cse303
ELEMENTS OF THE THEORY OF COMPUTATION

Professor Anita Wasilewska

LECTURE 12

CHAPTER 3
SHORT REVIEW

Context-free Grammars

Finite Automata are formal **language recognizers**- they are devises that accept valid strings

Context-free Grammars are a certain type of formal **language generators**- they are devises that produce valid strings

Such a **devise** begins, when given a start symbol, to construct a string

Its operation is not completely determined from he beginning but is nevertheless limited by a finite set of rules

The process stops, and the **devise** outputs a completed string

The **language** defined by then devise is the set of all strings it can **produce**

Context-free Grammar Definition

Definition

A Context-Free Grammar is a quadruple

$$G = (V,\ \Sigma,\ R,\ S)$$

where

V is an **alphabet**

$\Sigma \subseteq V$ is a set of **terminals**

$V - \Sigma$ is the set of **nonterminals**

R is a finite set of **rules**

$$R \subseteq (V - \Sigma) \times V^*$$

$S \in V - \Sigma$ is the **start symbol**

Context-free Grammar Definition

Given a **context-free grammar**

$$G = (V, \Sigma, R, S)$$

Definition

For any $u, v \in V^*$, we define a **one step derivation**

$$u \underset{G}{\Rightarrow} v$$

of v from u as follows

$\Rightarrow_G$ if and only if there are $A, x, y, v' \in V^*$ such that

1. A $\in V - \Sigma$

2. $u = xAy$ and $v = xv'y$

2. A $\rightarrow$ v' for certain $r \in R$

Language of Context-free Grammar

Definition of the language L(G) **generated** by G

$$L(G) = \{w \in \Sigma^* : S \overset{*}{\underset{G}{\Rightarrow}} w\}$$

where $\Rightarrow_G^*$ is a transitive, reflexive closure of $\Rightarrow_G$

Given a **derivation** of $w \in \Sigma$ in G

$$S \overset{*}{\underset{G}{\Rightarrow}} w$$

We write is in detail (by definition of $\Rightarrow_G^*$) as

$$S \underset{G}{\Rightarrow} w_1 \underset{G}{\Rightarrow} w_2 \underset{G}{\Rightarrow} \dots \underset{G}{\Rightarrow} w \quad \text{for} \quad w_i \in V^*, \; w \in \Sigma$$

or in a simpler form when G is known as

$$S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \dots \Rightarrow w$$

Context-free Languages

Definition

A language L is a **context-free language** if and only if there is a context-free grammar G such that

$$L = L(G)$$

Context-free and Regular Languages

Observe that we also proved that the language
$L = \{a^n b^n : \ n \geq 0\}$ is **not regular**

Denote by RL the class of all regular languages and by

CFL the class of all contex-free languages

Hence we have

Fact 2 $RL \neq CFL$

Our next GOAL will be to prove the following

Theorem

The the class o f all regular languages is a proper subset of the class of all contex-free languages, i.e.

$$RL \subset CFL$$

Exercises

Exercise 1

Show that the regular language $L = \{a^* : \ a \in \Sigma\}$ is **context-free**

Proof By definition of **context-free** language we have to construct a CF grammar G such that

$$L = L(G) \quad \text{i.e} \quad L(G) = \{a^* : \ a \in \Sigma\}$$

Here is the grammar $G = (V, \ \Sigma, \ R, \ S)$

for $V = \{S, a\}$, $\Sigma = \{a\}$ and

$R = \{ \ S \ \rightarrow \ aS, \quad S \rightarrow e\}$

We write rules of R in a shorter way as

$$R = \{ \ S \ \rightarrow \ aS \mid e\}$$

Exercises

Here is a **derivation** in G:

$$S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaS \Rightarrow aaae$$

and we have that

$$aaaa \in L(G)$$

We prove, by induction on the length of derivation that

$$L(G) = \{a^* : \quad a \in \Sigma\}$$

Exercise 2

Show that the NOT regular language

$$L = \{ww^R : \quad w \in \{a,b\}^*\}$$

is **context-free**

Exercises

We construct a context-free grammar G such that

$$L(G) = \{ww^R : \quad w \in \{a,b\}^*\}$$

as follows

$$G = (V, \Sigma, R, S)$$

where $V = \{a, b, S\}, \ \ \Sigma = \{a, b\}$

$$R = \{S \to aSa \mid bSb \mid e\}$$

Derivation example:

$S \ \Rightarrow \ aSa \ \Rightarrow abSab \ \Rightarrow \ abbSbba \ \Rightarrow abbbba$

We prove, by induction on the length of derivation that $ww^R \in L(G)$ for any $w \in \Sigma^*$

Exercises

Remark

The set of rules

$$R = \{S \rightarrow aSa \mid aSb \mid c\}$$

defines a grammar G with the language

$$L(G) = \{wcw^R : \quad w \in \{a, b\}^*\}$$

Exercise 3

Show that the NOT regular language

$$L = \{w \in \{a, b\}^* : \quad w = w^R\}$$

is **context-free**

Exercises

We construct a context-free grammar G such that

$$L(G) = \{w \in \{a, b\}^* : \; w = w^R\}$$

as follows

$$G = (V, \Sigma, R, S)$$

where $V = \{a, b, S\}, \;\; \Sigma = \{a, b\}$

$$R = \{S \to aSa \mid bSb \mid a \mid b \mid e\}$$

Derivation example:

$$S \Rightarrow aSa \Rightarrow abSba \Rightarrow ababa$$

We check:

$$(ababa)^R = ((ab)a(ba))^R = (ba)^R a^R (ab)^R = ababa$$

Regular Grammars

Definition

A context-free grammar

$$G = (V,\ \Sigma,\ R,\ S)$$

is called **regular**, or **right-linear** if and only if

$$R \subseteq (V - \Sigma) \times \Sigma^*((V - \Sigma) \cup \{e\})$$

That is, a regular grammar is a context-free grammar such that the right-hand side of every rule contains at most one nonterminal, which if present, must be the last symbol in the string

The rules must have a form

$$A \rightarrow\ wB, \quad A \rightarrow\ w \quad \text{for any} \quad A, B \in V - \Sigma, \quad w \in \Sigma^*$$

Remark that we didn't say $A \neq B$!

Exercise 4

Given a **regular grammar** $G = (V, \Sigma, R, S)$, where $V = \{a, b, S, A\}$, $\Sigma = \{a, b\}$

$$R = \{S \to aS \mid A \mid e, \;\; A \to abA \mid a \mid b\}$$

1 Construct a finite automaton M, such that

$$L(G) = L(M)$$

Solution

We construct a non-deterministic finite automaton

$$M = (K, \Sigma, \Delta, s, F)$$

as follows:

$$K = (V - \Sigma) \cup \{f\}, \; \Sigma = \Sigma, s = S, \;\; F = \{f\}$$

$$\Delta = \{(S, a, S), (S, e, A), (S, e, f), (A, ab, A), (A, a, f), (A, b, f)\}$$

2. Write a computation of M that leads to the acceptance of the string *aaaababa*

Compare it with a derivation of the same string in *G*

Solution

The accepting computation of M is:

$$(S, aaaababa) \vdash_M (S, aaababa) \vdash_M (S, aababa) \vdash_M (S, ababa)$$

$$\vdash_M (A, ababa) \vdash_M (A, aba) \vdash_M (A, a) \vdash_M (f, e)$$

Corresponding derivation in *G* is:

$$S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaS \Rightarrow aaaA \Rightarrow aaaabA$$

$$\Rightarrow aaaababA \Rightarrow aaaababa$$

Regular and Context-free Languages

We are going to prove the following theorem that establishes the relationship between the Regular Languages and Regular Grammars

L-G Theorem

Language L is **regular** if and only if there exists a **regular grammar** G such that L = L(G)

By definition, any regular grammar is context free and hence generates a context-free language and we get that

R -CF Theorem

The the class RL of all regular languages is a proper subset of the class CFL of all context-free languages, i.e.

$$RL \subset CFL$$

L-G Theorem

Language L is **regular** if and only if there exists a **regular** grammar G such that

$$L = L(G)$$

Proof part 1

Suppose that L is **regular**; then L is accepted by a **deterministic** finite automaton

$$M = (K,\ \Sigma,\ \delta,\ s,\ F)$$

We **construct** a regular grammar G as follows

$$G = (V,\ \Sigma,\ R,\ S)$$

for $V = \Sigma \cup K, \quad S = s$

$$R = \{q \rightarrow ap : \quad \delta(q,a) = p\} \cup \{q \rightarrow e : \quad q \in F\}$$

Proof of L-G Theorem

We need now to show that L(M) = L(G)

Observe that the rules of G are designed to mimic exactly the moves of M

For any $\sigma_1, \ldots, \sigma_n \in \Sigma$ and $p_0, \ldots, p_n \in K$

$$(p_0, \sigma_1, \ldots, \sigma_n) \vdash_M (p_1, \sigma_2, \ldots, \sigma_n) \vdash_M \ldots \vdash_M (p_n, e)$$

if and only if

$$p_0 \overset{*}{\underset{G}{\Rightarrow}} \sigma_1 p_1 \overset{*}{\underset{G}{\Rightarrow}} \sigma_1\sigma_2 p_2 \ \ldots \ \overset{*}{\underset{G}{\Rightarrow}} \sigma_1\sigma_2 \ \ldots \ \sigma_n p_n$$

This is because

$\delta(q, a) = p$ if and only if $q \rightarrow ap$

Proof of L-G Theorem

We **prove** now that $L(M) \subseteq L(G)$

Suppose that $w \in L(M)$

Then for some $p \in F$

$$(s,\ w) \vdash_M{}^* (p, e)$$

Then $s \Rightarrow^* wp$, and since is also a rule $p \rightarrow e$ for $p \in F$ in G we get

$$S \overset{*}{\underset{G}{\Rightarrow}} w$$

and so $w \in L(G)$

Proof of L-G Theorem

We **prove** now that $L(G) \subseteq L(M)$

Suppose that $w \in L(G)$

Then

$$S \overset{*}{\underset{G}{\Rightarrow}} w \quad \text{that is} \quad s \overset{*}{\underset{G}{\Rightarrow}} w$$

The rule **used** at the last step of the derivation must have been of the form

$$p \rightarrow e \quad \text{for some} \quad p \in F$$

and so

$$s \overset{*}{\underset{G}{\Rightarrow}} wp \underset{G}{\Rightarrow} w$$

But then

$$(s,\ w) \vdash_M{}^* (p, e)$$

and so $w \in L(M)$ and

$$L(M) = L(G)$$

Proof of L-G Theorem

Proof part 2

Let now G be any **regular** grammar

$$G = (V, \Sigma, R, S)$$

We define a **nondeterministic** automaton M such that

$$L(M) = L(G)$$

as follows

$$M = (K, \Sigma, \Delta, s, F)$$

$$K = (V - \Sigma) \cup \{f\}$$ where f is a new element

$$s = S, \quad F = \{f\}$$

Proof of L-G Theorem

The set Δ of transitions is

$$\Delta = \{(A, w, B) : \ A \to wB \in R; \ A, B \in V - \Sigma, \ \ w \in \Sigma^*\}$$

$$\cup\{(A, w, f) : \ A \to w \in R; \ A, B \in V - \Sigma, \ \ w \in \Sigma^*\}$$

Once again, derivations are mimicked by the moves, i.e, for any

$A_1, \ldots, A_n \in V - \Sigma, \ \ w_1, \ldots w_n \in \Sigma^*$

$$A_1 \Rightarrow_G \ w_1 A_2 \ \Rightarrow_G \cdots \Rightarrow_G \ w_1 \ldots w_{n-1} A_n \Rightarrow_G \ w_1 \ldots w_n$$

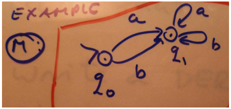
if and only if

$$(A_1, w_1 \ldots w_n) \vdash_M (A_2, w_2 \ldots w_n) \vdash_M \ \ldots \vdash_M (A_n, w_n) \vdash_M (f, e)$$

Exercises

Exercise 1

Given M defined by the **diagram** below, **construct** a regular grammar G, such that L(M) = L(G)



We follow the **proof** of L-G Theorem and we "read" the rules of G as follows

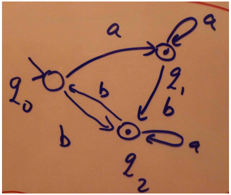
$$R = \{q_0 \rightarrow aq_1 \mid bq_1, \quad q_1 \rightarrow aq_1 \mid bq_1, \quad q_0 \rightarrow e, \quad q_1 \rightarrow e\}$$

We re-write the rules using a standard notation for nonterminals as

$$R = \{S \rightarrow aA \mid bA, \quad A \rightarrow aA \mid bA, \quad S \rightarrow e, \quad A \rightarrow e\}$$

Exercise 2

Given M defined by the **diagram** below, **construct** a regular grammar G, such that L(M) = L(G)



We "read" the rules of G as follows

$$R = \{q_0 \rightarrow aq_1 \mid bq_2, \quad q_1 \rightarrow aq_1 \mid bq_2 \mid e, \quad q_2 \rightarrow aq_2 \mid bq_0 \mid e\}$$

We re-write the rules using a standard notation for nonterminals as

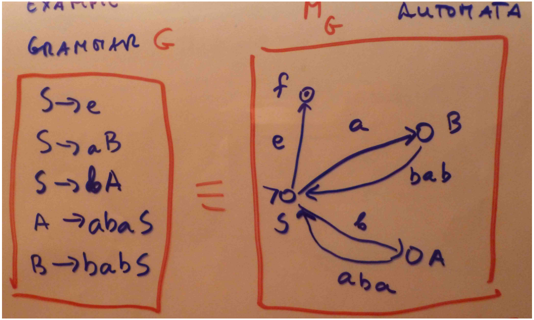
$$R = \{S \rightarrow aA \mid bB, \quad A \rightarrow aa \mid bB \mid e, \quad B \rightarrow aB \mid bS \mid e\}$$

Exercises

Exercise 3

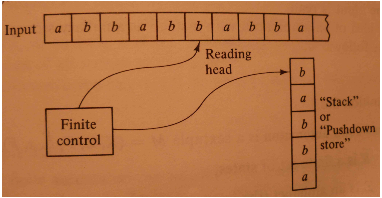
Given a grammar G defined by the set of rules, **construct** a finite automata G, such that L(M) = L(G)

Here is a **picture** depicting the pattern of such constructions



Pushdown Automata PDA

Computational Model of Pushdown Automata PDA



C1: Automata "**remembers**" what it has already read by putting it, one symbol at the time on **stack**, or **pushdown store**

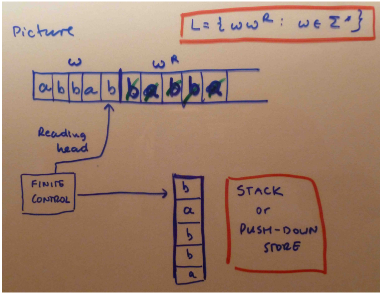
C2: It always puts symbols on the **top** of the stack

C3: symbols could be **removed** from the **top** of the stack and can be checked against the input

C4: Word is **accepted** when it has been read, **stack** is empty and automaton is in a final state

Pushdown Automata PDA

Pushdown Automata for the context-free language
$L == \{ww^R : \quad w \in \{a, b\}^*\}$



Idea: Automata will read abbab putting its reverse babba on the **stack** from down -to- up

It will **stop** nondeterministically and

start to **compare** the stack content with the rest of the input removing content of the stack

PD Automata and CF Grammars

Goal

Our goal now it to prove a theorem similar to the theorem for finite automata establishing equivalence of regular languages and finite automata, i.e. we want now to prove the following

Main Theorem

The class of languages accepted by Pushdown Automata is exactly the class of Context-free Languages

It means that we want to find best way to define Pushdown Automata in order to achieve this goal

Definition Idea

We have constructed, for any regular grammar G a finite automaton M such that L(G) = L(M) by transforming any rule

$A \rightarrow wB$ into a corresponding transition $(A, w, B) \in \Delta$ of M that said: "in state A read w and move to B"

Pushdown Automata PDA

We extend this idea to non-regular rules and pushdown automata as follows

Given a context-free grammar G and a rule

$$A \rightarrow aBb \quad \text{for} \quad a, b \in \Sigma, \quad A, B \in V - \Sigma$$

We now translate it to a corresponding transition (to be defined formally) of a PD automata M that says:

M in state A **reads** a, **puts** b on **stack** and **goes** to state B

Later, the symbols on the stack can be **removed** and **checked** agains the input when needed

Word is **accepted** when it has been read, **stack** is empty and automaton is in a final state

PDA - Mathematical Model

Definition

A Pushdown Automata is a sextuple

$$M = (K,\ \Sigma,\ \Gamma,\ \ \Delta,\ s,\ F)$$

where

K is a finite set of **states**

Σ as an alphabet of **input symbols**

Γ as an alphabet of **stack symbols**

$s \in K$ is the **initial state**

$F \subseteq K$ is the set of **final states**

Δ is a **transition relation**

$$\Delta \ \subseteq \ (K \times \Sigma^* \times \Gamma^*) \times (K \times \Gamma^*)$$

Δ is a **finite** set

We usually use different symbols for $K,\ \Sigma$, i.e. we have that $K \cap \Sigma = \emptyset$

Transition Relation

Given PDA

$$M = (K, \Sigma, \Gamma, \Delta, s, F)$$

We denote elements of **stack alphabet** by α, β, γ, with indices if necessary

Consider

$$\Delta \subseteq (K \times \Sigma^* \times \Gamma^*) \times (K \times \Gamma^*)$$

and let an element

$$((p, u, \beta), (q, \gamma)) \in \Delta$$

This means that the automaton M in the state p with β to the top of the stack,

reads u from the input,

replaces β by γ on the top of the stack, and

goes to state q

Pushdown automata is **nondeterministic**; Δ is not a function

Special Transitions

Given a transition

$$((p,\ u,\ \beta),\ (q,\ \gamma)) \in \Delta$$

Here are some spacial cases, i.e some **special transitions** that operate on the stack

Push a - **adds** symbol a **to** the top of the stack

$$((p,\ u,\ e),\ (q,\ a)) \qquad \textbf{push } a$$

Pop a - **removes** symbol a **from** the top of the stack

$$((p,\ u,\ a),\ (q,\ e)) \qquad \textbf{pop } a$$

Configuration and Transition

In order to define a notion of computation of M on an input string $w \in \Sigma^*$ we introduce, as always, a notion of a **configuration** and **transition** relation

A **configuration** is any tuple

$$(q, w, \gamma) \in K \times \Sigma^* \times \Gamma^*$$

where $q \in K$ represents a current state of M and $w \in \Sigma^*$ is unread part of the input, and γ is a content of the stack read top-down

The **transition relation** acts between two **configurations** and hence $\vdash_M$ is a certain binary relation defined on $K \times \Sigma^* \times \Gamma^*$, i.e.

$$\vdash_M \subseteq (K \times \Sigma^* \times \Gamma^*)^2$$

Formal definition follows

Transition Relation

Given

$$M = (K, \Sigma, \Gamma, \Delta, s, F))$$

Transition relation

Definition

For any $p, q \in K, u, x \in \Sigma^*, \alpha, \beta, \gamma$

$$(p, ux, \beta\alpha) \vdash_M (q, x, \gamma\alpha)$$

if and only if

$$((p, u, \beta), (q, \gamma)) \in \Delta$$

Language L(M)

We **denote** as usual, the reflexive, transitive closure of $\vdash_M$ denoted by $\vdash_M{}^*$ and define

Definition

$$L(M) = \{w \in \Sigma^* : \ (s,\ w,\ e) \vdash_M{}^* (p,\ e,\ ,e) \ \text{ for certain } p \in F\}$$

M **accepts** $w \in \Sigma^*$ if and only if $w \in L(M)$

In plain English:

$(s,\ w,\ e)$ means:

start with w and empty **stack**

$(p,\ e,\ e)$ for certain $p \in F$ means:

finish in a final state after reading w and emptying all of the **stack**

Pushdown and Finite Automata)

Theorem

The class **FA** of finite automata is a proper subset of the class **PDA** of pushdown automata, i.e.

$$\textbf{FA} \subset \textbf{PDA}$$

Proof

We show that every FA automaton is a PDA automaton that operates on an empty stack

Given a FA automaton $M = (K, \Sigma, \delta, s, F)$

We construct PDA automaton

$$M' = (K, \Sigma, \Gamma, \Delta', s, F))$$

where $\Gamma = \emptyset$ and

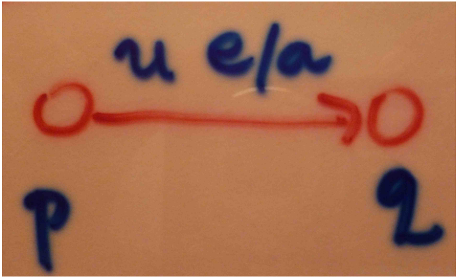
$$\Delta' = \{((p, u, e), (q, e) : \quad (p, u, q) \in \Delta\}$$

Obviously, L(M) = L(M') and hence we proved that

$$M \approx M'$$

State Diagrams for Pushdown Automata

Diagram



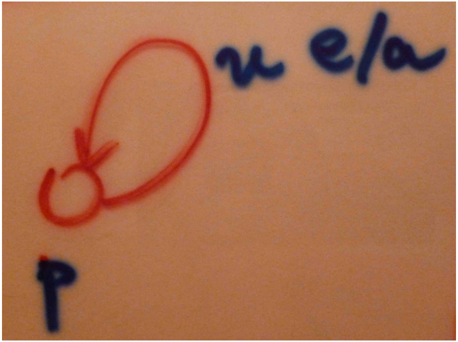
$((p, u, e), (q, a)) \in \Delta$ **push** a

M in state p

1. reads u
2. pushes a on the top of the stack
3. goes to the state g

State Diagrams for Pushdown Automata

Diagram



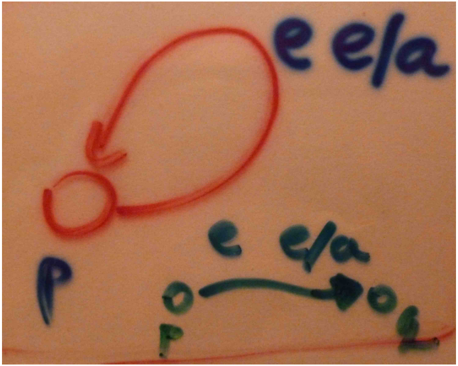
M pushes a with no change of state

In state p

1. reads u
2. pushes a on the top of the stack
3. goes to the state p

State Diagrams for Pushdown Automata

Diagram



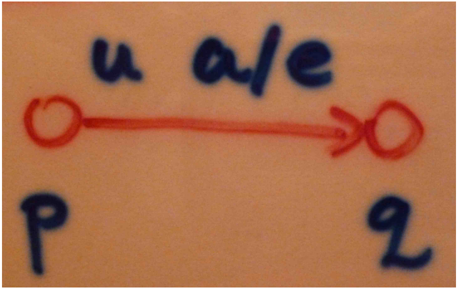
M pushes a with no change of state, reading nothing

In state p

1. reads e
2. pushes a on the top of the stack
3. goes to the state p OR goes to the state q

State Diagrams for Pushdown Automata

Diagram



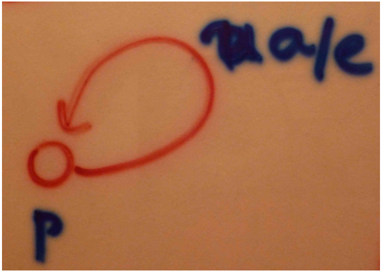
$((p, u, a), (q, e)) \in \Delta$ **pop** a

M in state p

1. reads u
2. pops a from the top of the stack
3. goes to the state q

State Diagrams for Pushdown Automata

Diagram



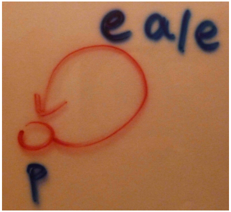
M pushes a with no change of state

In state p

1. reads u
2. pops a from the top of the stack
3. goes to the state p

State Diagrams for Pushdown Automata

Diagram



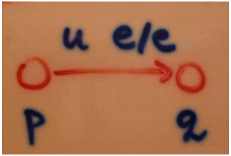
M pushes a with no change of state, reading nothing

In state p

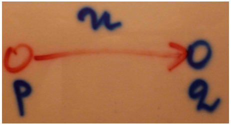
1. reads e
2. pushes a on the top of the stack
3. goes to the state p

State Diagrams for Pushdown Automata

Diagram of PD M'



that imitates the FA colorred M

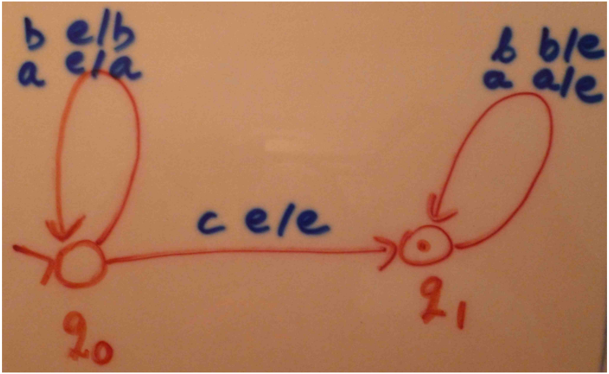


Theorem: Any FA automaton is a PD automaton

Exercise

Exercise

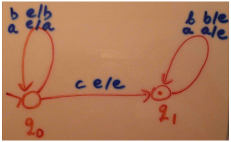
Diagram of M



Write components of M and find its language L(M)

Exercise Solution

Diagram of M



Δ components are

$((q_0, a, e), (q_0, a))$ - **push** a

$((q_0, b, e), (q_0, b))$ - **push** b

$((q_0, c, e), (q_1, e))$ - **switches** to **final** q_1 when sees c

$((q_1, a, a), (q_1, e))$ - **compares** and **pop** a

$((q_1, b, b), (q_1, e))$ - **compares** and **pop** b

$$L(M) = \{wcw^R : \quad w \in \{a, b\}^*\}$$

CHAPTER 3
PART 3: Pushdown automata and context -free grammars

Main Theorem

We are going to show now that the PD automaton is exactly what is needed to accept arbitrary context-free languages, i.e. we are going to prove the following

Main Theorem

The class of languages **accepted** by PD automata is exactly the class of context-free languages

Proof

We break the proof into two parts

Lemma 1

Each context free language is **accepted** by **some** PD automaton

Lemma 2

If a language is **accepted** by a PD automaton, it is a context free language

Proof of Lemma 1

We prove here only first Lemma; the poof of the second one is very complicated and we haven't yet covered all material needed to carry it on.

It is included in the Book on pages 139 - 142

Lemma 1

Each context free language is **accepted** by **some** PD automaton

Proof

Let $G = (V, \Sigma, R, S)$ be a context-free grammar; we must construct a PD automaton M, such that L(G) = L(M)

M we construct has only two states p and q and

M remains in state q after its first move

M uses V, the set of grammar terminals and nonterminals as its stack alphabet

Proof of Lemma 1

Given context-free grammar the $G = (V, \Sigma, R, S)$

We **define** corresponding PD automaton as

$$M = (K = \{p, q\}, \Sigma, \Gamma = V, \Delta, p, \{q\})$$

where Δ contains the following transitions

1. ((p, e, e), (q, S))

2. ((q, e, A), (q, x)) - for each rule $A \rightarrow x$ in R

3. ((q, c, c), (q, e)) - for each $c \in \Sigma$

The PD automaton M **starts** operation by **pushing** grammar start symbol S on empty its initially store and entering state q (transition **1.**)

Proof of Lemma 1

Remaining Δ transitions are

1. ((p, e, e), (q, S))

2. ((q, e, A), (q, x)) - for each rule $A \rightarrow x$ in R

3. $((q, \sigma, \sigma), (q, e))$ - for each $\sigma \in \Sigma$

M on each subsequent step

either replaces the topmost **nonterminal** symbol A on the stack by the right side x of some rule $A \rightarrow x$ in R (transition of type **2.**)

or M **pops** the topmost symbol of the stock provided it matches the input symbol (transition of type **3.**)

Proof of Lemma 1

The transitions of M are designed so that the stack during the **accepting** computation mimics a leftmost grammar **derivation** of the input string

M intermittently carries out a step of such derivation on the **stack** and

between such steps it **pops** from the stack any terminal symbols that match the symbols read from the input

Popping the terminals exposes in turn the leftmost **nonterminal**, so that the process can continue until the input **is read** and the stack is empty

All these steps are carried while M is in the **final** state, hence we get that the input word is **accepted**

We conduct the **formal proof** on induction of the length of derivation and computation

Example

Example of the construction of the proof of **Lemma 1**

Let G be such that

$$L(G) = \{wcw^R : \quad w \in \{a, b\}^*\}$$

i.e. G is as follows

$$G = (V,\ \Sigma,\ R,\ S)$$

where $V = \{a, b, c, S\},\ \ \Sigma = \{a, b, c\}$

$$R = \{S \to aSa\ | bSb\ |\ c\}$$

The corresponding PD automaton is

$$M = (K = \{p, q\},\ \ \Sigma = \{a, b, c\},\ \ \Gamma = \{a, b, c, S\},\ \ \Delta,\ p,\ \{q\})$$

with Δ corresponding to rules of G , i.e.

Example

Δ transitions corresponding to rules of G are

$$
\begin{aligned}
\Delta = \{&((p,e,e),(q,S)), && (T1)\\
&((q,e,S),(q,aSa)), && (T2)\\
&((q,e,S),(q,bSb)), && (T3)\\
&((q,e,S),(q,c)), && (T4)\\
&((q,a,a),(q,e)), && (T5)\\
&((q,b,b),(q,e)), && (T6)\\
&((q,c,c),(q,e))\} && (T7).
\end{aligned}
$$

Example

The word *abbcbba* $\in L(M)$

Here is a computations accepting abbcbba

State	Unread Input	Stack	Transition
p	$abbcbba$	e	-
q	$abbcbba$	S	T1
q	$abbcbba$	aSa	T2
q	$bbcbba$	Sa	T5
q	$bbcbba$	$bSba$	T3
q	$bcbba$	Sba	T6
q	$bcbba$	$bSbba$	T3
q	$cbba$	$Sbba$	T6
q	$cbba$	$cbba$	T4
q	bba	bba	T7
q	ba	ba	T6
q	a	a	T6
q	e	e	T5

Languages that are and are not Context- free

Closure Properties

Closure Theorem 1

The context-free languages are **closed** under union, concatenation, and Kleene star

Closure Theorem 2

The context-free languages are **not closed** under intersection and complementation

Closure Theorem 3

The **intersection** of a context-free language with a regular language is a context-free language

Pumping Lemma

Let G be a context-free grammar

Then there is a number K, depending on G, such that

any word $w \in L(G)$ of length greater than K

can be re-written as

$$w = uvxyz \quad \text{for} \quad v \neq e \quad \text{or} \quad y \neq e$$

and for any $n \geq 0$

$$uv^n xy^n z \in L(G)$$

Not Context-free Languages

We use the **Pumping Lemma** to prove the following

Theorem 4

The language

$$L = \{a^n b^n c^n : \quad n \geq 0\}$$

is NOT context-free

Theorem 5

The following languages are NOT context-free

$$L_1 = \{a^i b^j a^i b^j : \quad i, j \geq 0\}$$

$$L_2 = \{a^p : \quad \text{p is prime}\}$$

$$L_3 = \{a^{n^2} : \quad n \geq 0\}$$

$$L_4 = \{www : \quad w \in \{a, b\}^*\}$$

Power of Pumping Lemma

We use the **Pumping Lemma** to prove that MANY languages are NOT context-free

Unfortunately, there are very simple non-context-free languages which **cannot** be shown not to be context-free by a direct application of the **Pumping Lemma**

For example, we use a theorem called **Parikh Theorem** to show that

$$L = \{a^m b^n : \text{ either } m > n, \text{ or } m \text{ is prime and } n \geq m\}$$

is NOT context-free and we **can't prove** it by the use of **Pumping Lemma**

We also use **Parikh Theorem** to show the following interesting property of contex-free languages

Theorem 6

Every contex-free language over a one-symbol alphabet is **regular**

Context-free/ NOT Context-free Languages

Exercise

Prove that the language

$$L = \{ww : \quad w \in \{a, b\}^*\}$$

is NOT context-free

Hint

We know that

$$L_1 = \{a^i b^j a^i b^j : \quad i, j \geq 0\}$$

is NOT context-free

Solution

Assume that

$$L = \{ww : \ w \in \{a,b\}^*\}$$

is context-free; then the language

$$L \cap a^*b^*a^*b^*$$

is context-free by

Theorem 3

The **intersection** of a context-free language with a regular language is a context-free language

But

$$\{ww : \ w \in \{a,b\}^*\} \cap a^*b^*a^*b^* = \{a^ib^ja^ib^j : \quad i,j \geq 0\}$$

which we know to be NOT context-free

Contradiction

Some YES/NO questions

Q1

The set of **terminals** in a context free grammar G is a subset of the **alphabet** of G

Q2

$L(G) = \{w \in V : \; S \Rightarrow^*{}_G \, w\}$

Q3

A language $L \subseteq \Sigma^*$ is **context-free** if and only if there is a grammar G, such that $L = L(G)$

Q4

Any regular language is context-free

ANSWERS to YES/NO questions

Q1

The set of **terminals** in a context free grammar G is a subset of the **alphabet** of G

YES

By definition : $\Sigma \subseteq V$ for Σ terminals and V alphabet of any context free grammar G

Q2

$L(G) = \{w \in V : \; S \Rightarrow^*{}_G \; w\}$

NO Must be that $w \in \Sigma^*$

Q3

A language $L \subseteq \Sigma^*$ is **context-free** if and only if there is a grammar G, such that $L = L(G)$

NO It os true only when G is a context -free grammar

ANSWERS to YES/NO questions

Q3

A language $L \subseteq \Sigma^*$ is **context-free** if and only if there is a grammar G, such that $L = L(G)$

NO

It is true only when G is a context -free grammar

Q4

Any regular language is context-free

YES

Regular languages are generated by regular grammars, that are also context-free

Some YES/NO questions

Q5

The language $L = \{w \in \{a,b\}^* : w = w^R\}$ is **context-free**

Q6

Any regular language is accepted by a pushdown automaton

Q7

Context-free languages are **closed** under intersection

Q8

The **union** of a context-free language and regular language is a context-free language

Q9

Every **subset** of a regular language is a language

Q10

Any regular language is accepted by some PD automata

ANSWERS to YES/NO questions

Q5

The language $L = \{w \in \{a,b\}^* : w = w^R\}$ is **context-free**

YES

G with the rules:

$$S \rightarrow aSa|bSb|a|b|e$$

is such that $L = L(G)$

Q6

Any regular language is accepted by a pushdown automaton

YES

By FA Main Theorem, any regular language is accepted by a certain FA, and any finite automata is a pushdown automata operating on an empty stock

ANSWERS to YES/NO questions

Q7

Context-free languages are **closed** under intersection

NO

Take $L_1, \ L_2$ such that

$$L_1 = \{a^n b^n c^m : \ n, m \geq 0\} \text{ and } L_2 = \{a^m b^n c^n : \ n, m \geq 0\}$$

Both $L_1, \ L_2$ are CF languages and we get

$$L_1 \cap L_2 = \{a^n b^n c^n : \ n \geq 0\}$$

that is **not** CF

ANSWERS to YES/NO questions

Q8

The **union** of a context-free language and regular language is a context-free language

YES Regular language is also a context free language and context free languages are **closed** under union

Q9

Every subset of a regular language is a language

YES A subset of a set is a set

Q10

Any regular language is accepted by some PD automata

YES

Any regular language is accepted by a FA and a finite automaton is a PD automaton (that never operates on the stock)