


Lecture 7
January 22, 2025

CS21
Decidability
and
Tractability



1

NPDA, CFG equivalence

Theorem: a language L is recognized by a NPDA iff L is described by a CFG.

Must prove *two* directions:

- (\Rightarrow) L is recognized by a NPDA **implies** L is described by a CFG.
- (\Leftarrow) L is described by a CFG **implies** L is recognized by a NPDA (done last lecture)

January 22, 2025 CS21 Lecture 7 2

2

NPDA, CFG equivalence

Proof of (\Rightarrow) : L is recognized by a NPDA **implies** L is described by a CFG.

- harder direction
- first step: convert NPDA into “normal form”:
 - single accept state
 - empties stack before accepting
 - each transition *either pushes or pops* a symbol

January 22, 2025 CS21 Lecture 7 3

3

NPDA, CFG equivalence

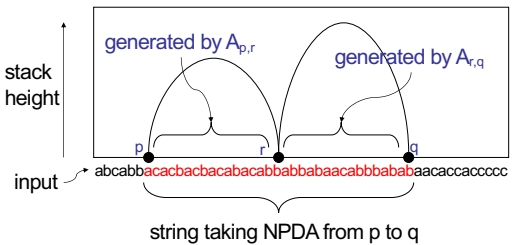
- **main idea:** non-terminal $A_{p,q}$ generates exactly the strings that take the NPDA from state p (w/ empty stack) to state q (w/ empty stack)
- then $A_{\text{start, accept}}$ generates all of the strings in the language recognized by the NPDA.

January 22, 2025 CS21 Lecture 7 4

4

NPDA, CFG equivalence

- Two possibilities to get from state p to q :



January 22, 2025 CS21 Lecture 7 5

5

NPDA, CFG equivalence

- NPDA $P = (Q, \Sigma, \Gamma, \delta, \text{start}, \{\text{accept}\})$
- CFG G :
 - non-terminals $V = \{A_{p,q} : p, q \in Q\}$
 - start variable $A_{\text{start, accept}}$
 - productions:
 - for every $p, r, q \in Q$, add the rule $A_{p,q} \rightarrow A_{p,r}A_{r,q}$

January 22, 2025 CS21 Lecture 7 6

6

NPDA, CFG equivalence

- Two possibilities to get from state p to q :

January 22, 2025 CS21 Lecture 7 7

7

NPDA, CFG equivalence

- NPDA $P = (Q, \Sigma, \Gamma, \delta, s, A_{start, accept})$
- CFG G :
 - non-terminals $V = \{A_{p,q} : p, q \in Q\}$
 - start variable $A_{start, accept}$
 - productions:
 - for every $p, r, s, q \in Q, d \in \Gamma$ and $a, b \in (\Sigma \cup \{\epsilon\})$
 - if $(r, d) \in \delta(p, a, \epsilon)$, and
 - $(q, \epsilon) \in \delta(s, b, d)$, add the rule
 - $A_{p,q} \rightarrow aA_{r,s}b$

from state p ,
read a , push d ,
move to state r

from state s ,
read b , pop d ,
move to state q

January 22, 2025 CS21 Lecture 7 8

8

NPDA, CFG equivalence

- NPDA $P = (Q, \Sigma, \Gamma, \delta, start, \{accept\})$
- CFG G :
 - non-terminals $V = \{A_{p,q} : p, q \in Q\}$
 - start variable $A_{start, accept}$
 - productions:
 - for every $p \in Q$, add the rule
 - $A_{p,p} \rightarrow \epsilon$

January 22, 2025 CS21 Lecture 7 9

9

NPDA, CFG equivalence

- two claims to verify correctness:
 - if $A_{p,q}$ generates string x , then x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack)
 - if x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack), then $A_{p,q}$ generates string x

January 22, 2025 CS21 Lecture 7 10

10

NPDA, CFG equivalence

- if $A_{p,q}$ generates string x , then x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack)
 - induction on length of derivation of x .
 - base case: 1 step derivation. must have only terminals on rhs. In G , must be production of form $A_{p,p} \rightarrow \epsilon$.

January 22, 2025 CS21 Lecture 7 11

11

NPDA, CFG equivalence

- if $A_{p,q}$ generates string x , then x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack)
 - assume true for derivations of length at most k , prove for length $k+1$.
 - verify case: $A_{p,q} \rightarrow A_{p,r}A_{r,q} \rightarrow^k x = yz$
 - verify case: $A_{p,q} \rightarrow aA_{r,s}b \rightarrow^k x = ayb$

January 22, 2025 CS21 Lecture 7 12

12

NPDA, CFG equivalence

2. if x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack), then $A_{p,q}$ generates string x

- induction on # of steps in P 's computation
- base case: 0 steps. starts and ends at same state p . only has time to read empty string ϵ .
- G contains $A_{p,p} \rightarrow \epsilon$.

January 22, 2025 CS21 Lecture 7 13

13

NPDA, CFG equivalence

2. if x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack), then $A_{p,q}$ generates string x

- induction step. assume true for computations of length at most k , prove for length $k+1$.
- if stack becomes empty sometime in the middle of the computation (at state r)
 - y is read going from state p to r $(A_{p,r} \rightarrow^* y)$
 - z is read going from state r to q $(A_{r,q} \rightarrow^* z)$
 - conclude: $A_{p,q} \rightarrow A_{p,r}A_{r,q} \rightarrow^* yz = x$

January 22, 2025 CS21 Lecture 7 14

14

NPDA, CFG equivalence

2. if x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack), then $A_{p,q}$ generates string x

- if stack becomes empty only at beginning and end of computation.
 - first step: state p to r , read a , push d
 - go from state r to s , read string y $(A_{r,s} \rightarrow^* y)$
 - last step: state s to q , read b , pop d
 - conclude: $A_{p,q} \rightarrow aA_{r,s}b \rightarrow^* ayb = x$

January 22, 2025 CS21 Lecture 7 15

15

NPDA, CFG equivalence

2. if x can take NPDA P from state p (w/ empty stack) to q (w/ empty stack), then $A_{p,q}$ generates string x

- if stack becomes empty only at beginning and end of computation.
 - first step: state p to r , read a , push d
 - go from state r to s , read string y $(A_{r,s} \rightarrow^* y)$
 - last step: state s to q , read b , pop d
 - conclude: $A_{p,q} \rightarrow aA_{r,s}b \rightarrow^* ayb = x$

January 22, 2025 CS21 Lecture 7 16

16

Pumping Lemma for CFLs

CFL Pumping Lemma: Let L be a CFL. There exists an integer p ("pumping length") for which every $w \in L$ with $|w| \geq p$ can be written as

$$w = uvxyz \quad \text{such that}$$

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

January 22, 2025 CS21 Lecture 7 17

17

CFL Pumping Lemma Example

Theorem: the following language is not context-free:

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

- Proof:
 - let p be the pumping length for L
 - choose $w = a^p b^p c^p$
 - $w = aaaa\dots abbbb\dots bcccc\dots c$
 - $w = uvxyz$, with $|vy| > 0$ and $|vxy| \leq p$.

January 22, 2025 CS21 Lecture 7 18

18

CFL Pumping Lemma Example

– possibilities:

$$w = \underbrace{aaaa\dots}_{u} \underbrace{aa}_{v} \underbrace{bbb\dots}_{x} \underbrace{bb}_{y} \underbrace{cccc\dots}_{z} c$$

(if v, y each contain only one type of symbol, then pumping on them produces a string not in the language)

January 22, 2025

CS21 Lecture 7

19

19

CFL Pumping Lemma Example

– possibilities:

$$w = \underbrace{aaaa\dots}_{u} \underbrace{ab}_{v} \underbrace{bbb\dots}_{x} \underbrace{bc}_{y} \underbrace{cccc\dots}_{z} c$$

(if v or y contain more than one type of symbol, then pumping on them might produce a string with equal numbers of a's, b's, and c's – if vy contains equal numbers of a's, b's, and c's. But they will be out of order.)

January 22, 2025

CS21 Lecture 7

20

20

CFL Pumping Lemma Example

Theorem: the following language is not context-free:

$$L = \{xx : x \in \{0,1\}^*\}$$

• Proof:

- let p be the pumping length for L
- try $w = 0^p 1 0^p 1$
- can this be pumped?

January 22, 2025

CS21 Lecture 7

21

21

CFL Pumping Lemma Example

$$L = \{xx : x \in \{0,1\}^*\}$$

- try $w = 0^{2p} 1^{2p} 0^{2p} 1^{2p}$
- $w = uvxyz$, with $|vy| > 0$ and $|vxy| \leq p$.
- case: vxy in first half.
 - then $uv^2xy^2z = 0^{2p} 1^{2p} 0^{2p} 1^{2p}$
- case: vxy in second half.
 - then $uv^2xy^2z = 0^{2p} 1^{2p} 0^{2p} 1^{2p}$
- case: vxy straddles midpoint
 - then $uv^i xy^j z = uxz = 0^{2p} 1^{i+j} 0^{2p}$ with $i \neq 2p$ or $j \neq 2p$

January 22, 2025

CS21 Lecture 7

22

22

Pumping Lemma for CFLs

CFL Pumping Lemma: Let L be a CFL. There exists an integer p (“pumping length”) for which every $w \in L$ with $|w| \geq p$ can be written as

$$w = uvxyz \quad \text{such that}$$

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

January 22, 2025

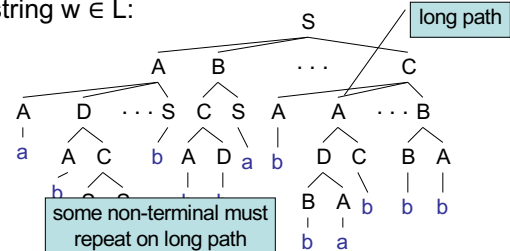
CS21 Lecture 7

23

23

CFL Pumping Lemma

Proof: consider a parse tree for a very long string $w \in L$:



January 22, 2025

CS21 Lecture 7

24

24

CFL Pumping Lemma

• Schematic proof:

January 22, 2025 CS21 Lecture 7 25

25

CFL Pumping Lemma

• Schematic proof:

January 22, 2025 CS21 Lecture 7 26

26

CFL Pumping Lemma

- how large should pumping length p be?
- need to ensure other conditions:
 - $|vy| > 0$ $|vxy| \leq p$
- $b = \max$ # symbols on rhs of any production (assume $b \geq 2$)
- if parse tree has height $\leq h$, then string generated has length $\leq b^h$ (so length $> b^h$ implies height $> h$)

January 22, 2025 CS21 Lecture 7 27

27

CFL Pumping Lemma

- let m be the # of nonterminals in the grammar
- to ensure path of length at least $m+2$, require
 - $|w| \geq p = b^{m+2}$
- since $|w| > b^{m+1}$, any parse tree for w has height $> m+1$
- let T be the **smallest** parse tree for w
- longest root-leaf path must consist of $\geq m+1$ non-terminals and 1 terminal.

January 22, 2025 CS21 Lecture 7 28

28

CFL Pumping Lemma

- must be a repeated non-terminal **A** on long path
- select a repetition among the **lowest** $m+1$ non-terminals on path.
- pictures show that for every $i \geq 0$, $uv^ixy^iz \in L$
- is $|vy| > 0$?
 - smallest parse tree T ensures
- is $|vxy| \leq p$?
 - red path has length $\leq m+2$, so $\leq b^{m+2} = p$ leaves

January 22, 2025 CS21 Lecture 7 29

29