

CSC444

October 26, 2014

Contents

1	Context-free languages	1
2	Context-free grammars	2
2.1	Definition	4
2.2	Examples	5
3	All regular languages are context-free	7

1 Context-free languages

We have seen two distinct ways of representing languages:

- *Recognizer* – Determines if a string is or is not in a language
- *Generator* – Produces strings in a language

In the remainder of the quarter we will introduce another class of languages and two different representations for the class.

As with regular languages, we will:

- Introduce the language using one of its representations.

- Introduce the other representation.
- Show that both representations capture the same set of languages.

Context-free languages have two representations:

- Context-free grammars – generator
- Pushdown automata – recognizer

We will introduce the notion using the generator, then give the recognizer, and finally show their equivalence.

2 Context-free grammars

If you have ever seen the formal specification of a programming language, then you have likely seen a context-free grammar.

Let us start with the regular expressions and see how we can generalize them to get a grammar.

Consider the regular expression $a(a^* \cup b^*)b$.

We generate a string in this language by doing the following:

- Write down an a
- Write down a string of a 's or a string of b 's, possibly of length 0
- Write down a b

So a string in the language has a beginning (the a), a middle (the string of a 's or the string of b 's) and an end (the b).

Let S be a symbol that stands for a string in this language. Let the “middle part” of the string be represented by the symbol M . Then we can express the above by the following expression:

$$S \rightarrow aMb$$

You should read the arrow as meaning “can be”.

How can we specify M ?

It is either a string of a 's or a string of b 's, possibly of length 0. Let's consider the string of a 's. (The string of b 's will obviously be very similar). Let A represent a string of a 's.

$A \rightarrow \varepsilon$ (since it may be empty)

$A \rightarrow aA$

So we can get a string of a 's by either taking the empty string or by taking an a and appending onto it a string of a 's.

The rules for the string of B 's are then:

$B \rightarrow \varepsilon$

$B \rightarrow bB$

So what is M ?

$M \rightarrow A$

$M \rightarrow B$

Sample derivations:

- $abbbb : aMb \Rightarrow aBb \Rightarrow abBb \Rightarrow abbBb \Rightarrow abbbBb \Rightarrow abbbb$
- $aaab : aMb \Rightarrow aAb \Rightarrow aaAb \Rightarrow aaaAb \Rightarrow aaab$

This is an example of a *context-free grammar*.

Why is it called context-free?

Suppose we are considering the derivation of $aaab$ above. If we are at some intermediate stage we have something like $aaAb$.

We can consider the strings surrounding the A to be the “context” for the symbol.

The rule $A \rightarrow aA$ says that we can replace any occurrence of A with aA *without taking the context of the symbol into account*.

So no matter what is surrounding it, we can simply replace it with the text on the right side of the rule.

This means that the grammar is context-free, since context does not play a role in the derivation of the string.

2.1 Definition

Notation. The symbols S , M , A and B are called *non-terminals*. Only non-terminals may appear on the left side of a rule.

Definition 2.1. A *context-free grammar* G is a quadruple (V, Σ, R, S) where:

- V is an alphabet,
- Σ (the set of terminals) is a subset of V ,
- R (the *set of rules*) is a finite subset of $(V - \Sigma) \times V^*$, and
- S (the *start symbol*) is an element of $V - \Sigma$.

Notation. • $V - \Sigma$ are called *non-terminals*

- For any $A \in V - \Sigma$ and $u \in V^*$ we write $A \rightarrow_G u$ whenever $(A, u) \in R$.
- For any strings $u, v \in V^*$ we write $u \Rightarrow_G v$ if and only if there are strings $x, y \in V^*$ and $A \in V - \Sigma$ s.t. $u = xAy$, $v = xv'y$, and $A \rightarrow_G v'$.
- \Rightarrow_G^* is the reflexive, transitive closure of \Rightarrow_G
- If the grammar is obvious we will leave off the G .

$\mathcal{L}(G)$, the language generated by some context-free grammar G , is $\{ w \in \Sigma^* \mid S \Rightarrow_G^* w \}$.

Notation. • We say that G *generates* each string in $\mathcal{L}(G)$.

- L is a *context-free language* if $L = \mathcal{L}(G)$ for some context-free grammar G
- Any sequence of the form $w_0 \Rightarrow_G w_1 \Rightarrow_G w_2 \Rightarrow_G \dots \Rightarrow_G w_n$ is called a *derivation* in G of w_n from w_0 . The derivation has n steps.

2.2 Examples

Example 2.2. Let G be the context-free grammar with $V = \{S, A, A', a\}$, $\Sigma = \{a\}$ and $R = \{S \rightarrow AAA, A \rightarrow aA', A' \rightarrow aA', A' \rightarrow \varepsilon\}$.

Sample derivations:

- $S \Rightarrow AAA \Rightarrow aA'aA'aA' \Rightarrow aaa$
- $S \Rightarrow AAA \Rightarrow aA'aA'aA' \Rightarrow aaA'aa \Rightarrow aaaa$

It is easy to recognize this language. $\mathcal{L}(G) = \{w \in \{a\}^* \mid |w| \geq 3\}$

Example 2.3. Let G be the context-free grammar with $V = \{S, a, b\}$, $\Sigma = \{a, b\}$ and $R = \{S \rightarrow aSb, S \rightarrow \varepsilon\}$.

Sample derivations:

- $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$
- $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaaSbbbb \Rightarrow aaaabbbb$

It's fairly easy to recognize that $\mathcal{L}(G) = \{a^i b^i \mid i \geq 0\}$

So our generalization of regular expressions is more powerful!

Example 2.4. Let G be given by $V = \{S, A, N, V, P\} \cup \Sigma$, $\Sigma = \{Amber, Simone, likes, furry, loud\}$ and $R = \{S \rightarrow PVP, P \rightarrow N, P \rightarrow AP, A \rightarrow fuzzy, A \rightarrow loud, N \rightarrow Simone, N \rightarrow Amber, V \rightarrow likes\}$.

G is intended to be a grammar for a part of English. S stands for sentence, A for adjective, V for verb, and P for phrase.

Sample derivations:

- Amber likes Simone
- loud Amber likes fuzzy Simone
- loud loud loud loud loud Simone likes Amber
- loud Simone likes fuzzy Amber

- loud fuzzy loud Simone likes loud loud loud loud Simone

Note. English is not a context-free grammar, so the fault is not with us.

Example 2.5. Let G be given by $V = \{S, (,)\}$, $\Sigma = \{ (,) \}$, $R = \{S \rightarrow \varepsilon, S \rightarrow SS, S \rightarrow (S)\}$.

Sample derivations:

- $S \Rightarrow SS \Rightarrow S(S) \Rightarrow S((S)) \Rightarrow S(()) \Rightarrow ()(())$
- $S \Rightarrow SS \Rightarrow (S)S \Rightarrow ()S \Rightarrow ()(S) \Rightarrow ()(())$

Note. • $\mathcal{L}(G)$ is the set of all strings with balanced parentheses. Again we have devised a way to represent a non-regular language.

- A single string can have more than one derivation. We will get back to that issue later.

Example 2.6. The grammar for the MindStorms robot.

Example 2.7. Let G be given by $V = \{S, A, B\} \cup \Sigma$, $\Sigma = \{a, b\}$ and $R = \{S \rightarrow aB, S \rightarrow bA, A \rightarrow a, A \rightarrow aS, A \rightarrow bAA, B \rightarrow b, B \rightarrow bS, B \rightarrow aBB\}$.

Sample derivations:

- $S \Rightarrow bA \Rightarrow baS \Rightarrow baaB \Rightarrow baab$
- $S \Rightarrow aB \Rightarrow aaBB \Rightarrow aabSaBB \Rightarrow aabaBabb \Rightarrow aabababb$

It is not easy to determine, but $\mathcal{L}(G) = \{w \in \{a, b\}^+ \mid w \text{ has an equal number of } a\text{'s and } b\text{'s}\}$.

We can show this by proving:

- $S \Rightarrow^* w$ if and only if w consists of an equal number of a 's and b 's
- $A \Rightarrow^* w$ if and only if w has one more a than it has b 's
- $B \Rightarrow^* w$ if and only if w has one more b than it has a 's

I will leave the proof as a suggested problem.

3 All regular languages are context-free

Theorem 3.1. *All regular languages are context-free.*

We will see this result again once we introduce pushdown automata, since it is a trivial result of the definition.

For now, however, let's give a direct construction.

Proof. Let L be a regular language and consider the $DFAM = (K, \Sigma, \delta, s, F)$ where $\mathcal{L}(M) = L$. \square Then L is also generated by the grammar $G(M) = (V, \Sigma, R, S)$ where $V = K \cup \Sigma$, $S = s$, and $R = \{ q \rightarrow ap \mid \delta(q, a) = p \} \cup \{ q \rightarrow \varepsilon \mid q \in F \}$.

Each state of the automaton becomes a non-terminal and each transition of a state q into a state p consuming input a is transformed into a rule.

Example 3.2. Let $K = \{S, A\}$, $\Sigma = \{a, b\}$, $s = S$, $F = \{S\}$, and $\delta(S, a) = A$, $\delta(S, b) = S$, $\delta(A, a) = S$, $\delta(A, b) = A$.

What language is this? The set of all strings over $\{a, b\}^*$ with an even number of a 's.

Then we can generate that language with the grammar: $V = \{S, A, a, b\}$ and $R = \{S \rightarrow \varepsilon, S \rightarrow bS, S \rightarrow aA, A \rightarrow aS, A \rightarrow bA\}$.

$S \Rightarrow bS \Rightarrow bbS \Rightarrow bbbS \Rightarrow bbbaA \Rightarrow bbbaaS \Rightarrow bbbaaS$

Claim 3.3. $\mathcal{L}(G(M)) = \mathcal{L}(M) = L$

Proof. Left as an exercise. \square