Grammars and Pumping Lemmas

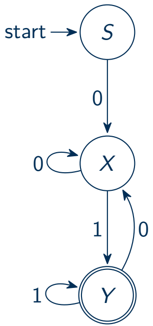
Dr. Chuck Rocca
roccac@wcsu.edu
http://sites.wcsu.edu/roccac

WESTERN CONNECTICUT STATE UNIVERSITY
MACRICOSTAS SCHOOL OF ARTS & SCIENCES

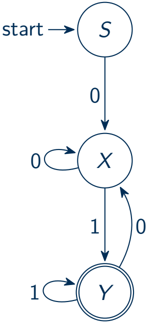
Table of Contents

Table of Contents

Finite Automaton

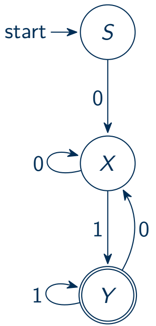


Finite Automaton



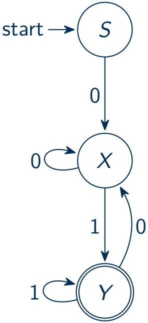
$$S \rightarrow 0X$$

Finite Automaton



$$S \rightarrow 0X$$
$$X \rightarrow 0X|1Y$$

Finite Automaton



$$S \rightarrow 0X$$
$$X \rightarrow 0X|1Y$$
$$Y \rightarrow 0X|1Y|\varepsilon$$

Regular Grammar

Definition (Regular Grammar)

A ***regular grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,

$$S \rightarrow 0X$$
$$X \rightarrow 0X|1Y$$
$$Y \rightarrow 0X|1Y|\varepsilon$$

Regular Grammar

Definition (Regular Grammar)

A ***regular grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,

$$S \to 0X$$
$$X \to 0X|1Y$$
$$Y \to 0X|1Y|\varepsilon$$

Regular Grammar

Definition (Regular Grammar)

A ***regular grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule associating a variable to a terminal or a terminal followed by a variable, and

$$S \to 0X$$
$$X \to 0X|1Y$$
$$Y \to 0X|1Y|\varepsilon$$

Regular Grammar

Definition (Regular Grammar)

A ***regular grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule associating a variable to a terminal or a terminal followed by a variable, and
4. $S \in V$ is a ***start variable***.

$$S \rightarrow 0X$$
$$X \rightarrow 0X|1Y$$
$$Y \rightarrow 0X|1Y|\varepsilon$$

Regular Grammar to Automaton

$$S \to aX|bY$$
$$X \to aX|bX|b$$
$$Y \to aY|bY|a$$

Regular Grammar to Automaton

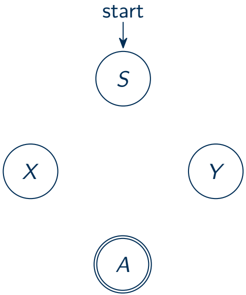
$$S \to aX|bY$$
$$X \to aX|bX|b$$
$$Y \to aY|bY|a$$

$$S \to aX|bY$$
$$X \to aX|bX|bA$$
$$Y \to aY|bY|aA$$
$$A \to \varepsilon$$

Regular Grammar to Automaton

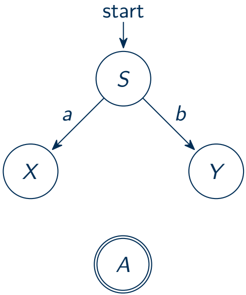
$$S \to aX|bY$$
$$X \to aX|bX|b$$
$$Y \to aY|bY|a$$

$$S \to aX|bY$$
$$X \to aX|bX|bA$$
$$Y \to aY|bY|aA$$
$$A \to \varepsilon$$



Regular Grammar to Automaton

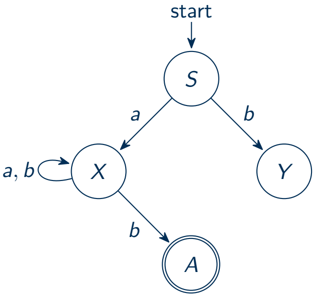
$$S \to aX|bY$$
$$X \to aX|bX|b$$
$$Y \to aY|bY|a$$



$$S \to aX|bY$$
$$X \to aX|bX|bA$$
$$Y \to aY|bY|aA$$
$$A \to \varepsilon$$

Regular Grammar to Automaton

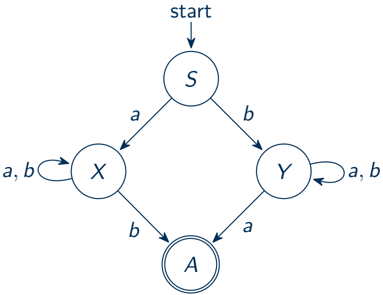
$$S \rightarrow aX|bY$$
$$X \rightarrow aX|bX|b$$
$$Y \rightarrow aY|bY|a$$



$$S \rightarrow aX|bY$$
$$X \rightarrow aX|bX|bA$$
$$Y \rightarrow aY|bY|aA$$
$$A \rightarrow \varepsilon$$

Regular Grammar to Automaton

$$S \rightarrow aX|bY$$
$$X \rightarrow aX|bX|b$$
$$Y \rightarrow aY|bY|a$$



$$S \rightarrow aX|bY$$
$$X \rightarrow aX|bX|bA$$
$$Y \rightarrow aY|bY|aA$$
$$A \rightarrow \varepsilon$$

Pumping Lemma for Regular Languages

Theorem (Pumping Lemma)

If A is a regular language, then there exists $p \in \mathbb{N}$ such that, any string $s \in A$ of length at least p may be written as $s = xyz$ such that:

1. *for each $i \geq 0$, $xy^iz \in A$,*
2. *$|y| > 0$, and*
3. *$|xy| \leq p$.*

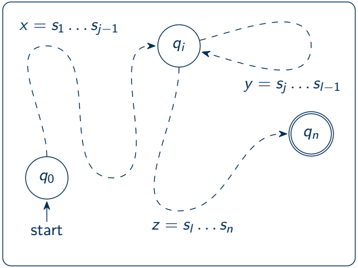


Table of Contents

Context-Free-Grammar to NPDA

$S \rightarrow aSc|bSc|\varepsilon$

Context-Free-Grammar to NPDA

$$S \rightarrow aSc|bSc|\varepsilon$$

$$S_0 \rightarrow S$$
$$S \rightarrow aU|bU|\varepsilon$$
$$U \rightarrow Sc$$

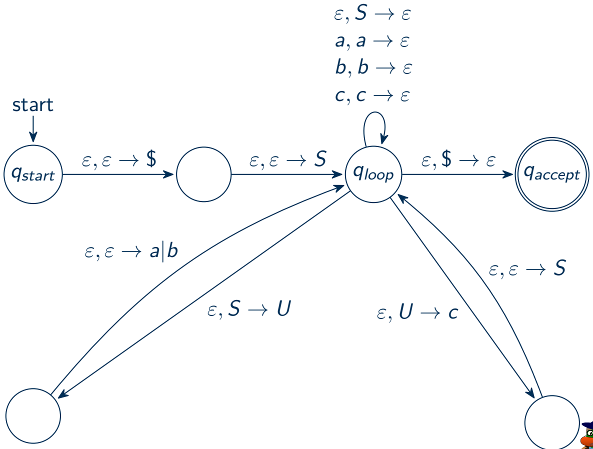
Context-Free-Grammar to NPDA

$S \rightarrow aSc|bSc|\varepsilon$

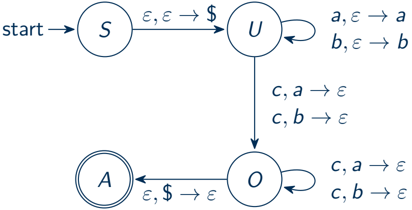
$S_0 \rightarrow S$

$S \rightarrow aU|bU|\varepsilon$

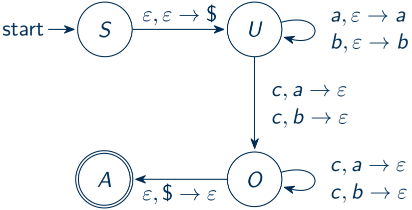
$U \rightarrow Sc$



NPDA to Context-Free-Grammar

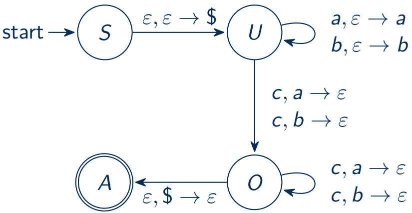


NPDA to Context-Free-Grammar



$$\delta(U, a, \varepsilon) = \{(U, a)\}$$
$$\delta(U, b, \varepsilon) = \{(U, b)\}$$
$$\delta(U, c, a) = \{(O, \varepsilon)\}$$
$$\delta(U, c, b) = \{(O, \varepsilon)\}$$
$$\delta(O, c, a) = \{(O, \varepsilon)\}$$
$$\delta(O, c, b) = \{(O, \varepsilon)\}$$

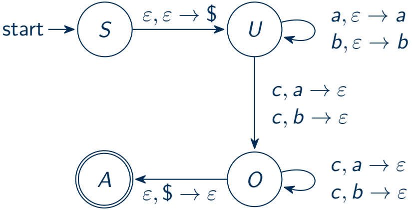
NPDA to Context-Free-Grammar



$$\delta(U, a, \varepsilon) = \{(U, a)\}$$
$$\delta(U, b, \varepsilon) = \{(U, b)\}$$
$$\delta(U, c, a) = \{(O, \varepsilon)\}$$
$$\delta(U, c, b) = \{(O, \varepsilon)\}$$
$$\delta(O, c, a) = \{(O, \varepsilon)\}$$
$$\delta(O, c, b) = \{(O, \varepsilon)\}$$

1. $S \rightarrow A_{start,accept}$
2. Add $A_{pq} \rightarrow aA_{rs}b.$, if $(r, u) \in \delta(p, a, \varepsilon)$ and $(q, \varepsilon) \in \delta(s, b, u)$
3. Add $A_{pq} \rightarrow A_{pr}A_{rq}$.
4. Add $A_{pp} \rightarrow \varepsilon$.

NPDA to Context-Free-Grammar



$$\delta(U, a, \varepsilon) = \{(U, a)\}$$
$$\delta(U, b, \varepsilon) = \{(U, b)\}$$
$$\delta(U, c, a) = \{(O, \varepsilon)\}$$
$$\delta(U, c, b) = \{(O, \varepsilon)\}$$
$$\delta(O, c, a) = \{(O, \varepsilon)\}$$
$$\delta(O, c, b) = \{(O, \varepsilon)\}$$

1. $S \to A_{start,accept}$
2. Add $A_{pq} \to aA_{rs}b.$, if $(r, u) \in \delta(p, a, \varepsilon)$ and $(q, \varepsilon) \in \delta(s, b, u)$
3. Add $A_{pq} \to A_{pr}A_{rq}$.
4. Add $A_{pp} \to \varepsilon$.

$$S \to R_{SA}$$
$$R_{SA} \to R_{UO}$$
$$R_{UO} \to aR_{UO}c | bR_{UO}c$$
$$R_{UO} \to aR_{UU}c | bR_{UU}c$$
$$R_{UU} \to \varepsilon$$

Table of Contents

Pumping Example

$$S \rightarrow A|B|C|SS|\varepsilon$$
$$A \rightarrow aBc|aCc|aSc$$
$$B \rightarrow bAc|bCc|bSc$$
$$C \rightarrow cAa|cBb|cSa|cSb$$

Consider the parse tree for

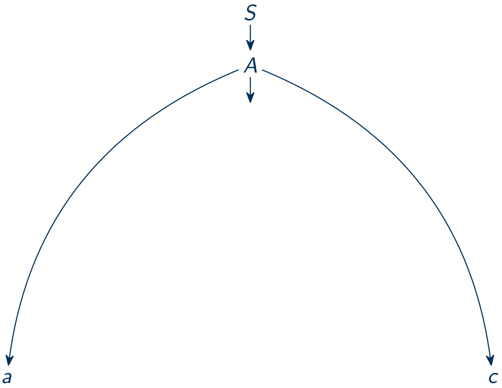
$$abcabccbcc$$

Pumping Example

$$S \to A|B|C|SS|\varepsilon$$
$$A \to aBc|aCc|aSc$$
$$B \to bAc|bCc|bSc$$
$$C \to cAa|cBb|cSa|cSb$$

Consider the parse tree for

$$abcabccbcc$$

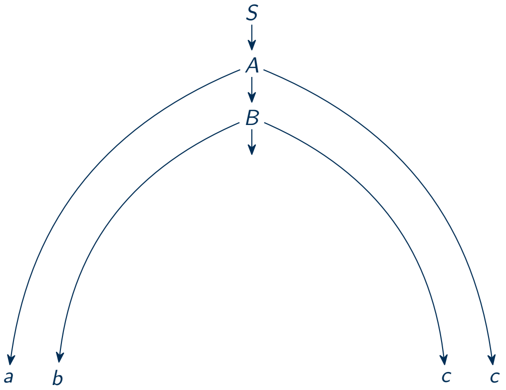


Pumping Example

$$S \rightarrow A|B|C|SS|\varepsilon$$
$$A \rightarrow aBc|aCc|aSc$$
$$B \rightarrow bAc|bCc|bSc$$
$$C \rightarrow cAa|cBb|cSa|cSb$$

Consider the parse tree for

$$abcabccbcc$$



Pumping Example

$S \to A|B|C|SS|\varepsilon$

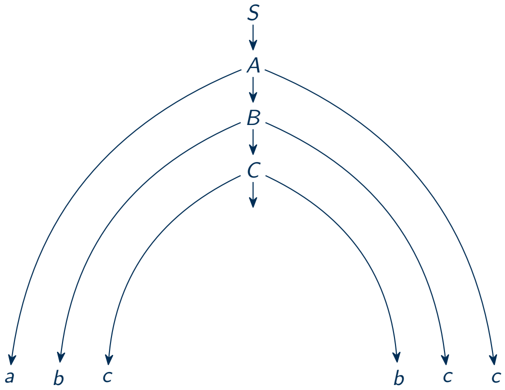
$A \to aBc|aCc|aSc$

$B \to bAc|bCc|bSc$

$C \to cAa|cBb|cSa|cSb$

Consider the parse tree for

$abcabccbcc$



Pumping Example

$S \to A|B|C|SS|\varepsilon$

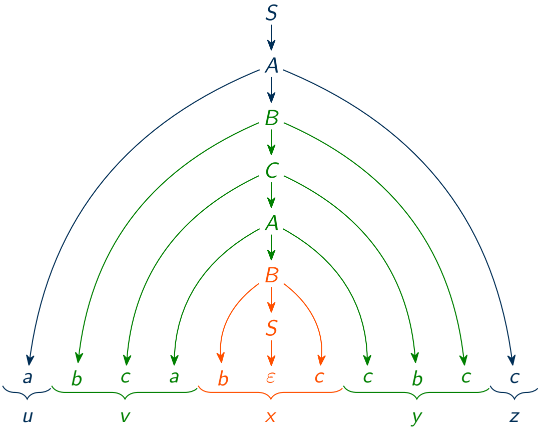
$A \to aBc|aCc|aSc$

$B \to bAc|bCc|bSc$

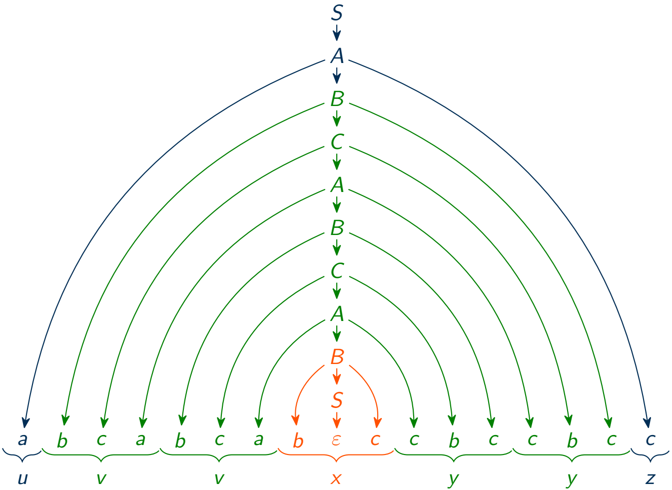
$C \to cAa|cBb|cSa|cSb$

Consider the parse tree for

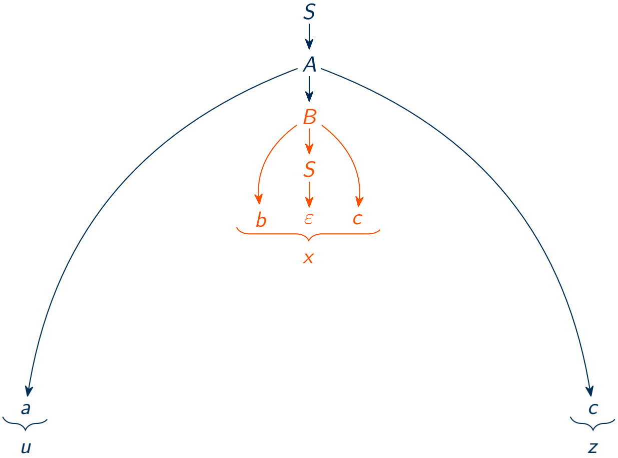
$abcabccbcc$



Pumping Up



Pumping Down



Pumping Lemma for CFL's

Theorem (Pumping Lemma for Context-Free Languages)

If A is a context-free language, then there is a number p (the pumping length) *such that, if s is any string in A of length at least p, then s may be divided into five pieces $s = uvxyz$ satisfying the conditions:*

1. *for each $i \geq 0$, $uv^ixy^iz \in A$,*
2. $|vy| > 0$, *and*
3. $|vxy| \leq p$.

General Idea (Finding p)

1. Let b equal the max. length on the right-hand of the rules (≥ 2).

General Idea (Finding p)

1. Let b equal the max. length on the right-hand of the rules (≥ 2).
2. A parse tree that is h tall has at most b^h leaves.

General Idea (Finding p)

1. Let b equal the max. length on the right-hand of the rules (≥ 2).
2. A parse tree that is h tall has at most b^h leaves.
3. Given at least $b^h + 1$ leaves, the parse tree is at least $h + 1$ tall.

General Idea (Finding p)

1. Let b equal the max. length on the right-hand of the rules (≥ 2).
2. A parse tree that is h tall has at most b^h leaves.
3. Given at least $b^h + 1$ leaves, the parse tree is at least $h + 1$ tall.
4. Let $p = b^{|V|+1}$, where $|V|$ is the number of variables.

General Idea (Finding a Repeated Variable)

5. If $s \in A$ and $|s| \geq p$, then its parse tree is at least $|V| + 1$ tall.

General Idea (Finding a Repeated Variable)

5. If $s \in A$ and $|s| \geq p$, then its parse tree is at least $|V| + 1$ tall.
6. Paths of length $|V| + 1$ or more have at least $|V| + 1$ variables.

General Idea (Finding a Repeated Variable)

5. If $s \in A$ and $|s| \geq p$, then its parse tree is at least $|V| + 1$ tall.
6. Paths of length $|V| + 1$ or more have at least $|V| + 1$ variables.
7. At least one variable is repeated, call it R.

General Idea (Finding *vxy*)

8. Choose the shortest parse tree, τ, for s.

General Idea (Finding *vxy*)

8. Choose the shortest parse tree, τ, for s.
9. Choose repeated $R's$ within $|V| + 1$ of the bottom of τ.

General Idea (Finding *vxy*)

8. Choose the shortest parse tree, τ, for s.
9. Choose repeated $R's$ within $|V| + 1$ of the bottom of τ.
10. Sub-trees of height no more than $|V| + 1$ have at most $b^{|V|+1}$ leaves.

General Idea (Finding *vxy*)

8. Choose the shortest parse tree, τ, for s.
9. Choose repeated $R's$ within $|V|+1$ of the bottom of τ.
10. Sub-trees of height no more than $|V|+1$ have at most $b^{|V|+1}$ leaves.
11. The string vxy under the first R has length at most $p = b^{|V|+1}$.

General Idea (Finding *vxy*)

8. Choose the shortest parse tree, τ, for s.
9. Choose repeated $R's$ within $|V|+1$ of the bottom of τ.
10. Sub-trees of height no more than $|V|+1$ have at most $b^{|V|+1}$ leaves.
11. The string vxy under the first R has length at most $p = b^{|V|+1}$.
12. Finally, $|vy| > 0$ because τ is minimal.

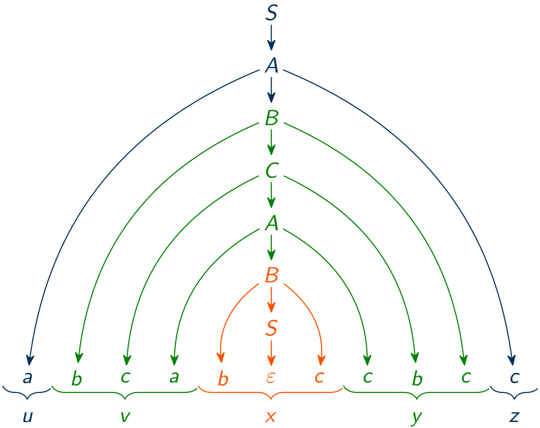
Pumping Example

$S \to A|B|C|SS|\varepsilon$

$A \to aBc|aCc|aSc$

$B \to bAc|bCc|bSc$

$C \to cAa|cBb|cSa|cSb$



Pumping Example

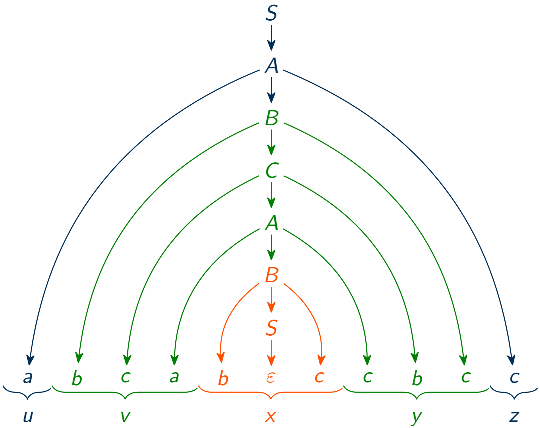
$S \to A|B|C|SS|\varepsilon$

$A \to aBc|aCc|aSc$

$B \to bAc|bCc|bSc$

$C \to cAa|cBb|cSa|cSb$

- $b = 3$



Pumping Example

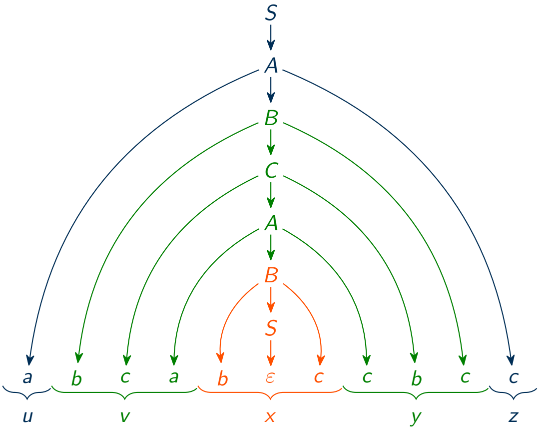
$S \rightarrow A|B|C|SS|\varepsilon$

$A \rightarrow aBc|aCc|aSc$

$B \rightarrow bAc|bCc|bSc$

$C \rightarrow cAa|cBb|cSa|cSb$

- $b = 3$
- $|V| = 4$



Pumping Example

$S \to A|B|C|SS|\varepsilon$

$A \to aBc|aCc|aSc$

$B \to bAc|bCc|bSc$

$C \to cAa|cBb|cSa|cSb$

- $b = 3$
- $|V| = 4$
- $p \leq b^{|V|+1} = 81$

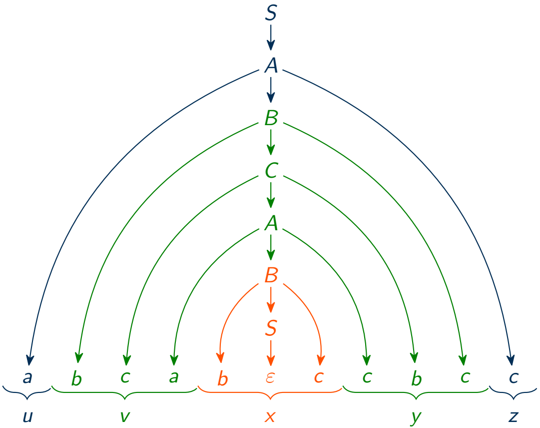


Table of Contents

Non-Context-Free Language (Example 1)

Suppose the language $B = \{a^n b^n c^n | n \geq 0\}$ has pumping length p and consider $w = a^p b^p c^p$:

$$\overbrace{\quad}^{u}\ \overbrace{\quad}^{v}\ \overbrace{\quad}^{x}\ \overbrace{\quad}^{y}\ \overbrace{\qquad\qquad\qquad\qquad}^{z}$$

$$\underbrace{aaaaa\cdots a}_{a^p}\ \underbrace{bbbbb\cdots b}_{b^p}\ \underbrace{ccccc\cdots c}_{c^p}$$

1. $uv^ixy^iz \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 1)

Suppose the language $B = \{a^n b^n c^n | n \geq 0\}$ has pumping length p and consider $w = a^p b^p c^p$:

$$\overbrace{\qquad\qquad}^{u}\ \overbrace{\quad}^{v}\ \overbrace{\quad}^{x}\ \overbrace{\quad}^{y}\ \overbrace{\qquad\qquad}^{z}$$

$$\underbrace{aaaaa\cdots a}_{a^p}\ \underbrace{bbbbb\cdots b}_{b^p}\ \underbrace{ccccc\cdots c}_{c^p}$$

1. $uv^i xy^i z \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 1)

Suppose the language $B = \{a^n b^n c^n | n \geq 0\}$ has pumping length p and consider $w = a^p b^p c^p$:

$$\underbrace{\overbrace{aaaaa \cdots a \; bb}^{u}}_{a^p}\ldots$$

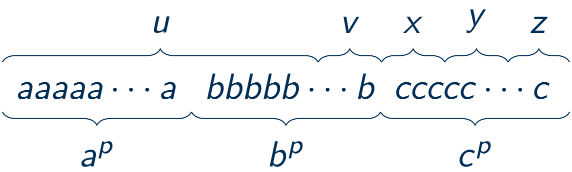
$$\underbrace{aaaaa \cdots a}_{a^p} \; \underbrace{bbbbb \cdots b}_{b^p} \; \underbrace{ccccc \cdots c}_{c^p}$$

with w split above as u, v, x, y, z.

1. $uv^ixy^iz \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 1)

Suppose the language $B = \{a^n b^n c^n | n \geq 0\}$ has pumping length p and consider $w = a^p b^p c^p$:



1. $uv^i xy^i z \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 2)

Suppose the language $D = \{ww | w \in \{01\}^*\}$ has pumping length p and consider $w = 0^p1^p0^p1^p$:

$$\overbrace{00000\cdots}^{u}\overbrace{0\;\,1}^{v}\overbrace{11}^{x}\overbrace{11}^{y}\overbrace{\cdots1\;\,00000\cdots0\;\,11111\cdots1}^{z}$$

$$\underbrace{00000\cdots0}_{0^p}\;\underbrace{11111\cdots1}_{1^p}\;\underbrace{00000\cdots0}_{0^p}\;\underbrace{11111\cdots1}_{1^p}$$

1. $uv^ixy^iz \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 2)

Suppose the language $D = \{ww | w \in \{01\}^*\}$ has pumping length p and consider $w = 0^p1^p0^p1^p$:

$$\overbrace{\underbrace{00000\cdots0}_{0^p}\ \underbrace{11111\cdots1}_{1^p}}^{u\ \ \ v}\ \overbrace{\underbrace{00000\cdots0}_{0^p}}^{x\ \ y}\ \overbrace{\underbrace{11111\cdots1}_{1^p}}^{z}$$

1. $uv^ixy^iz \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Non-Context-Free Language (Example 2)

Suppose the language $D = \{ww | w \in \{01\}^*\}$ has pumping length p and consider $w = 0^p 1^p 0^p 1^p$:

$$\overbrace{00000\cdots0 \quad 11111\cdots1 \quad 000}^{u}\overbrace{00}^{v}\overbrace{\cdots}^{x}\overbrace{0 \quad 1}^{y}\overbrace{1111\cdots1}^{z}$$

$$\underbrace{00000\cdots0}_{0^p} \quad \underbrace{11111\cdots1}_{1^p} \quad \underbrace{00000\cdots0}_{0^p} \quad \underbrace{11111\cdots1}_{1^p}$$

1. $uv^ixy^iz \in B$,
2. $|vy| > 0$, and
3. $|vxy| \leq p$.

Table of Contents

Deterministic Pushdown Automaton

Definition

A *deterministic pushdown automaton* is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, and F are all finite sets, and

1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to (Q \times \Gamma_\varepsilon) \cup \{\emptyset\}$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

The Transition function δ must satisfy the following conditions, for every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values $\delta(q, a, x)$, $\delta(q, a, \varepsilon)$, $\delta(q, \varepsilon, x)$, and $\delta(q, \varepsilon, \varepsilon)$ is not $\emptyset$

Notes from the Text

- “In contrast [to regular languages], nondeterministic pushdown automata are more powerful than their deterministic counterparts.”

Notes from the Text

- “In contrast [to regular languages], nondeterministic pushdown automata are more powerful than their deterministic counterparts.”
- “This subclass of context-free languages is relevant to practical applications, such as the design of parsers in compilers for programming languages, because the parsing problem is generally easier for DCFLs than for CFLs.”

Notes from the Text

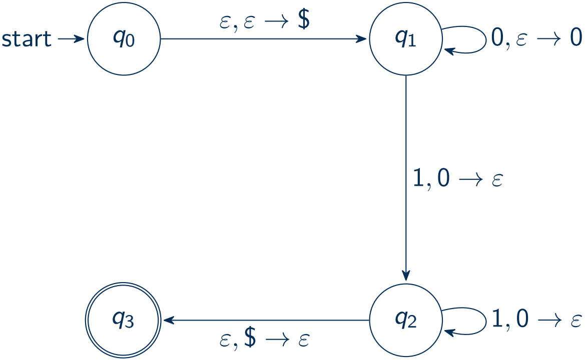
- "In contrast [to regular languages], nondeterministic pushdown automata are more powerful than their deterministic counterparts."
- "This subclass of context-free languages is relevant to practical applications, such as the design of parsers in compilers for programming languages, because the parsing problem is generally easier for DCFLs than for CFLs."
- "Arguments involving DPDAs tend to be somewhat technical in nature, ... Later material in the book doesn't depend on this section, so it may be skipped if desired."

$L = \{0^n 1^n | n \in \mathbb{N}\}$

"The Transition function δ must satisfy the following conditions, for every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values

$$\delta(q, a, x),\ (\delta(q, a, \varepsilon),\ \delta(q, \varepsilon, x),\ \textit{and}\ \delta(q, \varepsilon, \varepsilon)$$

is not $\emptyset$"

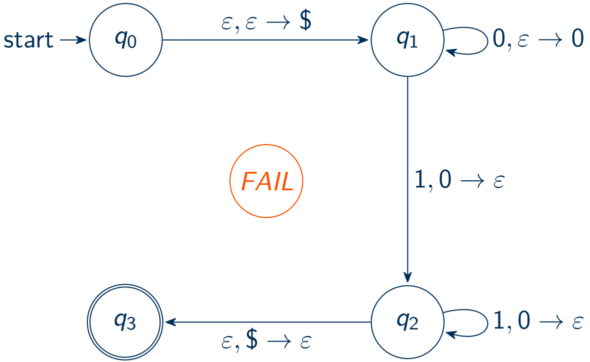


$L = \{0^n 1^n | n \in \mathbb{N}\}$

"The Transition function δ must satisfy the following conditions, for every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values

$$\delta(q, a, x),\ (\delta(q, a, \varepsilon),\ \delta(q, \varepsilon, x),\ \text{and}\ \delta(q, \varepsilon, \varepsilon)$$

is not $\emptyset$"

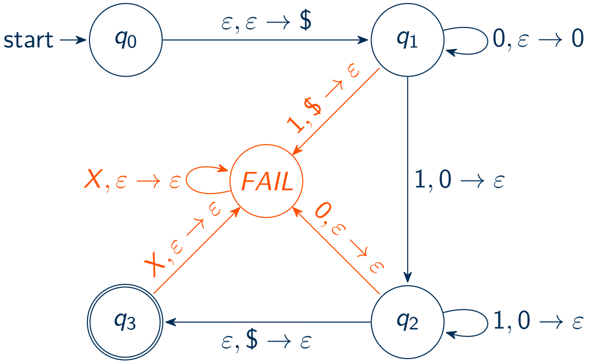


$L = \{0^n 1^n | n \in \mathbb{N}\}$

"The Transition function δ must satisfy the following conditions, for every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values

$$\delta(q, a, x),\ (\delta(q, a, \varepsilon),\ \delta(q, \varepsilon, x),\ \text{and}\ \delta(q, \varepsilon, \varepsilon)$$

is not $\emptyset$"



$L = \{0^n 1^n | n \in \mathbb{N}\}$

"The Transition function δ must satisfy the following conditions, for every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values

$$\delta(q, a, x),\ (\delta(q, a, \varepsilon),\ \delta(q, \varepsilon, x),\ \text{and}\ \delta(q, \varepsilon, \varepsilon)$$

is not $\emptyset$"

Input:	0			1			ε		
Stack:	0	\$	ε	0	\$	ε	0	\$	ε
q_0	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$(q_1, \$)$
q_1	$\emptyset$	$\emptyset$	$(q_1, 0)$	(q_2, ε)	*fail*	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
q_2	$\emptyset$	$\emptyset$	*fail*	(q_2, ε)	$\emptyset$	$\emptyset$	$\emptyset$	(q_3, ε)	$\emptyset$
q_3	$\emptyset$	$\emptyset$	*fail*	$\emptyset$	$\emptyset$	*fail*	$\emptyset$	$\emptyset$	$\emptyset$

Table of Contents

Next Class

- Turing Machines

Next Class

- Turing Machines
- Multiple Turing Machines

Next Class

- Turing Machines
- Multiple Turing Machines
- Nondeterministic Turing Machines

Next Class

- Turing Machines
- Multiple Turing Machines
- Nondeterministic Turing Machines
- Enumerators

Grammars and Pumping Lemmas

Dr. Chuck Rocca
roccac@wcsu.edu
http://sites.wcsu.edu/roccac

WESTERN CONNECTICUT STATE UNIVERSITY
MACRICOSTAS SCHOOL OF ARTS & SCIENCES