CS21
Decidability and Tractability

Lecture 7

January 22, 2014

Outline

- proof of CFL pumping lemma
- deterministic PDAs
- deciding CFLs

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. $|vxy| \leq p$.

CFL Pumping Lemma Example

Theorem: the following language is not context-free:

$$L = \{a^nb^nc^n : n \geq 0\}.$$

- Proof:
  - let p be the pumping length for L
  - choose $w = a^pb^pc^p$

    w = aaaa…abbbb…bcccc…c
  - $w = uvxyz$, with $|vy| > 0$ and $|vxy| \leq p$.

CFL Pumping Lemma Example

– possibilities:

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

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

CFL Pumping Lemma Example

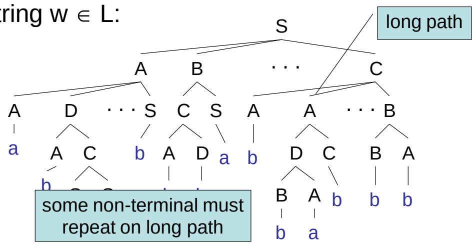
– possibilities:

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

(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.)

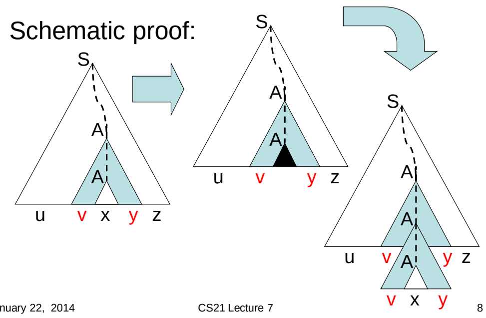
CFL Pumping Lemma

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



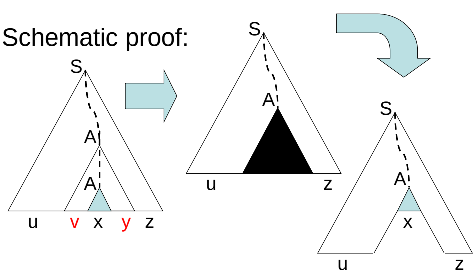
CFL Pumping Lemma

- Schematic proof:



CFL Pumping Lemma

- Schematic proof:



CFL Pumping Lemma

- how large should pumping length p be?
- need to ensure other conditions:

  $|vy| > 0$ $|vxy| \le p$

- b = max # symbols on rhs of any production (assume $b \ge 2$)
- if parse tree has height $\le h$, then string generated has length $\le b^h$ (so length $> b^h$ implies height $> h$)

CFL Pumping Lemma

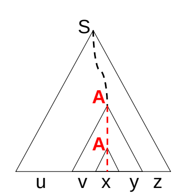
- let m be the # of nonterminals in the grammar
- to ensure path of length at least m+2, require

  $|w| \ge \mathbf{p} = b^{m+2}$

- 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 $\ge$ m+1 non-terminals and 1 terminal.

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 $i \ge 0$, $uv^ixy^iz \in L$



- is $|vy| > 0$?
  - smallest parse tree T ensures
- is $|vxy| \le p$?
  - red path has length $\le m+2$, so $\le b^{m+2} = p$ leaves

Deterministic PDA

- A NPDA is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where:
  - $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \rightarrow \wp(Q \times (\Gamma \cup \{\varepsilon\}))$ is a function called the transition function
- A deterministic PDA has only one option at every step:
  - for every state $q \in Q$, $a \in \Sigma$, and $t \in (\Gamma \cup \{\varepsilon\})$, exactly 1 element in $\delta(q, a, t)$, or
  - exactly 1 element in $\delta(q, \varepsilon, t)$, and $\delta(q, a, t)$ empty for all $a \in \Sigma$

Deterministic PDA

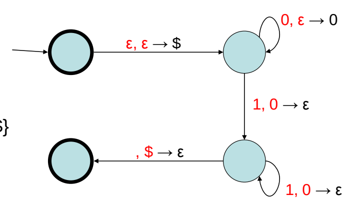
- A technical detail:

  we will give our deterministic machine the ability to detect end of input string
  - add special symbol   to alphabet
  - require input tape to contain x
- language recognized by a deterministic PDA is called a deterministic CFL (DCFL)

Example deterministic PDA

$\Sigma = \{0, 1\}$

$\Gamma = \{0, 1, \$\}$



$L = \{0^n1^n : n \geq 0\}$

(unpictured transitions go to a "reject" state and stay there)

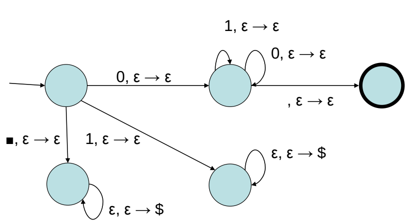
Deterministic PDA

Theorem: DCFLs are closed under complement

(complement of L in Σ^* is $(\Sigma^* - L)$)

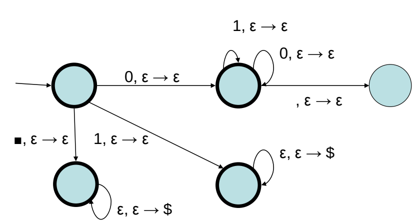
Proof attempt:
- swap accept/non-accept states
- problem: might enter infinite loop before reading entire string
- machine for complement must accept in these cases, and read to end of string

Example of problem



Language of this DPDA is $0\Sigma^*$

Example of problem



Language of this DPDA is $\{\varepsilon\}$

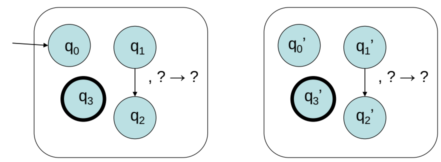
Deterministic PDA

Proof:

- convert machine into “normal form”
  - always reads to end of input
  - always enters either an accept state or single distinguished “reject” state
- step 1: keep track of when we have read to end of input
- step 2: eliminate infinite loops

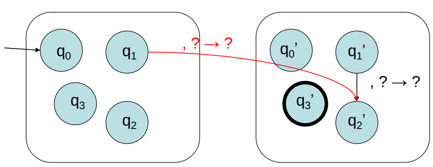
Deterministic PDA

step 1: keep track of when we have read to end of input



Deterministic PDA

step 1: keep track of when we have read to end of input

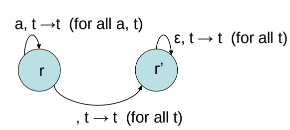


for accept state q’: replace outgoing “ε, ? → ?” transition with self-loop with same label

Deterministic PDA

step 2: eliminate infinite loops

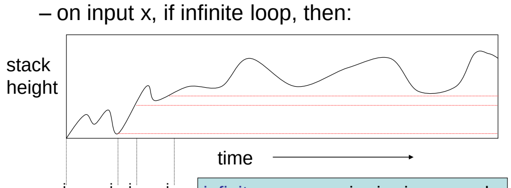
- add new “reject” states



Deterministic PDA

step 2: eliminate infinite loops

- on input x, if infinite loop, then:



infinite sequence $i_0 < i_1 < i_2 < \dots$ such that for all k, stack height never decreases below $ht(i_k)$ after time i_k