

CS21 Decidability and Tractability

Lecture 6
January 18, 2008

January 18, 2008

CS21 Lecture 6

1

Outline

- equivalence of NPDAs and CFGs (cont.)
- Pumping Lemma for CFLs
- deterministic PDAs
- deciding CFLs

January 18, 2008

CS21 Lecture 6

2

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.

January 18, 2008

CS21 Lecture 6

3

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 18, 2008

CS21 Lecture 6

4

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.

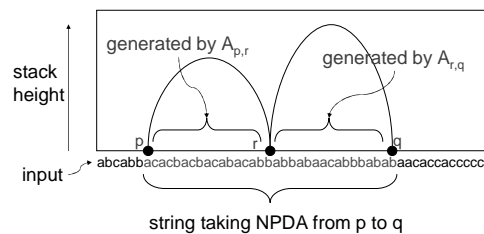
January 18, 2008

CS21 Lecture 6

5

NPDA, CFG equivalence

- Two possibilities to get from state p to q :



January 18, 2008

CS21 Lecture 6

6

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 $A_{\text{start}, \text{accept}}$
 - productions:
 - for every $p, r, q \in Q$, add the rule

$$A_{p,q} \rightarrow A_{p,r}A_{r,q}$$

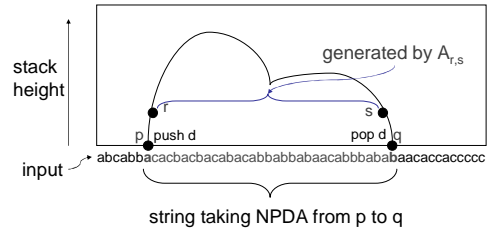
January 18, 2008

CS21 Lecture 6

7

NPDA, CFG equivalence

- Two possibilities to get from state p to q :



January 18, 2008

CS21 Lecture 6

8

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 $A_{\text{start}, \text{accept}}$
 - 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

$$A_{p,q} \rightarrow aA_{r,s}b$$

January 18, 2008

CS21 Lecture 6

9

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 $A_{\text{start}, \text{accept}}$
 - productions:
 - for every $p \in Q$, add the rule

$$A_{p,p} \rightarrow \epsilon$$

January 18, 2008

CS21 Lecture 6

10

NPDA, CFG equivalence

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

January 18, 2008

CS21 Lecture 6

11

NPDA, CFG equivalence

1. 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 18, 2008

CS21 Lecture 6

12

NPDA, CFG equivalence

1. 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 18, 2008

CS21 Lecture 6

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 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 18, 2008

CS21 Lecture 6

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
 - 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 18, 2008

CS21 Lecture 6

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 18, 2008

CS21 Lecture 6

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

$w = uvxyz$ such that

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

January 18, 2008

CS21 Lecture 6

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 $w = a^p b^p c^p$
 - $w = uvxyz$, with $|vy| > 0$ and $|vxy| \leq p$.

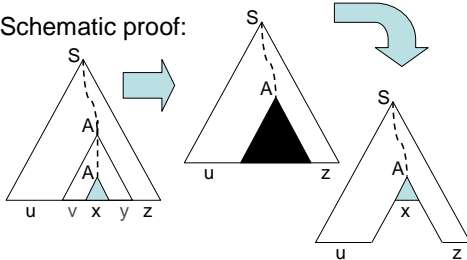
January 18, 2008

CS21 Lecture 6

18

CFL Pumping Lemma

- Schematic proof:



January 18, 2008

CS21 Lecture 6

25

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 18, 2008

CS21 Lecture 6

26

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.

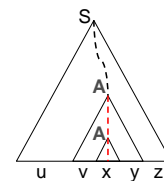
January 18, 2008

CS21 Lecture 6

27

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 18, 2008

CS21 Lecture 6

28

Deterministic PDA

- A NPDA is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where:
 - $\delta: Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \rightarrow \wp(Q \times (\Gamma \cup \{\epsilon\}))$ is a function called the transition function
- A deterministic PDA has only one option at every step:
 - for every state $q \in Q$, $a \in (\Sigma \cup \{\epsilon\})$, and $t \in \Gamma$, at most 1 element in $\delta(q, a, t)$
 - if $\delta(q, \epsilon, t)$ not empty, then $\delta(q, a, t)$ empty for all $a \in \Sigma$

January 18, 2008

CS21 Lecture 6

29

Deterministic PDA

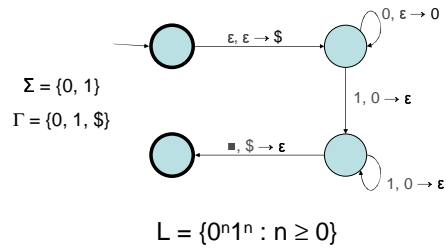
- A detail: a NPDA can “guess” the end of input string
- to be fair, we should give our deterministic machine ability to detect end of input string
 - add special symbol \blacksquare to alphabet
 - require input tape to contain $x\blacksquare$
- language recognized by a deterministic PDA is called a deterministic CFL (DCFL)

January 18, 2008

CS21 Lecture 6

30

Example deterministic PDA



January 18, 2008

CS21 Lecture 6

31

Deterministic PDA

Theorem: DCFLs are closed under complement
 (complement of L in Σ^* is $(\Sigma^* - L)$)

Proof attempt:

- swap accept/non-accept states
- problem: might enter infinite loop before reading entire string
- machine for complement must accept in these cases, and read to end of string

January 18, 2008

CS21 Lecture 6

32