Theory of Computation

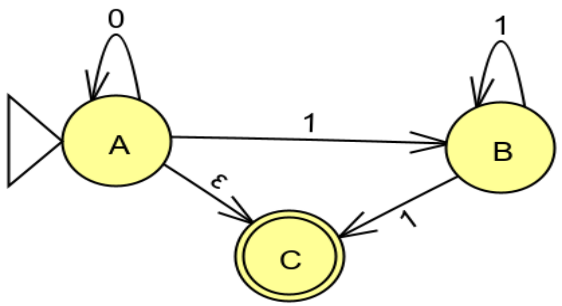
MIEIC, 2nd Year

Exam 1, January 19, 2021
(possible solutions)

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO
DEI DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Group I. [5 Pts] Regular Languages (RLs)

Consider the following ε-NFA:



a) Using the state elimination technique, convert the ε-NFA into a regular expression;

b) Using the subset construction technique, convert the ε-NFA into a DFA and show the transition table and the transition diagram of the DFA;

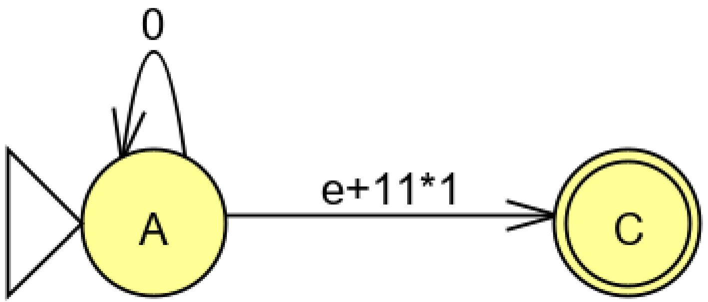
c) Prove by induction that all the strings in the language given by the regular expression 111^* belong to the language of the ε-NFA;

d) For any given ε-NFA, does the inclusion of transitions from the states to a dead state, in a similar way of that of completing a DFA, changes the language of the ε-NFA? Justify your answer.

Group I. [5 Pts] Regular Languages (RLs)

a) Using the state elimination technique, convert the ε-NFA into a regular expression;

Eliminate state B:



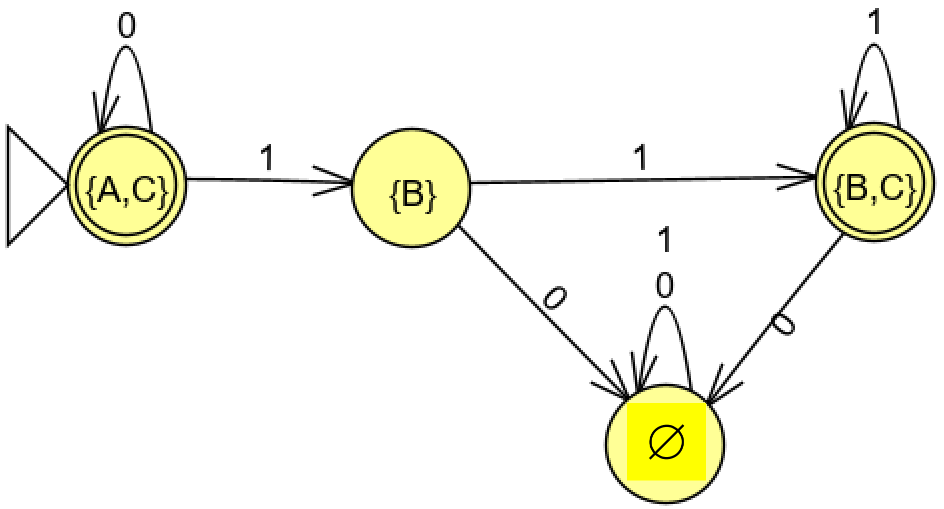
The ‘e’ in the automaton represents ‘ε’

RE = $0^*(\varepsilon+11^*1)$

Group I. [5 Pts] Regular Languages (RLs)

b) Using the subset construction technique, convert the ε-NFA into a DFA and show the transition table and the transition diagram of the DFA;

	0	1
->*{A,C}	{A,C}	{B}
{B}	∅	{B,C}
*{B,C}	∅	{B,C}
∅	∅	∅



Group I. [5 Pts] Regular Languages (RLs)

c) Prove by induction that all the strings in the language given by the regular expression 111* belong to the language of the ε-NFA;

The proof can be done using the ε-NFA, but here I use the DFA:

Be L1 the language L(111) and L2 the language of the ε-NFA, which is the language of the DFA previously obtained from the ε-NFA by the subset construction.*

Hypothesis: every string $\in$ L1 also $\in$ L2

Basis: w=11 ($\in$ L1), δ^(11)={{B,C}} and {B,C} is a final state, thus w $\in$ L2

Induction step:

Assume $w_n \in$ L1 and by hypothesis $w_n \in$ L2. All the strings of L1 are obtained by prefixing w_n with 0 or more occurrences of 1. So, the string $w_{n+1} \in$ L1 is w_n1 (w_n prefixed with 1)

w_n by hypothesis $\in$ L2 and this means that the DFA after processing w_n is at state {B,C}. δ^(w_{n+1})= δ^(w_n1)= δ(δ^(w_n),1)=δ({B,C},1)={B,C}, which is a final state, and thus $w_{n+1} \in$ L2 qed

Group I. [5 Pts] Regular Languages (RLs)

d) For any given ε-NFA, does the inclusion of transitions from the states to a dead state, in a similar way of that of completing a DFA, changes the language of the ε-NFA? Justify your answer.

It does not change the language. The inclusion of transitions in a ε-NFA to a dead state does not change the possible paths taken by the original ε-NFA, and thus maintains the language accepted (the new transitions will make possible other possible non-deterministic paths).

Group II. [5 Pts] Context-Free Languages (CFLs)

Consider the following CFG G1:

$S \rightarrow ScS \mid A \mid B \mid \varepsilon$

$A \rightarrow \varepsilon \mid Aa$

$B \rightarrow \varepsilon \mid Bb$

a) Show a leftmost derivation for the string “aacb” and draw the respective syntax tree;
b) Is this CFG ambiguous? Justify your answer and provide a non-ambiguous grammar in the case this CFG is ambiguous;
c) Draw the PDA, which accepts by empty stack, directly obtained from the CFG G1;
d) Indicate a sequence of instantaneous descriptions that result in the PDA acceptance of the string: “aac”.

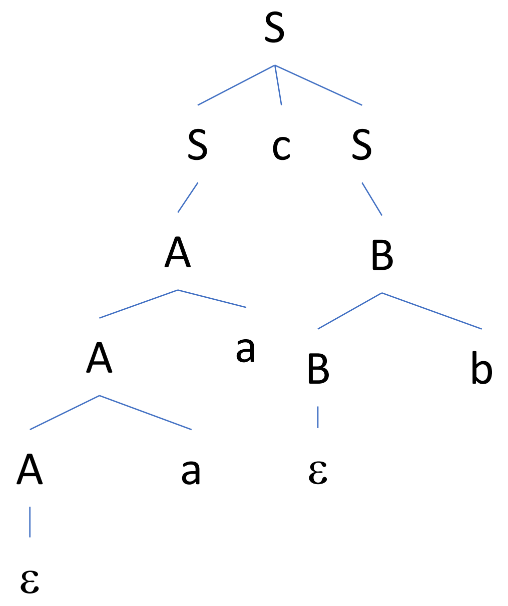
Group II. [5 Pts] Context-Free Languages (CFLs)

a) Show a leftmost derivation for the string "aacb" and draw the respective syntax tree;

Leftmost derivation:

s => ScS => AcS => AacS => AaacS => εaacS => aacB => aacBb => aacεb

Syntax tree:



Group II. [5 Pts] Context-Free Languages (CFLs)

b) Is this CFG ambiguous? Justify your answer and provide a non-ambiguous grammar in the case this CFG is ambiguous;

The CFG is ambiguous. For example, there are three syntax trees for the empty string ε

```
S      S      S
|      |      |
ε      A      B
       |      |
       ε      ε
```

CFG modified to be non-ambiguous:

$S \rightarrow TcS \mid A \mid B \mid \varepsilon$

$T \rightarrow A \mid B \mid \varepsilon$

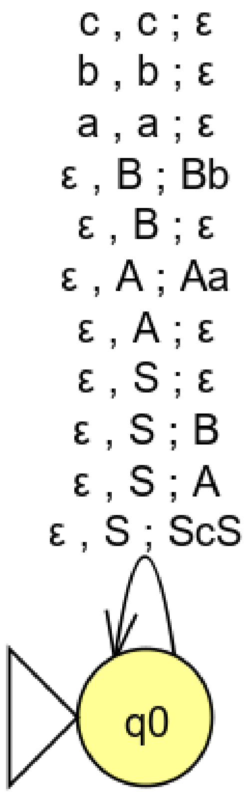
$A \rightarrow a \mid Aa$

$B \rightarrow b \mid Bb$

Group II. [5 Pts] Context-Free Languages (CFLs)

c) Draw the PDA, which accepts by empty stack, directly obtained from the CFG G1;

The PDA starts with S in the stack



Group II. [5 Pts] Context-Free Languages (CFLs)

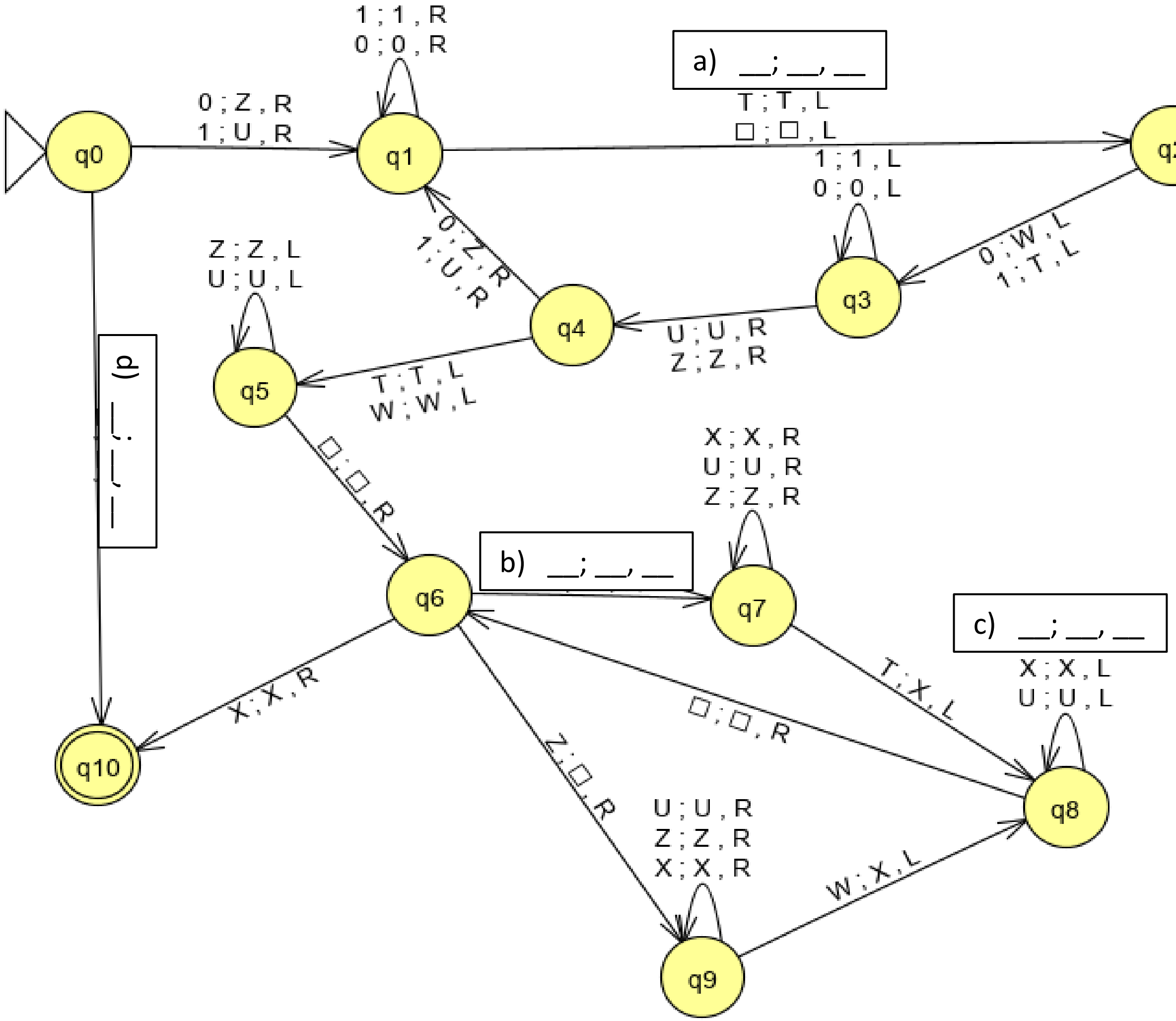
d) Indicate a sequence of instantaneous descriptions that result in the PDA acceptance of the string: “aac”.

$(q_0, aac, S) \vdash (q_0, aac, ScS) \vdash (q_0, aac, AcS) \vdash (q_0, aac, AacS) \vdash (q_0, aac, AaacS) \vdash (q_0, aac, aacS) \vdash (q_0, ac, acS) \vdash (q_0, c, cS) \vdash (q_0, \varepsilon, S) \vdash (q_0, \varepsilon, \varepsilon)$

Group III. [5 Pts] Turing Machine

- We intend a Turing Machine (TM) to implement the language L1=$\{ww \mid w \in \{0,1\}^*\}$. The following TM represents an incomplete solution in which the symbol □ identifies the blank cell (B), R the right step, and L the left step.

Group III. Turing Machine



Group III.

a) Complete the missing transitions of the TM (from a) to d));
b) Describe the intention of the section of the TM consisting of states q0 to q5;
c) Indicate the first 6 steps of the computing trace when the input to the TM is: 00;
d) Identify the modifications to the TM needed to implement the language L2={ww | w $\in$ {0,1}*, and |w| > 0}.

Turing Machine

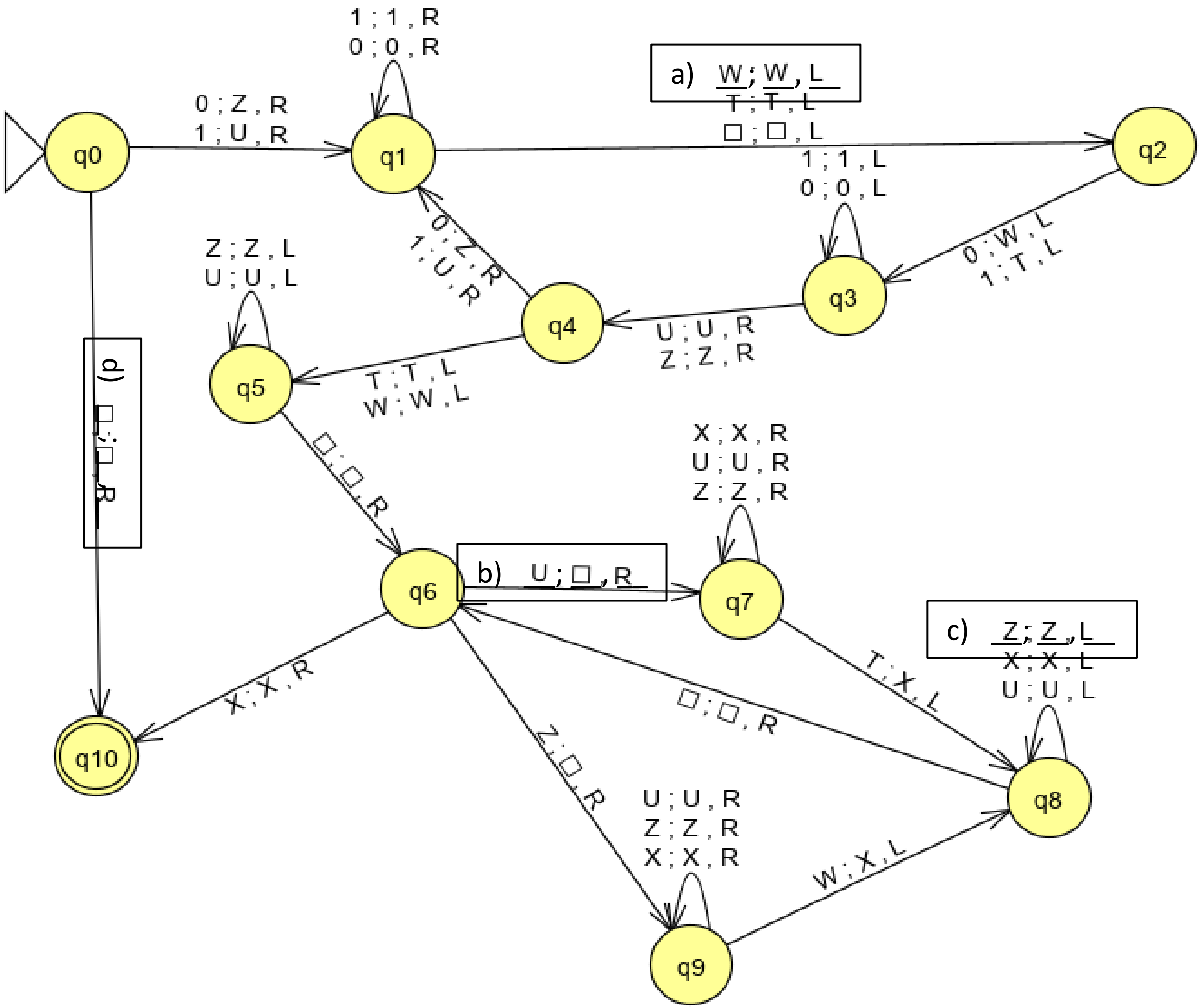
a) On the right

b) That section of the TM is responsible to identify if the input string consists of 0s and 1s and can be decomposed in ww, if so, the result will be a w_1w_2 with w_1 obtained from w by h(0)=Z and h(1)=U and w_2 obtained from w by h(0)=W and H(1)=T. If not the TM halts.

c) First 6 computing traces steps for input 00:

- $q_0 00$ |- $Zq_1 0$ |- $Z0q_1$ |- $Zq_2 0$ |- $q_3 ZW$ |- $Zq_4 W$ |- $q_5 ZW$

d) Remove the transition from q0 to q10



Group IV. [5 Pts] Statements about Languages

(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).

FALSE a) The language A = $\{x \in \{0,1\}^* \mid$ the length of x is odd, and its middle symbol is 1$\}$ is a regular language;

TRUE b) The language represented by the CFG, $S \rightarrow TcT$ and $T \rightarrow Ta \mid Tb \mid \varepsilon$, is a regular language;

FALSE c) The language $B=\{w\#w \mid w \in \{0,1\}^*\}$ is a CFL;

TRUE d) The complement of the language $C = \{0^n1^m \mid n, m \geq 0$ are integers with $n \neq m\}$ is a context-free language;

TRUE e) There is always a deterministic Turing Machine able to implement any language represented by a non-deterministic PDA.

Group IV. [5 Pts] Statements about Languages
(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).
  a) The language A = {x $\in$\{0,1\}* | the length of x is odd, and its middle symbol is 1} is a regular language;

  False.

  Possibility 1: The language is y1z with y,z $\in$\{0,1\}. A $\cap$ L(0*10*) = L(0^n10^n). By the closure property of RLs regarding intersection, and as L(0*10*) is an RL and L(0^n10^n) is not an RL (assuming this has been already proved), A cannot be an RL.*

  Possibility 2: There is no way to have a DFA that can accept a substring in Σ=\{0,1\} of length n and then a 1 and after that another substring Σ=\{0,1\} of length n. For this, the DFA would need to store/count the symbols of the first substring to compare to the number of symbols of the second substring.

Note that we could use the pumping lemma for regular languages to prove that this is a non-regular language. We note however that the justifications to these questions do not need a proof.

Group IV. [5 Pts] Statements about Languages
(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).
  b) The language represented by the CFG, $S \rightarrow TcT$ and $T \rightarrow Ta \mid Tb \mid \varepsilon$, is a regular language;

  *True. The language can be represented by the regular expression: (a+b)*c(a+b)**

Group IV. [5 Pts] Statements about Languages
(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).
    c) The language B=$\{w\#w \mid w \in \{0,1\}^*\}$ is a CFL;

    False.

    This would need that a PDA stores in the stack the left substring of # and then compare it with the right substring of #. As the stack is a LIFO (and the PDA only pushes/pops symbols), the PDA cannot compare the two substrings by the order their symbols appear. It would need to have a FIFO or two stacks.

Group IV. [5 Pts] Statements about Languages

(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).
  - d) The complement of the language C = $\{0^n1^m \mid n, m \geq 0$ are integers with $n \neq m\}$ is a context-free language;

True. $\overline{C} = \{0^n1^n \mid n \geq 0\} \cup L((0+1)^*10(0+1)^*)$

Both $\{0^n1^n \mid n \geq 0\}$ and L((0+1)*10(0+1)*) are CFLs and since the union is a closure operation of CFLs then $\overline{C}$ is also a CFL.

i) (0+1)*10(0+1)* is a regular expression and L((0+1)*10(0+1)*) is an RL and thus a CFL

ii) $\{0^n1^n \mid n \geq 0\}$ can be represented by the CFG: $S \rightarrow 0S1 \mid \varepsilon$, and it is thus a CFL.

Note that L((0+1)*10(0+1)*) = $\overline{L(0^*1^*)}$

Group IV. [5 Pts] Statements about Languages
(T/F: 20%, justification: 80%; wrong answer = reduction of 50%)

- Indicate, justifying succinctly (with a couple of sentences or a counter example), whether each of the following statements is True (T) or False (F).
  e) There is always a deterministic Turing Machine able to implement any language represented by a non-deterministic PDA.

True. The deterministic and the non-deterministic Turing Machines (TM) are computationally equivalent (both recognize the same kind of languages), i.e., the languages known as "recursively enumerable" or Turing-acceptable, Turing-recognizable. As non-deterministic PDAs represent CFLs (note that any PDA, deterministic or non-deterministic, can be converted to a CFG), and CFLs are a subset of the "recursively enumerable" languages, then any language represented by a non-deterministic PDA can be implemented by a deterministic TM.

As any PDA can be simulated/implemented by a software program and thus according to the Church-Turing thesis it is Turing-computable (i.e., implemented/simulated by a deterministic and a non-deterministic Turing machine).