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:

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

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

CFL Pumping Lemma Example

- possibilities:
  - $w = \text{aaaa...aabbb...bcccc...c}$

(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:
  - $w = \text{aaaa...abbbb...bcccccc...c}$

(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.)
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 \( w = 0^p10^p1 \)
- can this be pumped?

CFL Pumping Lemma Example

\[ L = \{ xx : x \in \{0,1\}^* \} \]  
- try \( w = 0^p10^p1 \)
- \( w = uvxyz \), with \( |vy| > 0 \) and \( |vxy| \leq p \).
- case: \( vxy \) in first half.
  - then \( uv^2xy^2z = ??...??...?? \)
- case: \( vxy \) in second half.
  - then \( uv^2xy^2z = ??...0??...?1 \)
- case: \( vxy \) straddles midpoint.
  - then \( uv^2xy^2z = uoz = 0^p10^p1p \) with \( i \neq 2p \) or \( j \neq 2p \)

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 \( i \geq 0 \), \( uv^i\cdot x^i \cdot y^i \cdot z^i \in L \), and
2. \( |vy| > 0 \), and
3. \( |vxy| \leq p \).

CFL Pumping Lemma Example

**Proof:** consider a parse tree for a very long string \( w \in L \):

- long path
- some non-terminal must repeat on long path

Schematic proof:

- \( u \) \( v \) \( x \) \( y \) \( z \)
- \( u \) \( v \) \( x \) \( y \) \( z \)
- \( u \) \( v \) \( x \) \( y \) \( z \)
- \( u \) \( v \) \( x \) \( y \) \( z \)
CFL Pumping Lemma

– how large should pumping length $p$ be?
– need to ensure other conditions:
  $|vy| > 0$  
  $|vxy| \leq p$

– $b = \text{max # symbols on rhs of any production (assume } b \geq 2)$
– if parse tree has height $\leq h$, then string generated has length $\leq b^h$ (so length $> b^h$ implies height $> h$)

Chomsky Normal Form

• Useful to deal only with CFGs in a simple normal form
• Most common: Chomsky Normal Form (CNF)
• Definition: every production has form
  $A \rightarrow BC$  
  or  
  $S \rightarrow \varepsilon$  
  or  
  $A \rightarrow a$

where $A, B, C$ are any non-terminals (and $B, C$ are not $S$) and $a$ is any terminal.

Deciding CFLs

• Useful to have an efficient algorithm to decide whether string $x$ is in given CFL
  – e.g. programming language often described by CFG. Determine if string is valid program.
• If CFL recognized by deterministic PDA, just simulate the PDA.
  – but not all CFLs are (homework)...
• Can simulate NPDA, but this takes exponential time in the worst case.
Deciding CFLs

- Convert CFG into Chomsky Normal Form
- Parse tree for string $x$ generated by nonterminal $A$:

$$A \rightarrow BC \text{ is a production and } \ A \rightarrow i \ y \text{ and } C \Rightarrow j \ z \text{ for } i, j < k$$

- If $A \rightarrow^k x$ ($k > 1$) then there must be a way to split $x$:

$$x = yz$$

- worst case running time $\exp(n)$
- Idea: avoid recursive calls
  - build table of YES/NO answers to calls to $\text{IsGenerated}$, in order of length of substring
  - example of general algorithmic strategy called dynamic programming
  - notation: $x[i, j] = \text{substring of } x \text{ from } i \text{ to } j$
  - table: $T(i, j)$ contains

$$\{A: \text{A nonterminal such that } A \rightarrow^* x[i, j] \}$$

An algorithm:

$$\text{IsGenerated}(x, A)$$

for $i = 1$ to $n$

$T[i, i] = \{A: "A \rightarrow x_i" \text{ is a production in } G\}$

for $k = 1$ to $n - 1$

for $i = 1$ to $n - k$

for $k$ splittings $x[i, i+k] = x[i,i+j]x[i+j+1,i+k]$

$T[i, i+k] = \{A: "A \rightarrow BC" \text{ is a production in } G \text{ and } B \in T[i,i+j] \text{ and } C \in T[i+j+1,i+k]\}$

output “YES” if $S \in T[1, n]$, else output “NO”

- worst case running time?

$\mathcal{O}(nm)$ steps

$\mathcal{O}(n^3m^3)$ steps