Week4 wednesday

Definition A **pushdown automaton** (PDA) is specified by a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where Q is the finite set of states, Σ is the input alphabet, Γ is the stack alphabet,

$$\delta: Q \times \Sigma_{\varepsilon} \times \Gamma_{\varepsilon} \to \mathcal{P}(Q \times \Gamma_{\varepsilon})$$

is the transition function, $q_0 \in Q$ is the start state, $F \subseteq Q$ is the set of accept states.

Draw the state diagram and give the formal definition of a PDA with $\Sigma = \Gamma$.

Draw the state diagram and give the formal definition of a PDA with $\Sigma \cap \Gamma = \emptyset$.

Mathematical description of language

State diagram of PDA recognizing language

$$\Gamma = \{\$, \#\}$$



$$\Gamma = \{ \stackrel{\Leftrightarrow}{,} 1 \}$$



$$\{0^i 1^j 0^k \mid i, j, k \ge 0\}$$

Note: alternate notation is to replace; with \rightarrow

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 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.
Language generated by the context-free grammar G	L(G)	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 $\{a^nb^n\mid n\geq 0\}$					
mple derivation:					

Week3 wednesday

Definition and Theorem: For an alphabet Σ , a language L over Σ is called **regular** exactly when L is recognized by some DFA, which happens exactly when L is recognized by some NFA, and happens exactly when L is described by some regular expression

We saw that: The class of regular languages is closed under complementation, union, intersection, set-wise concatenation, and Kleene star.

Prove or Disprove: There is some alphabet Σ for which there is some language recognized by an NFA but not by any DFA.

Prove or Disprove: There is some alphabet Σ for which there is some finite language not described by any regular expression over Σ .

Prove or Disprove: If a language is recognized by an NFA then the complement of this language is not recognized by any DFA.

Fix alphabet Σ . Is every language L over Σ regular?

Set	Cardinality
$\{0,1\}$	
$\{0,1\}^*$	
$\mathcal{P}(\{0,1\})$	
The set of all languages over $\{0,1\}$	
The set of all regular expressions over $\{0,1\}$	
The set of all regular languages over $\{0,1\}$	

Strategy: Find an **invariant** property that is true of all regular languages. When analyzing a given language, if the invariant is not true about it, then the language is not regular.

Pumping Lemma (Sipser Theorem 1.70): If A is a regular language, then there is a number p (a pumping length) where, if s is any string in A of length at least p, then s may be divided into three pieces, s = xyz such that

- |y| > 0
- for each $i \ge 0$, $xy^iz \in A$
- $|xy| \leq p$.

Proof illustration

True or False: A pumping length for $A = \{0, 1\}^*$ is p = 5.

Week3 friday

Recap so far: In DFA, the only memory available is in the states. Automata can only "remember" finitely far in the past and finitely much information, because they can have only finitely many states. If a computation path of a DFA visits the same state more than once, the machine can't tell the difference between the first time and future times it visits this state. Thus, if a DFA accepts one long string, then it must accept (infinitely) many similar strings.

Definition A positive integer p is a **pumping length** of a language L over Σ means that, for each string $s \in \Sigma^*$, if $|s| \ge p$ and $s \in L$, then there are strings x, y, z such that

$$s = xyz$$

and

$$|y| > 0$$
, for each $i \ge 0$, $xy^i z \in L$, and $|xy| \le p$.

Negation: A positive integer p is **not a pumping length** of a language L over Σ iff

$$\exists s \ (\ |s| \ge p \land s \in L \land \forall x \forall y \forall z \ (\ (s = xyz \land |y| > 0 \land |xy| \le p \) \rightarrow \exists i (i \ge 0 \land xy^iz \notin L)) \)$$

Informally:

Restating **Pumping Lemma**: If L is a regular language, then it has a pumping length.

Contrapositive: If L has no pumping length, then it is nonregular.

The Pumping Lemma cannot be used to prove that a language is regular.

The Pumping Lemma can be used to prove that a language is not regular.

Extra practice: Exercise 1.49 in the book.

Proof strategy: To prove that a language L is **not** regular,

- Consider an arbitrary positive integer p
- Prove that p is not a pumping length for L
- Conclude that L does not have any pumping length, and therefore it is not regular.

Example: $\Sigma = \{0, 1\}, L = \{0^n 1^n \mid n \ge 0\}.$

Fix p an arbitrary positive integer. List strings that are in L and have length greater than or equal to p:

 ${\rm Pick}\ s =$

Suppose s = xyz with $|xy| \le p$ and |y| > 0.

 $, xy^{i}z =$

Then when i =

Example: $\Sigma = \{0, 1\}$, $L = \{ww^{\mathcal{R}} \mid w \in \{0, 1\}^*\}$. Remember that the reverse of a string w is denoted $w^{\mathcal{R}}$ and means to write w in the opposite order, if $w = w_1 \cdots w_n$ then $w^{\mathcal{R}} = w_n \cdots w_1$. Note: $\varepsilon^{\mathcal{R}} = \varepsilon$. Fix p an arbitrary positive integer. List strings that are in L and have length greater than or equal to p: Pick s =Suppose s = xyz with $|xy| \le p$ and |y| > 0. $, xy^iz =$ Then when i =Example: $\Sigma = \{0, 1\}, L = \{0^j 1^k \mid j \ge k \ge 0\}.$ Fix p an arbitrary positive integer. List strings that are in L and have length greater than or equal to p: Pick s =Suppose s = xyz with $|xy| \le p$ and |y| > 0. $xy^iz =$ Then when i =Example: $\Sigma = \{0, 1\}, L = \{0^n 1^m 0^n \mid m, n \ge 0\}.$ Fix p an arbitrary positive integer. List strings that are in L and have length greater than or equal to p: Pick s =Suppose s = xyz with $|xy| \le p$ and |y| > 0. $xy^iz =$ Then when i =

$Extra\ practice:$

Language	$s \in L$	$s \notin L$	Is the language regular or nonregular?
$\{a^nb^n\mid 0\leq n\leq 5\}$			
$\{b^na^n\mid n\geq 2\}$			
$\{a^mb^n\mid 0\leq m\leq n\}$			
$\{a^mb^n\mid m\geq n+3, n\geq 0\}$			
$\{b^ma^n\mid m\geq 1, n\geq 3\}$			
$\{w \in \{a, b\}^* \mid w = w^{\mathcal{R}}\}$			
$\{ww^{\mathcal{R}} \mid w \in \{a, b\}^*\}$			