Context-Free Grammars

November 2, 2010

Context-Free Grammars

- An unrestricted grammar consists of
  1. A set V of **variables** (a.k.a. **nonterminals**)
  2. A disjoint set Σ (of **terminals**)
  3. A set of rules of the form LEFT $\rightarrow$ RIGHT where
     LEFT $\in (V \cup \Sigma)^+$ and RIGHT $\in (V \cup \Sigma)^*$
  4. One designated $S \in V$, called the **start variable** (a.k.a. **start symbol**
- If every LEFT is a single variable, the grammar is said to be **context-free**.

Rewriting Strings

- If we have a rule LEFT $\rightarrow$ RIGHT, we can replace LEFT by RIGHT <u>inside</u> any string:

$$\alpha\text{LEFT}\beta \Rightarrow \alpha\text{RIGHT}\beta$$

- The language of a grammar G is the set

$$L(G) = \{\ w \in \Sigma^* \mid S \Rightarrow^* w\ \}$$

- The same language L might be the language of many different grammars.
  - L is said to be a **context-free language** if it can be generated by **at least one** context-free grammar.

CFG Example

$S \rightarrow \varepsilon$

$S \rightarrow 0B$

$S \rightarrow 1A$

$A \rightarrow 0S$

$A \rightarrow 1AA$

$B \rightarrow 1S$

$B \rightarrow 0BB$

- What is the start symbol?
- What are the terminals and nonterminals?
- What strings does this grammar generate?

CFG Construction

- $\{ w \in \Sigma^* \mid \text{length of } w \text{ is odd with middle } 0 \}$
- $\{ w \in \Sigma^* \mid \text{length of } w \text{ is odd} \}$

Notational Variants

S → **if** *E* **then** *S* **else** *S*
| **begin** *S* *L*
| **print** *E*

```
<digit> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<expr>  ::= <digit>
          | <expr> - <expr>
          | ( <expr> - <expr> )
```

pointer:
* *type-qualifier-list*$_{opt}$
* *type-qualifier-list*$_{opt}$ *pointer*

Production Sequences

$$E \to E + E \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$E \Rightarrow E + E \Rightarrow 3 + E \Rightarrow 3 + E + E \Rightarrow 3 + 2 + E \Rightarrow 3 + 2 + 1$$

$$E \Rightarrow E + E \Rightarrow 3 + E \Rightarrow 3 + E + E \Rightarrow 3 + E + 1 \Rightarrow 3 + 2 + 1$$

$$E \Rightarrow E + E \Rightarrow E + 1 \Rightarrow E + E + 1 \Rightarrow 3 + E + 1 \Rightarrow 3 + 2 + 1$$

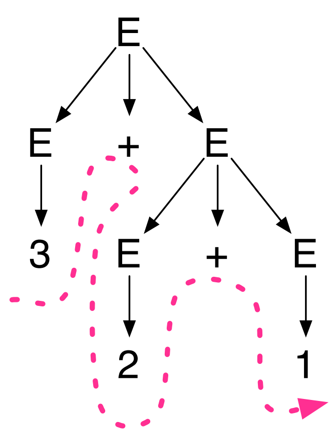
$$E \Rightarrow E + E \Rightarrow E + 1 \Rightarrow E + E + 1 \Rightarrow E + 2 + 1 \Rightarrow 3 + 2 + 1$$

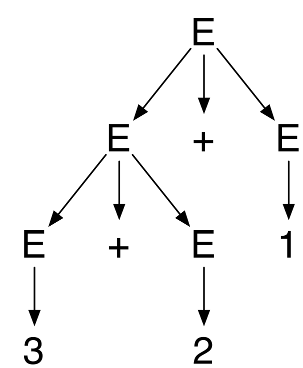
Parse Trees

$$E \to E + E \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$E \Rightarrow E + E \Rightarrow 3 + E \Rightarrow 3 + E + E \Rightarrow 3 + 2 + E \Rightarrow 3 + 2 + 1$$

$$E \Rightarrow E + E \Rightarrow 3 + E \Rightarrow 3 + E + E \Rightarrow 3 + E + 1 \Rightarrow 3 + 2 + 1$$





$$E \Rightarrow E + E \Rightarrow E + 1 \Rightarrow E + E + 1 \Rightarrow 3 + E + 1 \Rightarrow 3 + 2 + 1$$

$$E \Rightarrow E + E \Rightarrow E + 1 \Rightarrow E + E + 1 \Rightarrow E + 2 + 1 \Rightarrow 3 + 2 + 1$$

Ambiguity

- A grammar is **ambiguous** if there are strings having more than one parse tree.

Exercise

- Are these grammars ambiguous?

E → d
  | E - E

E → d
  | (E - E)

E → d
  | E - d

E → d
  | d - E

E → d
  | E E -

Encoding Precedence

```
E → d
  | E + E
  | E - E
  | E * E
  | E / E
  | ( E )
```

```
E → T
  | E + T
  | E - T

T → F
  | T * F
  | T / F

F → d
  | ( E )
```

Digression

- There exist **inherently ambigous** languages that have no unambiguous grammar, e.g.,

$$\{a^n b^n c^m d^m \mid n,m>0\} \cup \{a^n b^m c^m d^n \mid n,m>0\}$$

Another Pumping Lemma

If L is context-free, then there exists a number p such that:

for every $s \in L$ with $|s| \geq p$:

There's at least one way to decompose s into uvxyz where

$|vy| > 0$ and

$|vxy| \leq p$ and

$uv^ixy^iz \in L$ for every $i \geq 0$.

$\{a^n b^n c^n \mid n \geq 0\}$

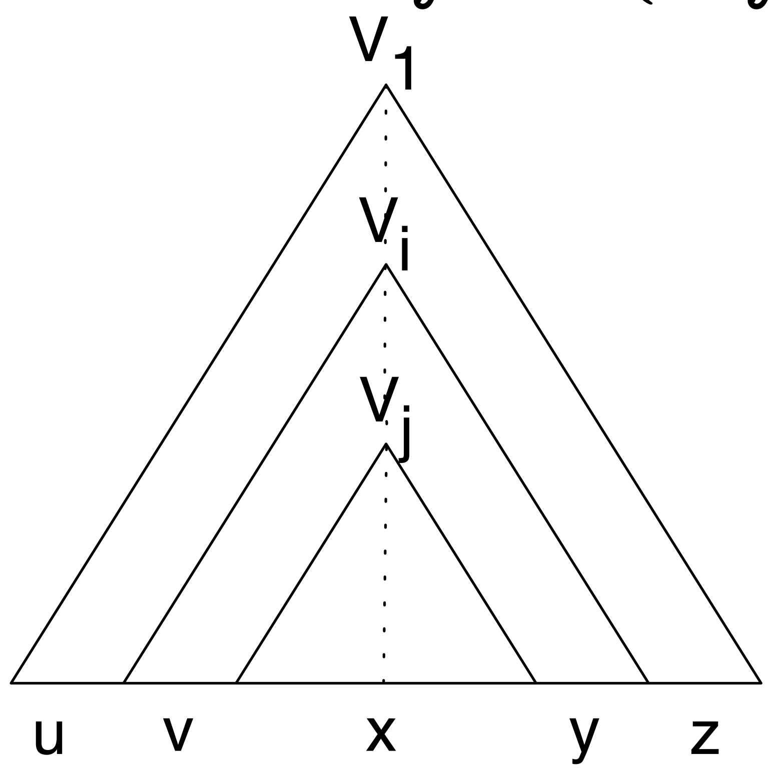
- Assume this language is context-free.
- Let p be its pumping length.
- Consider $s := a^p b^p c^p$. Can it be pumped?
  - Must consider all ways of dividing s into uvxyz with $|vy| > 0$ and $|vxy| \leq p$, and show that pumping will take us out of the language.
- Contradiction, so this language is not context-free.

$\{ww \mid w \in \Sigma^* \}$

- Assume this language is context-free.
- Let p be its pumping length.
- Consider $s := 0^p 1 0^p 1$. Can it be pumped?
  - Must consider all ways of dividing s into uvxyz with $|vy| > 0$ and $|vxy| \leq p$, and show that pumping will take us out of the language.
  - Oops…this string **can** be pumped. That's not helpful.
- Consider $s := 0^p 1^p 0^p 1^p$. Can it be pumped?
- Contradiction, so this language is not context-free.

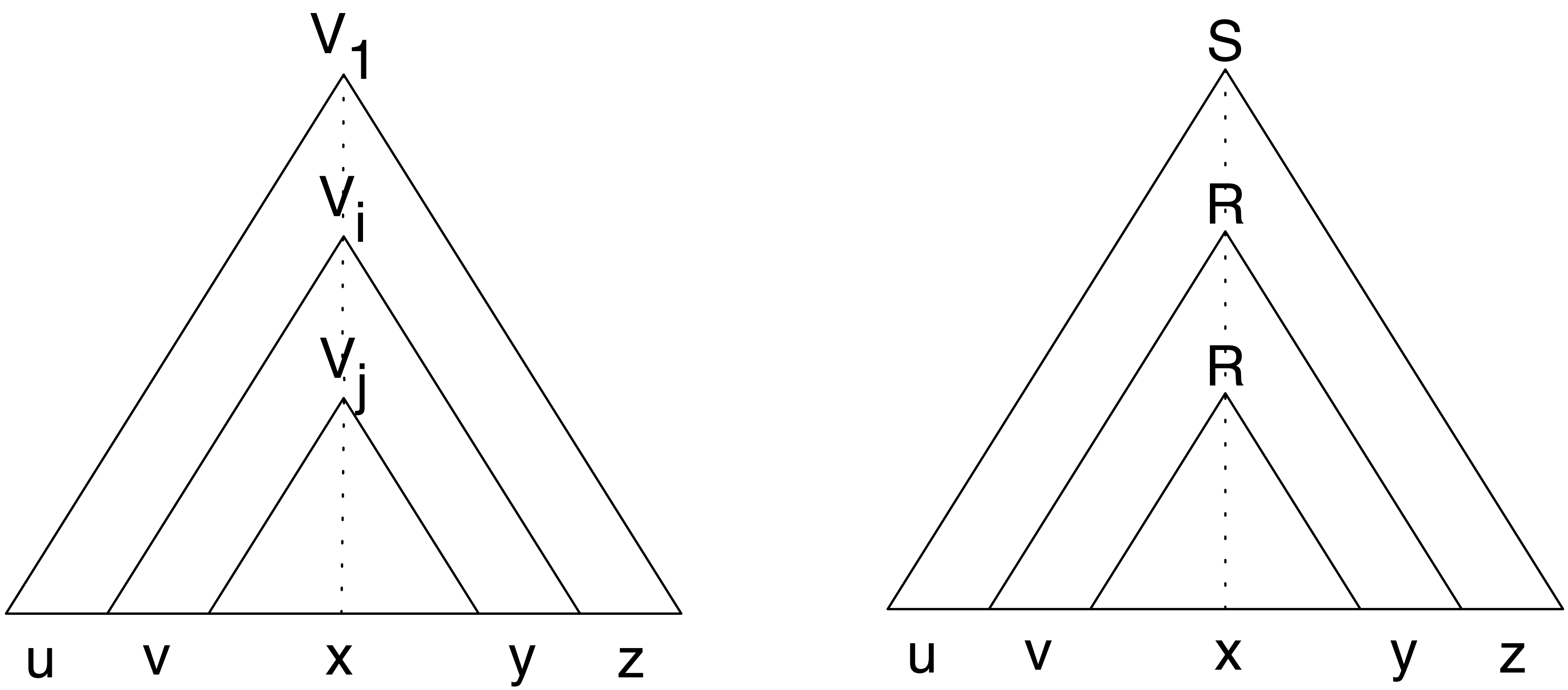
Intuition

- Consider the parse tree for a long string.
- If we make the string "long enough", the parse tree must grow arbitrarily tall. (Why?)



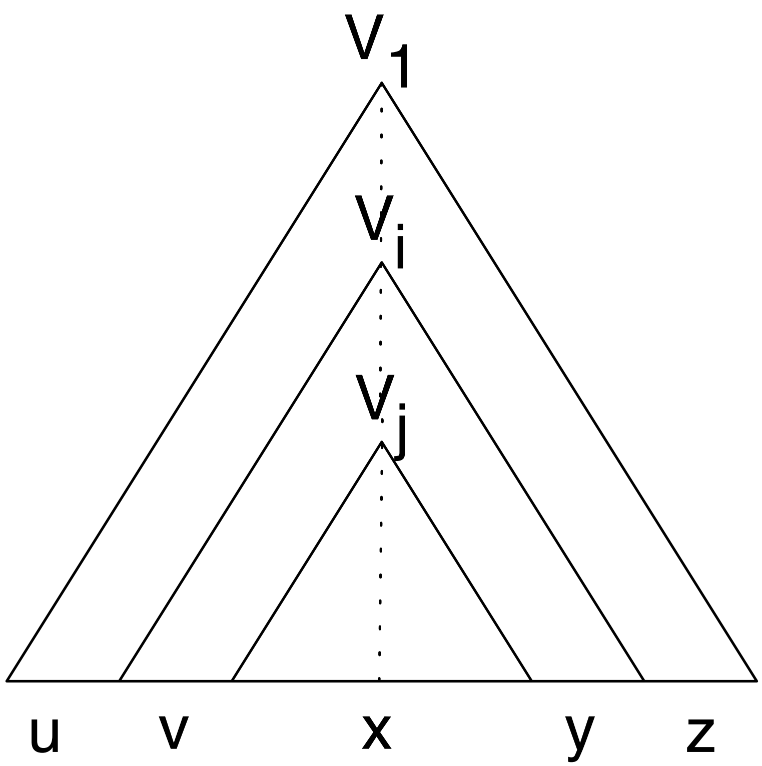
Applying Pigeonhole

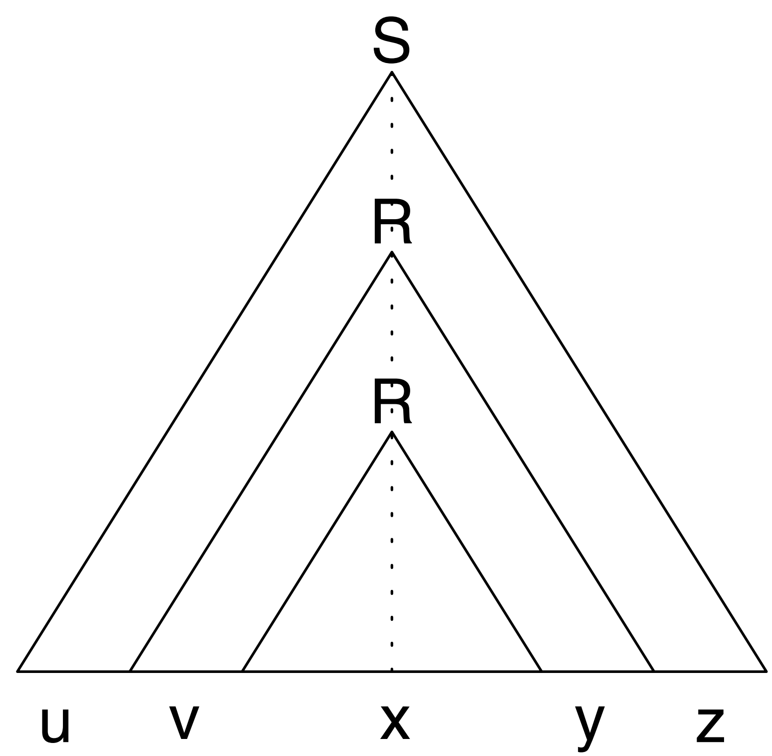
- Assume we have a string in L whose <u>shortest</u> parse tree has height ≥ |V|+1. [height = edges]



Applying Pigeonhole

- Assume we have a string in L whose <u>shortest</u> parse tree has height ≥ |V|+1. [height = edges]





Pumping

Final questions:

(1) How do we know v and y aren't both ε?

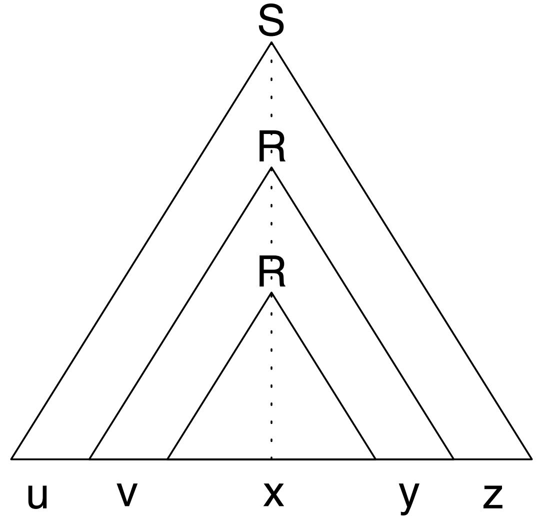
E.g.,

$R \Rightarrow P$

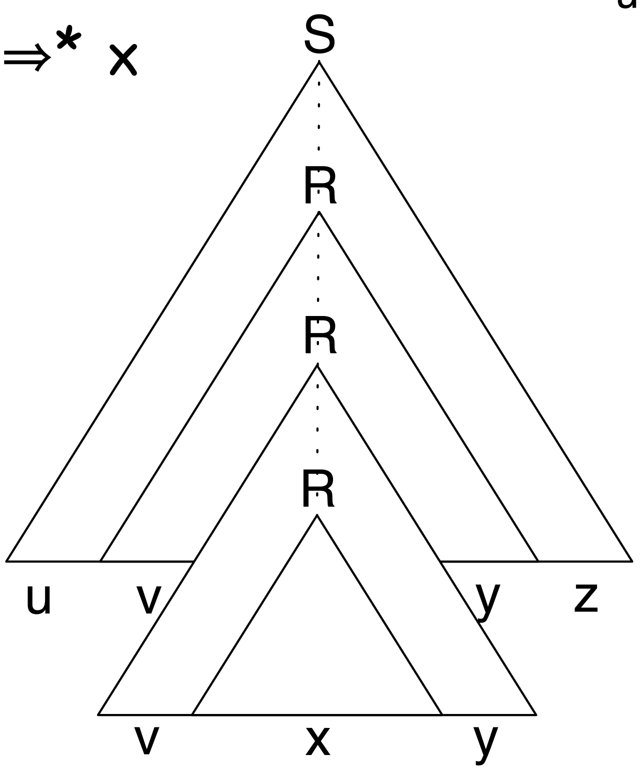
$P \Rightarrow Q$

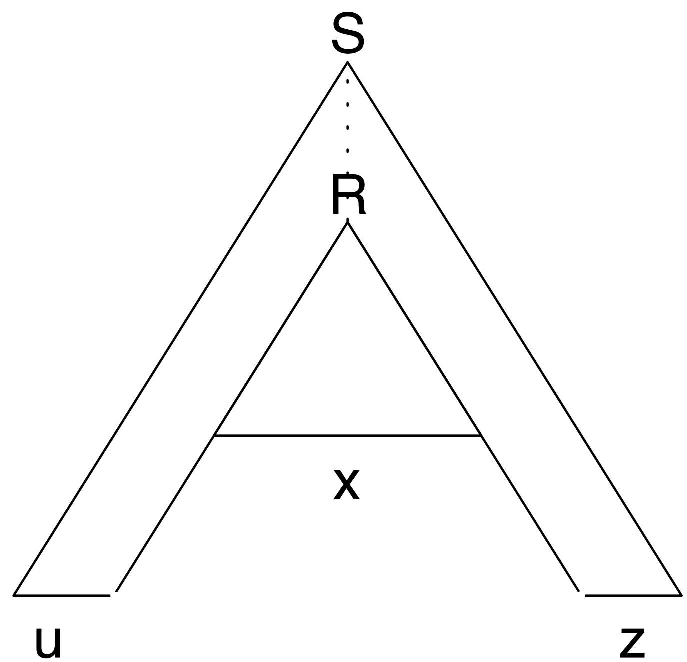
$Q \Rightarrow R$

$R \Rightarrow^* x$



(2) Is there a point beyond which all strings have "tall" parse trees?





Regular Grammars

- A grammar is said to be **regular** if its rules are of the following forms:

$$X \rightarrow a$$

$$X \rightarrow \varepsilon$$

$$X \rightarrow aY$$

Example

$S \rightarrow 1B$

$B \rightarrow 1B$

$B \rightarrow 0C$

$C \rightarrow 0S$

$S \rightarrow 0S$

$C \rightarrow 1B$

$C \rightarrow \varepsilon$