EXAM 3 SOLUTIONS - CSCI 341

Friday, November 9

George Voutsadakis

Read each problem **very carefully** before starting to solve it. Each problem is worth 10 points. It is necessary to show **all** your work. Correct answers without explanations are worth 0 points. GOOD LUCK!!

1. (a) Give the formal inductive definition of the set of regular languages over alphabet A.

   Basis: $\emptyset$, $\{\Lambda\}$ and $\{a\}$ are regular languages, for all $a \in A$;
   Induction: If L and M are regular languages, then the following languages are also regular: $L \cup M$, ML and L^*.

   (b) Give a regular expression for the language of all strings over alphabet $A = \{a, b, c\}$ that start with abc or end with cba.

   $abc(a + b + c)^* + (a + b + c)^* cba$

   (c) Give a regular expression for the language of all strings over the alphabet $A = \{0, 1\}$ that do not end with 01.

   $\Lambda + 0 + 1 + (0 + 1)^*(00 + 10 + 11)$

   (d) Give a formal recursive definition of the operator L that takes as input a regular expression over alphabet A and outputs the regular language that is represented by the given regular expression.

   Basis: $L(\emptyset) = \emptyset$; $L(\Lambda) = \{\Lambda\}$; $L(a) = \{a\}$, for all $a \in A$;
   Recursion: $L(R + S) = L(R) \cup L(S)$; $L(R \cdot S) = L(R)L(S)$; $L(R^*) = L(R)^*$.

   (e) The statement "The language L of all strings over alphabet $A = \{a, b\}$ that have odd length is regular" is <u>true</u>
   Proof:
   The language L of all strings over alphabet $A = \{a, b\}$ that have odd length is regular because it is the language of the following regular expression:

$$(a + b)(aa + ab + ba + bb)^*$$

2. (a) The transition function of a DFA is a function $\delta : \underline{S \times A} \to \underline{S}$, where

S is the set of states and A is the input alphabet of the DFA.

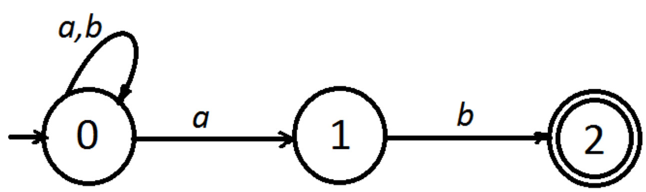
(b) The transition function of an NFA is a function $\delta : \underline{S \times (A \cup \{\Lambda\})} \to \underline{\mathcal{P}(S)}$, where

S is the set of states and A is the input alphabet of the DFA.

(c) An NFA N **accepts a string** w over its input alphabet A if, by definition

there exists an execution that consumes all the letters of w and ends in a final state.

(d) Give an NFA that recognizes the regular language $\{a, b\}^*\{ab\}$.

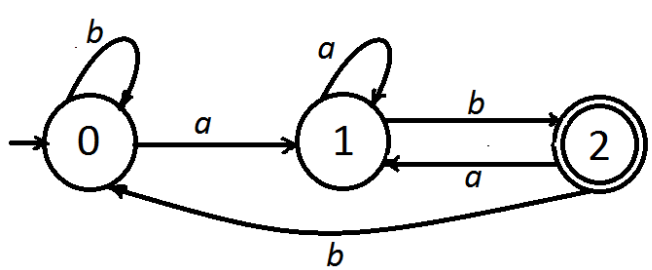


(e) Apply the NFA to DFA algorithm to obtain the DFA that corresponds to the NFA you constructed in Part (d).
We get the transition table on the left, where $\{0\}$ is the initial state and $\{0, 2\}$ is the only final state. By renaming the states we get the transition table on the right, with 0 the initial state and 2 the only final state.

	a	b
$\{0\}$	$\{0,1\}$	$\{0\}$
$\{0,1\}$	$\{0,1\}$	$\{0,2\}$
$\{0,2\}$	$\{0,1\}$	$\{0\}$

	a	b
0	1	0
1	1	2
2	1	0

The DFA is pictured below.



3. (a) Give the form of productions that are allowed in a regular grammar, explaining your symbols.

A grammar is called a **regular grammar** if each production takes one of the following forms, where the uppercase letters are nonterminals and w is a nonempty string of terminals:

$$S \to \Lambda, \quad S \to w, \quad S \to T, \quad S \to wT.$$

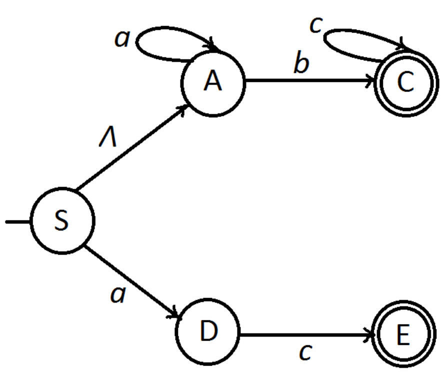
(b) Under each regular expression provide the regular grammar whose language is the same as that represented by the corresponding regular expression:

$a^* + b^*$	ba^*	$(aba)^*$
$S \to A\|B$ $A \to aA\|\Lambda$ $A \to bB\|\Lambda$	$S \to bA$ $A \to aA\|\Lambda$	$S \to \Lambda\|abaS$

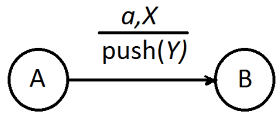
(c) Find a regular grammar for the language represented by the regular expression $a^*bc^* + ac$.

$$S \to ac|A$$
$$A \to aA|bC$$
$$C \to cC|\Lambda$$

(d) Apply the algorithm converting a regular grammar to an NFA to obtain an NFA corresponding to the regular grammar that you gave in Part (c).



4. (a) The PDA instruction $\langle A, a, X, \mathsf{push}(Y), B\rangle$ can be shown pictorially as follows:



The interpretation of it is that:

If at state A, reading input symbol a and X at the top of the stack, then transition to state B and push symbol Y at the top of the stack.

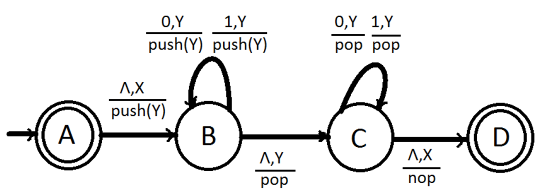
(b) Give the formal definition of an instantaneous description (ID) in a computation of a PDA.

An instantaneous description in the computation of a PDA is a triple of the form

$$(\text{current state}, \text{unconsumed input}, \text{stack contents})$$

that represents the current state and current status of input and stack during a computation of the PDA.

(c) Consider the following PDA with initial stack symbol X. Give a formal description of an accepting computation on input string 110011 in terms of IDs.



$$\begin{aligned}(A, 110011, X) &\to (B, 110011, YX) \to (B, 10011, YYX) \to (B, 0011, YYYX)\\ &\to (B, 011, YYYYX) \to (C, 011, YYYX) \to (C, 11, YYX)\\ &\to (C, 1, YX) \to (C, \Lambda, X) \to (D, \Lambda, X).\end{aligned}$$

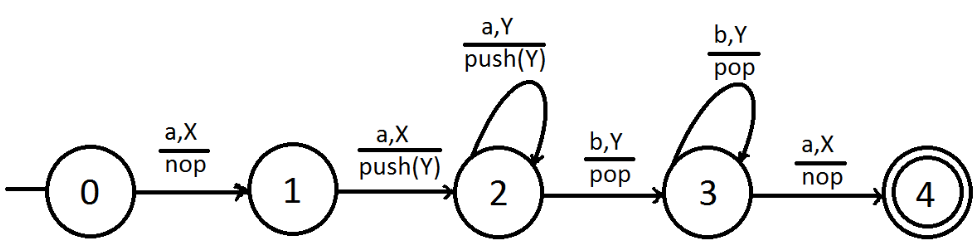
(d) "The language accepted by the PDA of Part (c) is the language of palindromes over $A = \{0, 1\}$" is <u>false</u>

proof:

The string 01 is also accepted and is not a palindrome:

$$(A, 01, X) \to (B, 01, YX) \to (B, 1, YYX) \to (C, 1, YX) \to (C, \Lambda, X) \to (D, \Lambda, X).$$

(e) Give a PDA that recognizes the language $\{a^{n+1}b^n a : n \geq 1\}$ (make sure to state which mode of acceptance your PDA is using).

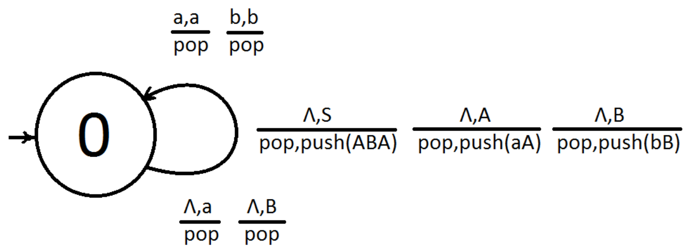


5. (a) Give an example of a language that is context-free but not regular (write formally).

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

(b) Apply the relevant algorithm to construct a PDA that accepts the context-free language described by

$S \to ABA$
$A \to aA|\Lambda$
$B \to bB|\Lambda.$



(c) Apply the relevant algorithm to eliminate Λ-productions from the following grammar:

$S \to aSA|bSB|A$
$A \to aBb|bBa$
$B \to aB|bB|\Lambda$

Step 1: Detect $B \to \Lambda$;

Step 2: B is involved in the right hand side of the rules on the left and cause addition of rules on right:

$S \to bSB$	$S \to bS$
$A \to aBb$	$A \to ab$
$A \to bBa$	$A \to ba$
$B \to aB$	$B \to a$
$B \to bB$	$B \to b$

Step 3: The grammar becomes:

$S \to aSA|bSB|bS|A$
$A \to aBb|bBa|ab|ba$
$B \to aB|bB|a|b$

(d) Apply the relevant algorithm to produce the Chomsky Normal Form of the following context-free grammar.

$$
\begin{aligned}
S &\rightarrow C \\
C &\rightarrow aCa|bCb|A \\
A &\rightarrow aBb|bBa \\
B &\rightarrow aB|bB|b
\end{aligned}
$$

Steps 1-2: Not needed.

Step 3: $S \rightarrow C$ gives $S \rightarrow aCa$ and $S \rightarrow bCb$, $S \rightarrow A$ gives $S \rightarrow aBb$ and $S \rightarrow bBa$, and $C \rightarrow A$ gives $C \rightarrow aBb$ and $C \rightarrow bBa$

So we get

$$
\begin{aligned}
S &\rightarrow aCa|bCb|aBb|bBa \\
C &\rightarrow aCa|bCb|aBb|bBa \\
B &\rightarrow aB|bB|b
\end{aligned}
$$

Step 4: New grammar

$$
\begin{aligned}
S &\rightarrow ACA|BCB|ABB|BBA \\
C &\rightarrow ACA|BCB|ABB|BBA \\
B &\rightarrow AB|BB|b \\
A &\rightarrow a
\end{aligned}
$$

Step 5: Final grammar in Chomsky Normal Form:

$$
\begin{aligned}
S &\rightarrow AD|BE|AF|BG \\
D &\rightarrow CA \\
E &\rightarrow CB \\
F &\rightarrow BB \\
G &\rightarrow BA \\
C &\rightarrow AD|BE|AF|BG \\
B &\rightarrow AB|BB|b \\
A &\rightarrow a
\end{aligned}
$$