1 a_2 A_2 \Rightarrow \dots \Rightarrow a_1 a_2 \dots a_{n-1} A_{n-1} \Rightarrow a_1 a_2 \dots a_n.$$

iff there is a sequence of variables A_1, A_2, \dots, A_{n-1} with

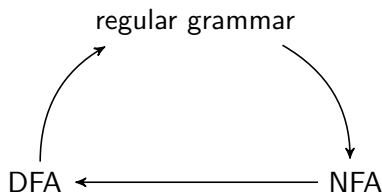
$$A_1 \in \delta(S, a_1), A_2 \in \delta(A_1, a_2), \dots, X \in \delta(A_{n-1}, a_n).$$

iff $w \in \mathcal{L}(M)$.

Case $w = \varepsilon$ is also covered because $S \in F$ iff $S \rightarrow \varepsilon \in R$.



Finite Automata and Regular Languages



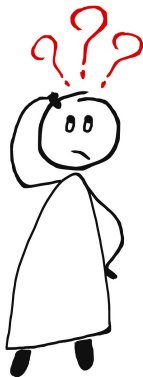
In particular, this implies:

Corollary

\mathcal{L} regular $\iff \mathcal{L}$ is recognized by a DFA.

\mathcal{L} regular $\iff \mathcal{L}$ is recognized by an NFA.

Questions



Questions?

Summary

Summary

- Regular grammars restrict the usage of ϵ in rules.
- This restriction is not necessary for the characterization of regular languages but convenient if we want to prove something for all regular languages.
- Finite automata (DFAs and NFAs) recognize exactly the regular languages.