Week4 friday

Big picture: PDAs were motivated by wanting to add some memory of unbounded size to NFA. How do we accomplish a similar enhancement of regular expressions to get a syntactic model that is more expressive?

DFA, NFA, PDA: Machines process one input string at a time; the computation of a machine on its input string reads the input from left to right.

Regular expressions: Syntactic descriptions of all strings that match a particular pattern; the language described by a regular expression is built up recursively according to the expression's syntax

Context-free grammars: Rules to produce one string at a time, adding characters from the middle, beginning, or end of the final string as the derivation proceeds.

Definitions below are on pages 101-102.

Term	Typical symbol	Meaning
	or Notation	
Context-free grammar (CFG)	G	$G = (V, \Sigma, R, S)$
The set of variables	V	Finite set of symbols that represent phases in pro-
		duction pattern
The set of terminals	Σ	Alphabet of symbols of strings generated by CFG $V \cap \Sigma = \emptyset$
The set of rules	R	Each rule is $A \to u$ with $A \in V$ and $u \in (V \cup \Sigma)^*$
The start variable	S	Usually on left-hand-side of first/ topmost rule
Derivation	$S \Rightarrow \cdots \Rightarrow w$	Sequence of substitutions in a CFG (also written $S \Rightarrow^* w$). At each step, we can apply one rule to one occurrence of a variable in the current string
Language generated by the context-free grammar G	L(G)	by substituting that occurrence of the variable with the right-hand-side of the rule. The derivation must end when the current string has only terminals (no variables) because then there are no instances of variables to apply a rule to. The set of strings for which there is a derivation in G . Symbolically: $\{w \in \Sigma^* \mid S \Rightarrow^* w\}$ i.e.
		$\{w \in \Sigma^* \mid \text{there is derivation in } G \text{ that ends in } w\}$
Context-free language		A language that is the language generated by some context-free grammar

Examples of context-free grammars, derivations in those grammars, and the languages generated by those grammars

$$G_1 = (\{S\}, \{0\}, R, S)$$
 with rules

$$S \to 0 S$$

$$S \to 0$$

In
$$L(G_1)$$
 ...

Not in $L(G_1)$...



 $S \to 0S \mid 1S \mid \varepsilon$

In $L(G_2)$...

Not in $L(G_2)$...

 $(\{S, T\}, \{0, 1\}, R, S)$ with rules

$$\begin{split} S &\to T1T1T1T \\ T &\to 0T \mid 1T \mid \varepsilon \end{split}$$

In $L(G_3)$...

Not in $L(G_3)$...

 $G_4 = (\{A, B\}, \{0, 1\}, R, A)$ with rules

 $A \rightarrow 0A0 \mid 0A1 \mid 1A0 \mid 1A1 \mid 1$

In $L(G_4)$...

Not in $L(G_4)$...

Design a CFG to generate the language $\{a^nb^n\mid n\geq 0\}$
Sample derivation: