COL 352 Introduction to Automata and Theory of Computation

Nikhil Balaji

Bharti 420
Indian Institute of Technology, Delhi
nbalaji@cse.iitd.ac.in

March 1, 2023

Lecture 18: Limitations of Context-Free Grammars

Chomsky normal form

Definition

A context-free grammar is said to be in Chomsky normal form if every rule is of the form

$$A \to BC$$

$$A \to a$$

where $a \in T$, $A, B, C \in V$, neither B nor C is the start variable, i.e. start variabe does not appear on the right of any rule. Moreover, epsilon does not appear on the right of any rule except as $S \to \epsilon$.

Lemma

Any context-free grammar G can be converted into another context-free grammar G' such that $L(G) = L(G')$ and G' is in the Chomsky normal form.

Chomsky normal form

Definition

A context-free grammar is said to be in Chomsky normal form if every rule is of the form

$$A \to BC$$

$$A \to a$$

where $a \in T$, $A, B, C \in V$. Moreover, epsilon does not appear on the right of any rule except as $S \to \epsilon$.

Lemma

Any context-free grammar G can be converted into another context-free grammar G' such that $L(G) = L(G')$ and G' is in the Chomsky normal form.

Weakness of CFGs

Let us start by considering CFG in CNF for $L_{a,b} = \{a^n b^n \mid n \geq 1\}$:

Weakness of CFGs

Let us start by considering CFG in CNF for $L_{a,b} = \{a^n b^n \mid n \geq 1\}$:

$$S \to AC|AB, C \to SB, A \to a, B \to b$$

Weakness of CFGs

Let us start by considering CFG in CNF for $L_{a,b} = \{a^n b^n \mid n \geq 1\}$:

$$S \to AC|AB, C \to SB, A \to a, B \to b$$

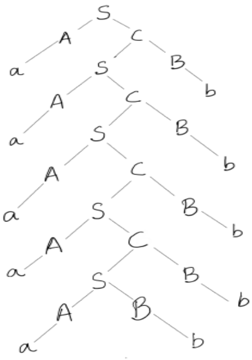
Consider a parse tree for $a^5 b^5 \in L_{a,b}$

Weakness of CFGs

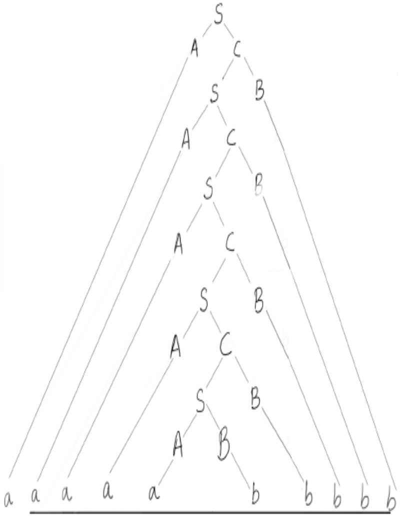
Let us start by considering CFG in CNF for $L_{a,b} = \{a^n b^n \mid n \geq 1\}$:

$$S \rightarrow AC|AB, C \rightarrow SB, A \rightarrow a, B \rightarrow b$$

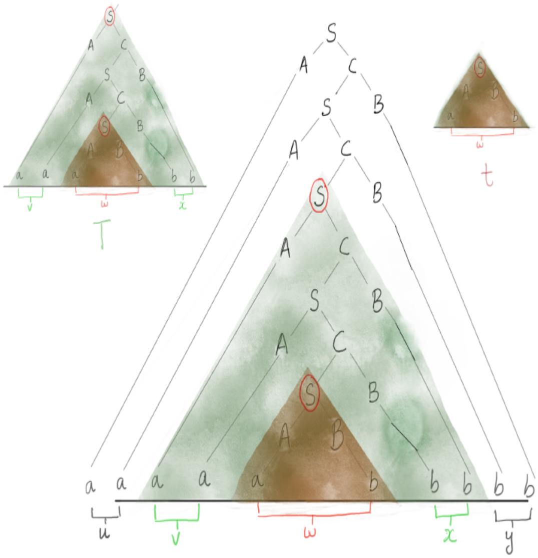
Consider a parse tree for $a^5 b^5 \in L_{a,b}$



Weakness of CFGs



Pumping Lemma for CFLs



Pumping Lemma for CFLs

Pumping Lemma for CFLs

Theorem

For every context-free language L there exists a constant k (that depends on L) such that for every string $z \in L$ of length greater or equal to k, there is an infinite family of strings belonging to L.

Pumping Lemma for CFLs

Theorem

For every context-free language L there exists a constant k (that depends on L) such that for every string $z \in L$ of length greater or equal to k, there is an infinite family of strings belonging to L.

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that

- $|vwx| \leq k$
- $vx \neq \epsilon$,
- *For all $i \geq 0$ the string $uv^iwx^iy \in L$.*

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof.

Let G be a CFG accepting L. Let b be an upper bound on the size of the RHS of any production rule of G.

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof.

Let G be a CFG accepting L. Let b be an upper bound on the size of the RHS of any production rule of G.

- Upper bound on the length of strings in L with parse-tree of height $\ell + 1 = b^\ell$

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof.

Let G be a CFG accepting L. Let b be an upper bound on the size of the RHS of any production rule of G.

- Upper bound on the length of strings in L with parse-tree of height $\ell + 1 = b^\ell$
- Let $N = |V|$ be the number of variables in G. What can we say about the strings z in L of size greater than b^N?

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof.

Let G be a CFG accepting L. Let b be an upper bound on the size of the RHS of any production rule of G.

- Upper bound on the length of strings in L with parse-tree of height $\ell + 1 = b^\ell$
- Let $N = |V|$ be the number of variables in G. What can we say about the strings z in L of size greater than b^N?
- In every parse tree of z, there is a path where a variable repeats.

Proof of pumping lemma for CFLs

Theorem

Let L be a CFL. Then there exists a constant k such that if z is a string in L of length at least k, then we can write $z = uvwxy$ such that $i)$ $|vwx| \leq k$ $ii)$ $vx \neq \epsilon$, $iii)$ For all $i \geq 0$ the string $uv^iwx^iy \in L$.

Proof.

Let G be a CFG accepting L. Let b be an upper bound on the size of the RHS of any production rule of G.

- Upper bound on the length of strings in L with parse-tree of height $\ell + 1 = b^\ell$
- Let $N = |V|$ be the number of variables in G. What can we say about the strings z in L of size greater than b^N?
- In every parse tree of z, there is a path where a variable repeats.
- Consider a minimum size parse-tree generating z, and consider a path where at least a variable repeats, and consider the last such variable.

Proof of pumping lemma for CFLs

Pumping Lemma for CFLs

Theorem (Pumping Lemma for CFLs)

$L \in \Sigma^*$ is a context-free language $\Longrightarrow$
there exists $k \geq 1$ such that
for all strings $z \in L$ with $|z| \geq k$ we have that
there exists $u, v, w, x, y \in \Sigma^*$ with $z = uvwxy, |vx| > 0$, $|vwx| \leq k$ such that
for all $i \geq 0$ we have that $uv^iwx^iy \in L$.

Theorem (Contrapositive of Pumping Lemma for CFLs)

For all $k \geq 1$ we have that
there exists strings $z \in L$ with $|z| \geq k$ such that
for all $u, v, w, x, y \in \Sigma^*$ with $z = uvwxy$, $|vx| > 0$, $|vwx| \leq k$ we have that
there exists $i \geq 0$ such that $uv^iwx^iy \notin L \Longrightarrow$
$L \in \Sigma^*$ is not a context-free language.

Games with the Demon

Games with the Demon

Theorem (Contrapositive of Pumping Lemma for CFLs)

For all $k \geq 1$ *we have that*
there exists strings $z \in L$ *with* $|z| \geq k$ *such that*
for all $u, v, w, x, y \in \Sigma^*$ *with* $z = uvwxy$, $|vx| > 0$, $|vwx| \leq k$ *we have that*
there exists $i \geq 0$ *such that* $uv^iwx^iy \notin L \implies$
$L \in \Sigma^*$ *is not a context-free language.*

Games with the Demon

Theorem (Contrapositive of Pumping Lemma for CFLs)

For all $k \geq 1$ we have that
there exists strings $z \in L$ with $|z| \geq k$ such that
for all $u, v, w, x, y \in \Sigma^*$ with $z = uvwxy$, $|vx| > 0$, $|vwx| \leq k$ we have that
there exists $i \geq 0$ such that $uv^iwx^iy \notin L \implies$
$L \in \Sigma^*$ is not a context-free language.

- Demon picks $k \geq 0$.
- You pick $z \in L$ of length at least k.
- The demon picks strings u, v, w, x, y such that $z = uvwry$, $|vr| > 0$, and $|vwx| \leq k$.
- You pick $i \geq 0$. If $uv^iwx^iy \notin L$ you win

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$
3. $L = \{ww : w \in \{0,1\}^*\}$

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$
3. $L = \{ww : w \in \{0,1\}^*\}$
4. $L = \{0^n : n \text{ is a prime number}\}$
5. $L = \{0^n : n \text{ is a perfect square}\}$
6. $L = \{0^n : n \text{ is a perfect cube}\}$

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$
3. $L = \{ww : w \in \{0,1\}^*\}$

A few examples

Prove!

Prove that the following languages are not context-free:

1. $L = \{0^n 1^n 2^n : n \geq 0\}$
2. $L = \{0^i 1^j 2^k : 0 \leq i \leq j \leq k\}$
3. $L = \{ww : w \in \{0,1\}^*\}$
4. $L = \{0^n : n \text{ is a prime number}\}$
5. $L = \{0^n : n \text{ is a perfect square}\}$
6. $L = \{0^n : n \text{ is a perfect cube}\}$

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.
- vwx cannot have both 0 and $2_{\text{why?}}$. WLOG, assume vwx has no 2.

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.
- vwx cannot have both 0 and $2_{\text{why?}}$. WLOG, assume vwx has no 2.

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 2}_{y}$$

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.
- vwx cannot have both 0 and $2_{\text{why?}}$. WLOG, assume vwx has no 2.

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 2}_{y}$$

- Consider all splits of vwx such that $|vx| > 0$.

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.
- vwx cannot have both 0 and $2_{\text{why?}}$. WLOG, assume vwx has no 2.

$$\underbrace{0\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots12\ldots2}_{y}$$

- Consider all splits of vwx such that $|vx| > 0$.
- In all splits, the length of either 0's or 1's will not be n in uwy. why?

Example 1

$$L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$$

.

- For each n, choose $z = uvwxy = 0^n 1^n 2^n \in L$.
- Consider all subwords vwx of $0^n 1^n 2^n$ such that $|vwx| \leq n$.
- vwx cannot have both 0 and $2_{\text{why?}}$. WLOG, assume vwx has no 2.

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 2}_{y}$$

- Consider all splits of vwx such that $|vx| > 0$.
- In all splits, the length of either 0's or 1's will not be n in uwy. why?
- Therefore, $uwy \notin L$. why?

Therefore L is not a CFL.

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.
- consider all the subwords vwx of $0^n 1^n 2^n 3^n$ such that $|vwx| \leq n$.

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.
- consider all the subwords vwx of $0^n 1^n 2^n 3^n$ such that $|vwx| \leq n$.
- vwx can not have more than two symbols.

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.
- consider all the subwords vwx of $0^n 1^n 2^n 3^n$ such that $|vwx| \leq n$.
- vwx can not have more than two symbols.
- There are three cases

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 23\ldots 3}_{y}$$

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.
- consider all the subwords vwx of $0^n 1^n 2^n 3^n$ such that $|vwx| \leq n$.
- vwx can not have more than two symbols.
- There are three cases

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 23\ldots 3}_{y}$$

$$\underbrace{0\ldots 01\ldots}_{u}\underbrace{\ldots 12\ldots}_{vwx}\underbrace{\ldots 23\ldots 3}_{y}$$

Example 2

$$L = \{0^n 1^m 2^n 3^m \mid n \geq 1\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 2^n 3^n \in L$.
- consider all the subwords vwx of $0^n 1^n 2^n 3^n$ such that $|vwx| \leq n$.
- vwx can not have more than two symbols.
- There are three cases

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 12\ldots 23\ldots 3}_{y}$$

$$\underbrace{0\ldots 01\ldots}_{u}\underbrace{\ldots 12\ldots}_{vwx}\underbrace{\ldots 23\ldots 3}_{y}$$

$$\underbrace{0\ldots 01\ldots 12\ldots}_{u}\underbrace{\ldots 23\ldots}_{vwx}\underbrace{\ldots 3}_{y}$$

Example 2: contd

- Now consider all splits of vwx such that $|vx| > 0$.

Example 2: contd

- Now consider all splits of vwx such that $|vx| > 0$.
- In all splits, the length of one of

$$0s, 1s, 2s, or 3s$$

will not be n in uwy and the length of the counterpart

$$2s, 3s, 0s, or 1s$$

will be n respectively.

Example 2: contd

- Now consider all splits of vwx such that $|vx| > 0$.
- In all splits, the length of one of

$$0s, 1s, 2s, or 3s$$

will not be n in uwy and the length of the counterpart

$$2s, 3s, 0s, or 1s$$

will be n respectively.
- Therefore $uv^0wx^0y \notin L$.

Therefore L is not a CFL.

Example 2: contd

- Now consider all splits of vwx such that $|vx| > 0$.
- In all splits, the length of one of

$$0s, 1s, 2s, or 3s$$

will not be n in uwy and the length of the counterpart

$$2s, 3s, 0s, or 1s$$

will be n respectively.
- Therefore $uv^0wx^0y \notin L$.

Therefore L is not a CFL.

Exercise: Is $\{0^n1^n2^m3^m \mid n \geq 1\}$ a CFL?

Example 3

$$L = \{ww \mid w \in \{0,1\}^*\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 0^n 1^n \in L$.
- consider subwords vwx of $0^n 1^n 0^n 1^n$ such that $|vwx| \leq n$ and $|vx| > 0$.
- There are three cases (v and x can be from the same block or neighboring blocks):

$$\underbrace{0\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots10\ldots01\ldots1}_{y}$$

Example 3

$$L = \{ww \mid w \in \{0,1\}^*\}$$

- For each n, choose $z = uvwxy = 0^n 1^n 0^n 1^n \in L$.
- consider subwords vwx of $0^n 1^n 0^n 1^n$ such that $|vwx| \leq n$ and $|vx| > 0$.
- There are three cases (v and x can be from the same block or neighboring blocks):

$$\underbrace{0\ldots}_{u}\underbrace{\ldots 01\ldots}_{vwx}\underbrace{\ldots 10\ldots 01\ldots 1}_{y}$$

$$\underbrace{0\ldots 01\ldots}_{u}\underbrace{\ldots 10\ldots}_{vwx}\underbrace{\ldots 01\ldots 1}_{y}$$

Example 3

$$L = \{ww \mid w \in \{0,1\}^*\}$$

- For each n, choose $z = uvwxy = 0^n1^n0^n1^n \in L$.
- consider subwords vwx of $0^n1^n0^n1^n$ such that $|vwx| \le n$ and $|vx| > 0$.
- There are three cases (v and x can be from the same block or neighboring blocks):

$$\underbrace{0\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots10\ldots01\ldots1}_{y}$$

$$\underbrace{0\ldots01\ldots}_{u}\underbrace{\ldots10\ldots}_{vwx}\underbrace{\ldots01\ldots1}_{y}$$

$$\underbrace{0\ldots01\ldots10\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots1}_{y}$$

Example 3

$$L = \{ww \mid w \in \{0,1\}^*\}$$

- For each n, choose $z = uvwxy = 0^n1^n0^n1^n \in L$.
- consider subwords vwx of $0^n1^n0^n1^n$ such that $|vwx| \le n$ and $|vx| > 0$.
- There are three cases (v and x can be from the same block or neighboring blocks):

$$\underbrace{0\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots10\ldots01\ldots1}_{y}$$

$$\underbrace{0\ldots01\ldots}_{u}\underbrace{\ldots10\ldots}_{vwx}\underbrace{\ldots01\ldots1}_{y}$$

$$\underbrace{0\ldots01\ldots10\ldots}_{u}\underbrace{\ldots01\ldots}_{vwx}\underbrace{\ldots1}_{y}$$

Equivalence of NPDAs and CFGs

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Rightarrow$).

- Assume CFG is in the Chomsky normal form.
- Push S_0 on the stack and make it the current variable.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Rightarrow$).

- Assume CFG is in the Chomsky normal form.
- Push S_0 on the stack and make it the current variable.
- Push non-deterministically one of the strings in the right hand side of the rule generated from the current variable on the stack.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Rightarrow$).

- Assume CFG is in the Chomsky normal form.
- Push S_0 on the stack and make it the current variable.
- Push non-deterministically one of the strings in the right hand side of the rule generated from the current variable on the stack.

  e.g. $A \rightarrow BC \mid DE$ then non-deterministically choose either BC or DE and depending on the choice, say it is BC, push the string BC on the stack with B on the top of the stack.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Rightarrow$).

- Assume CFG is in the Chomsky normal form.
- Push S_0 on the stack and make it the current variable.
- Push non-deterministically one of the strings in the right hand side of the rule generated from the current variable on the stack.

  e.g. $A \to BC \mid DE$ then non-deterministically choose either BC or DE and depending on the choice, say it is BC, push the string BC on the stack with B on the top of the stack.
- If the the top is a terminal, then match it off with the input bit,
- If the top of the stack is $\perp$ then accept else make that the new current variable.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Rightarrow$).

- Assume CFG is in the Chomsky normal form.
- Push S_0 on the stack and make it the current variable.
- Push non-deterministically one of the strings in the right hand side of the rule generated from the current variable on the stack.

  e.g. $A \to BC \mid DE$ then non-deterministically choose either BC or DE and depending on the choice, say it is BC, push the string BC on the stack with B on the top of the stack.
- If the the top is a terminal, then match it off with the input bit,
- If the top of the stack is $\perp$ then accept else make that the new current variable.

Repeat the above procedure. (It will either accept or loop forever.) □

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$ where

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \perp, F)$ where

- $Q = \{q\} = q_0$ (Single state)

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \perp, F)$ where

- $Q = \{q\} = q_0$ (Single state)
- $\Sigma = T$ (Input Alphabet is set of terminals)

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \perp, F)$ where

- $Q = \{q\} = q_0$ (Single state)
- $\Sigma = T$ (Input Alphabet is set of terminals)
- $\Gamma = V \cup T$ (Stack alphabet is terminals and non-terminals)

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \bot, F)$ where

- $Q = \{q\} = q_0$ (Single state)
- $\Sigma = T$ (Input Alphabet is set of terminals)
- $\Gamma = V \cup T$ (Stack alphabet is terminals and non-terminals)
- $\bot = S$ (stack bottom is start symbol of CFG)
- $F = \varnothing$ (Acceptance by empty stack)

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \perp, F)$ where

- $Q = \{q\} = q_0$ (Single state)
- $\Sigma = T$ (Input Alphabet is set of terminals)
- $\Gamma = V \cup T$ (Stack alphabet is terminals and non-terminals)
- $\perp = S$ (stack bottom is start symbol of CFG)
- $F = \varnothing$ (Acceptance by empty stack)
- δ is defined as:

$$\delta(q, \epsilon, B) := \{(q, \beta) \mid B \to \beta \text{ in } P\}$$

$$\delta(q, a, a) := \{(q, \epsilon)\}$$

CFG to NPDA

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Let $G = (V, T, P, S)$ then $A_G = (Q, \Sigma, \Gamma, \delta, q_0, \perp, F)$ where

- $Q = \{q\} = q_0$ (Single state)
- $\Sigma = T$ (Input Alphabet is set of terminals)
- $\Gamma = V \cup T$ (Stack alphabet is terminals and non-terminals)
- $\perp = S$ (stack bottom is start symbol of CFG)
- $F = \varnothing$ (Acceptance by empty stack)
- δ is defined as:

$$\delta(q, \epsilon, B) := \{(q, \beta) \mid B \to \beta \text{ in } P\}$$

$$\delta(q, a, a) := \{(q, \epsilon)\}$$

Guess production rule and push on to the stack and verify guess while popping.

NPDA to CFG

Theorem

$L = L(G)$ *for some context-free grammar* G *if and only if it is accepted by some NPDA.*

NPDA to CFG

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Leftarrow$).

- Want: Given PDA P need CFG G_P that generates all strings P accepts

NPDA to CFG

Theorem

$L = L(G)$ *for some context-free grammar* G *if and only if it is accepted by some NPDA.*

Proof $(\Leftarrow)$.

- Want: Given PDA P need CFG G_P that generates all strings P accepts
- G should generate a string if that string causes PDA to go from start to accept state.

NPDA to CFG

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Leftarrow$).

- Want: Given PDA P need CFG G_P that generates all strings P accepts
- G should generate a string if that string causes PDA to go from start to accept state.
- Idea: Design a CFG that for each pair of states p, q in P, have a variable $A_{p,q}$ which generates all strings that can take P from p (with empty stack) to q (with empty stack).

NPDA to CFG

Theorem

$L = L(G)$ for some context-free grammar G if and only if it is accepted by some NPDA.

Proof ($\Leftarrow$).

- Want: Given PDA P need CFG G_P that generates all strings P accepts
- G should generate a string if that string causes PDA to go from start to accept state.
- Idea: Design a CFG that for each pair of states p, q in P, have a variable $A_{p,q}$ which generates all strings that can take P from p (with empty stack) to q (with empty stack).
- Modify P so that
  - It has single accept state.
  - It empties its stack before accepting.
  - Each transition either pushes a symbol or pops a symbol (not both).

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ).

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \bot, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$
- $S = A_{q_0, q_F}$

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$
- $S = A_{q_0, q_F}$
- P has transitions of the following form:
  - $A_{q,q} \to \epsilon$ for all $q \in Q$;

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$
- $S = A_{q_0, q_F}$
- P has transitions of the following form:
  - $A_{q,q} \to \epsilon$ for all $q \in Q$;
  - $A_{p,q} \to A_{p,r} A_{r,q}$ for all $p, q, r \in Q$,

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$
- $S = A_{q_0, q_F}$
- P has transitions of the following form:
  - $A_{q,q} \to \epsilon$ for all $q \in Q$;
  - $A_{p,q} \to A_{p,r} A_{r,q}$ for all $p, q, r \in Q$,
  - $A_{p,q} \to a A_{r,s} b$ if $\delta(p, a, \epsilon)$ contains (r, X) and $\delta(s, b, X)$ contains (q, ϵ).

NPDA to CFG

Given a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, \perp, q_F)$ with restriction that every transition is either pushes a symbol or pops a symbol form the stack, i.e. $\delta(q, a, X)$ contains either (q_0, YX) or (q_0, ϵ). Consider the grammar $G_p = (V, T, P, S)$ such that

- $V = \{A_{p,q} : p, q \in Q\}$
- $T = \Sigma$
- $S = A_{q_0, q_F}$
- P has transitions of the following form:
  - $A_{q,q} \to \epsilon$ for all $q \in Q$;
  - $A_{p,q} \to A_{p,r} A_{r,q}$ for all $p, q, r \in Q$,
  - $A_{p,q} \to a A_{r,s} b$ if $\delta(p, a, \epsilon)$ contains (r, X) and $\delta(s, b, X)$ contains (q, ϵ).

Lemma

$L(G_p) = L(P)$.

PDA to CFG

Lemma

If $A_{p,q} \Longrightarrow^* x$ *then* x *can bring the PDA* P *from state* p *on empty stack to state* q *on empty stack.*

PDA to CFG

Lemma

If $A_{p,q} \Longrightarrow^* x$ then x can bring the PDA P from state p on empty stack to state q on empty stack.

Proof (by induction on number of steps in derivation of x from $A_{p,q}$.)

PDA to CFG

Lemma

If $A_{p,q} \Longrightarrow^* x$ then x can bring the PDA P from state p on empty stack to state q on empty stack.

Proof (by induction on number of steps in derivation of x from $A_{p,q}$.)

- **Base case.** If $A_{p,q} \Longrightarrow^* x$ in one step, then the only rule that can generate a variable free string in one step is $A_{p,p} \to \epsilon$.

PDA to CFG

Lemma

If $A_{p,q} \Longrightarrow^* x$ then x can bring the PDA P from state p on empty stack to state q on empty stack.

Proof (by induction on number of steps in derivation of x from $A_{p,q}$.)

- **Base case.** If $A_{p,q} \Longrightarrow^* x$ in one step, then the only rule that can generate a variable free string in one step is $A_{p,p} \to \epsilon$.
- **Inductive step.** If $A_{p,q} \Longrightarrow^* x$ in $n + 1$ steps. The first step in the derivation must be $A_{p,q} \to A_{p,r} A_{r,q}$ or $A_{p,q} \to a A_{r,s} b$.

PDA to CFG

Lemma

If $A_{p,q} \Longrightarrow^* x$ then x can bring the PDA P from state p on empty stack to state q on empty stack.

Proof (by induction on number of steps in derivation of x from $A_{p,q}$.)

- **Base case.** If $A_{p,q} \Longrightarrow^* x$ in one step, then the only rule that can generate a variable free string in one step is $A_{p,p} \rightarrow \epsilon$.
- **Inductive step.** If $A_{p,q} \Longrightarrow^* x$ in $n+1$ steps. The first step in the derivation must be $A_{p,q} \rightarrow A_{p,r}A_{r,q}$ or $A_{p,q} \rightarrow aA_{r,s}b$.
  - If it is $A_{p,q} \rightarrow A_{p,r}A_{r,q}$, then the string x can be broken into two parts x_1x_2 such that $A_{p,r} \Longrightarrow^* x_1$ and $A_{r,q} \Longrightarrow^* x_2$ in at most n steps. The claim easily follows in this case.
  - If it is $A_{p,q} \rightarrow aA_{r,s}b$, then the string x can be broken as ayb such that $A_{r,s} \Longrightarrow^* y$ in n steps. Notice that from p on reading a the PDA pushes a symbol X to stack, while it pops X in state s and goes to q.

□