**COT 4210 Fall 2014 Sample Midterm#2 Name:**

**1.** Present a Mealy Model finite state machine that reads an input **x ∈ {0, 1}\*** and produces the binary number that represents the result of adding the twos complement representation of decimal **-6**, that is adding binary **1…1010** to **x** (this assumes all numbers are in two’s complement notation, including results). Assume that **x** is read starting with its least significant digit.
Examples: **00010 → 11100; 11001 → 10011; 01011 → 00101**

****

 **2.**

**a.)**Let **L** be defined as the language accepted by the finite state automaton **A=({A,B,C},{0,1},δ,A,{C})**:

0

**A**

**B**

**C**

**A**

1

1

0

0

Present a right linear grammar that generates the language **L**.

***A → 0A | 0B | 1C***

***B → 1C***

***C → 1A | λ***

**b.)** Consider the regular grammar **G = ({S, A, B}, {0, 1}, R, S)**, where **R** is:

**S → 0 S | 1 A | 0**

**A → 1 B | λ**

**B → 0 S**

Present an automaton **A** that accepts the language generated by the **G**:

***A=({S,A,B, C},{0,1},δ,S,{A,C}) where δ is:***

***I’ll do in class. It’s easy.***

 **3.** Assuming you have computed the sets, **Rki,j**, for each pair of states, **(qi,qj)**, in some DFA. How is **Rk+1i,j** calculated, where **k+1** is no greater than the number of states in the associated DFA?

**Rk+1i,j** = **Rki,j** + **Rki,k+1 ( Rkk+1,k+1 )\* Rkk+1,j**

 **4.** Analyze the following language, **L**, proving it is **non**-regular by showing that there are an **infinite** number of equivalence classes formed by the relation **RL** defined by:
 **x RL y**  if and only if [ **∀z** ∈ **{a,b,c}\***, **xz** ∈ **L** exactly when **yz** ∈ **L ]**.
where **L = { ck an bm | n** < **m, k > 0 }**.
You don’t have to present all equivalence classes, but you must demonstrate a pattern that gives rise to an infinite number of classes, along with evidence that these classes are distinct from one another.

***Consider the set of equivalence classes [caib], i ≥ 0.***

***Clearly caibbi is in L, but cajbbi is not in L, when j>i.***

***Thus, [caib] ≠ [cajb] when j>i leading to an infinite number of equivalence classes induced by RL***

***Consequently L is not Regular.***

 **5.** Write a context-free grammar for the language **L = { ck an bm | n** < **m, k>0 }**. Yes, this is the one you just showed is not Regular.

***G = ( {S,T}, {a,b,c}, R, S ) where R is****:*

***S → c S | cT***

***T → a T b | Tb | b***

 **6.** Which of the following are correct definitions of an ambiguous grammar? Write **T**(true) or **F**(false).

There are two distinct derivations of some string **w** derived by the grammar ***F***

 There are two distinct parse trees for some string **w** derived by the grammar ***T***

 **7.** Use the Pumping Lemma for context-free languages to show **L = { an bn2 | n > 0 }** is **not** a CFL. *Be complete and remember to differentiate what you get to do and what the PL gets to do.*

***P.L: Provide N>0***

***We: Choose aNbN2 in L***

***P.L: Tells us aNbN2 = uvwxy, |vxy| ≤ N and |vx| > 0.***

***We: Choose i=2 and then analyze cases***

***a) Assume vx contains some a’s then uv2wx2y contains at least N+1 a’s and at most N2+N-1 b’s.***

***But (N+1)2 = N2 + 2N + 1. Thus, the number of b’s in uv2wx2y is insufficient in comparison to the number of a’s and hence uv2wx2y is not in L.***

***b) Assume vx contains no a’s the uv2wx2y contains N a’s and at least N2+1 b’s. Thus, the number of b’s in uv2wx2y is too many in comparison to the number of a’s and so uv2wx2y is not in L.***

***This covers all possibilities and hence L is not Context Free***

**8.** Consider the CFG **G = ( { S, T }, { a, b }, R, S )** where **R** is:

**S → a T T | T S | a**

**T → b S T | b**

**a.)** Present a pushdown automaton that accepts the language generated by this grammar. You may (and are encouraged) to use a transition diagram where transitions have arcs with labels of form **a, α → β** where **a** ∈ **Σ∪{λ}**, **α, β ∈ Γ\***. Note: I am encouraging you to use extended stack operations.

What parsing technique are you using? (Circle one) **top-down** or **bottom-up**
How does your PDA accept? (Circle one) **final state** or **empty stack** or **final state and empty stack**
What is the **initial state**? ***q***
What is the **initial stack content**? ***$***
What are your **final states** (if any)? ***f***


 What parsing technique are you using? (Circle one) **top-down** or **bottom-up**
How does your PDA accept? (Circle one) **final state** or **empty stack** or **final state and empty stack**
What is the **initial state**? ***q***
What is the **initial stack content**? ***$***
What are your **final states** (if any)? ***None***
**b.)** Now,using the notation of **ID**s (Instantaneous Descriptions, **[q, x, z]**), describe how your PDA accepts strings generated by **G**.

***[q, w, $] ⇒\* [f, λ, λ] if by final state and empty stack (my solution on (a) Bottom-Up)***

***[q, w, $] ⇒\* [f, λ, $] if by final state (I could have done this on (a) Bottom-Up)***

***[q, w, $] ⇒\* [q, λ, λ] if by empty stack (my solution on (a) Top-Down)***

 **9.** Consider the context-free grammar **G = ( { S, B, E }, { 0, 1, i, e, s }, R, S )**, where **R** is:

**S → i B S E | s**

**B → 0 | 1**

**E → λ | e S**

 **a.)** Remove all **λ**-rules from **G**, creating an equivalent grammar **G**’. Show all rules, including copied ones.

***S → iBS | i B S E | s***

***B → 0 | 1***

***E → e S***

 **b.)** Convert grammar **G’** to its Chomsky Normal Form equivalent, **G’’**. Show all rules, including copied ones from part (a).

***S → <I B> S |< i B> < S E> | s***

***B → 0 | 1***

***E → <e> S***

***<i B> → <i> B***

***<S E>→ S E***

***<e> → e***

***<i> → i***

 **10.** Let **C** be some class of formal languages that is closed under substitution of members of its own class and under intersection with Regular Languages. Prove that **C** is also closed under **RealWrappers**, where **RealWrappers(L) = { xz | x, z ∈Σ+, ∃y∈Σ+, w=xyz ∈ L }**. You may assume substitution **f(a) = {a, a’},**  and homomorphisms **g(a) = a’** and **h(a) = a, h(a’) = λ**. Here **a∈Σ** and **a’** is a new character associated with each **a∈Σ**.

***RealWrappers(L) = h( f(L) ∩ Σ+ g(Σ+) Σ+ )***

 **11.** Fill in the following table with **Y** (yes) or **N** (no) in each cell, depending upon whether or not the class of languages is closed under the stated operation.

|  |  |  |
| --- | --- | --- |
|  | **Regular Languages** | **Context Free Languages** |
| Concatenation with Regular | ***Y*** | ***Y*** |
| Quotient with Regular  | ***Y*** | ***Y*** |
| Complementation | ***Y*** | ***N*** |
| Superset | ***N*** | ***N*** |

 **12.** Present the **CKY** recognition matrix for the string **v + ( v – v)** assuming the Chomsky Normal Form grammar specified by the grammar
**G = ( { E, A, B, T, U, L, R, P, M }, { v, +, –, (, ) }, Rules, E )**, where the **Rules** set is:

**E → E A | E B | L U | v**

**A → P T**

**B → M T**

**T → L U | v**

**U → E R**

**L → (**

**R → )**

**P → +**

**M → –**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **v** | **+** | **(** | **v** | **–** | **v** | **)** |
| **1** | ***E,T*** | ***P*** | ***L*** | ***E,T*** | ***M*** | ***E.T*** | ***R*** |
| **2** |  |  |  |  | ***B*** | ***U*** |  |
| **3** |  |  |  | ***E*** |  |  |  |
| **4** |  |  |  | ***U*** |  |  |  |
| **5** |  |  | ***E,T*** |  |  |  |  |
| **6** |  | ***A*** |  |  |  |  |  |
| **7** | ***E*** |  |  |  |  |  |

***Note: Do not be surprised if the above table is sparsely populated.***

***Hint: Be concerned if it’s densely populated.***