COSC 3340/6309 — Final Examination

Saturday, July 9, 2016, 11 am – 2 pm

Open Book and Notes — **Final grades only through PeopleSoft**

<u>YOU MUST USE THE CONSTRUCTIONS GIVEN IN CLASS</u>

1. Construct a regular expression over {a,b,c} for the language accepted by this nfa:

	a	b	c	
→ A	/	B,C	/	1
B	B	/	C	0
C	/	A,B	/	0

2. Prove that the language L(G) is not regular where G is the following cfg:

$G = (\{S,A,B\}, \{a,b\}, \{S \to Aaa|B,\ A \to aS,\ B \to b\}, S)$.

Note: You <u>must</u> first determine L(G).

3. Construct a reduced dfa for the following extended regular expression over {0,1}:

$[(10^*)^* \cap 0^*10^*]$

Note: You <u>must</u> first determine nfas for (10*)* and 0*10*, then do the intersection. The answer must then be reduced.

4. Construct a **Chomsky** normal form grammar for L(G) for the following cfg G:

$G = (\{S,B\}, \{a,b,c,d\}, \{S \to Sb|Ba,\ B \to cBBd|S|\varepsilon\}, S)$.

Note: You <u>must</u> first remove all ε- and all unit productions.

5. Construct a **Greibach** normal form grammar for L(G) for the following CNF G:

$G = (\{S,A\}, \{a,b\}, \{S \to ASS|A,\ A \to SSS|bab\}, S)$.

Note: You <u>must</u> first remove all unit productions. You <u>must</u> derive all the productions for S and A; indicate how the result looks for S' and A'.

6. Prove that the following language L is not contextfree: $L = \{\ 0^n1^n0^{n+1} \mid n \geq 1\ \}$.

7. Consider the class CF_A of all context free languages over the fixed alphabet A.

(a) Is CF_A countable?

(b) Is the class $NOTCF_A$ countable where $NOTCF_A$ consists of all languages over A that are not context free?

(c) Is the class $CF_A \cap NOTCF_A$ countable?

For each question, you must give a precise argument substantiating your answer.

8. Construct a Turing machine for the language in Question 6, $L = \{\ 0^n1^n0^{n+1} \mid n \geq 1\ \}$.

Note: Describe first the process in English; then translate this into moves of the Turing machine.

9. Let L_1 and L_2 be arbitrary languages, subject to the specification in either (i) or (ii). Consider the following four questions:

(Q1) Does $L_1 - L_2$ contain a given fixed word w? (Q2) Is $L_1 - L_2$ empty?

(Q3) Does $L_1 \cap L_2$ contain a given fixed word w? (Q4) Is $L_1 \cap L_2$ empty?

For each of these four questions **<u>explain with reasons</u>** whether the general problem is <u>recursive</u>, <u>not recursive but r. e.</u>, or <u>non-r. e.</u>, provided

(i) Both L_1 and L_2 are <u>recursive</u>.

(ii) L_1 is <u>r. e., but not recursive</u> and L_2 is <u>recursive.</u>

Note that there are **eight** different questions to be answered.

Points: 1: 6 2: 8 3: 15 4: 11

Hadeel AlMubaireek Final Exam 2016

1571070

Q1:

	a	b	c	
→A	–	B,C	–	0
B	B	–	C	0
C	–	A,B	–	1

$L_A = bL_B \cup bL_C$

$L_B = aL_B \cup cL_C$

$L_C = bL_A \cup bL_B \cup \varepsilon$

1) Subsitute L_C in L_B:

$L_B = aL_B \cup c(bL_A \cup bL_B \cup \varepsilon)$

$L_B = aL_B \cup cbL_A \cup cbL_B \cup c$

$L_B = (a \cup cb)L_B \cup (cbL_A \cup c)$ $(L^* \cup M) = L^*M$

$\therefore L_B = (a \cup cb)^* (cbL_A \cup c) = (a \cup cb)^* cbL_A \cup (a \cup cb)^* c$

2) Subsitute L_C in L_A:

$L_A = bL_B \cup b(bL_A \cup bL_B \cup \varepsilon) = bL_B \cup bbL_A \cup bbL_B \cup b$

3) Subsitute L_B in L_A:

$L_A = b((a \cup cb)^* cbL_A \cup (a \cup cb)^* c) \cup bbL_A \cup bb((a \cup cb)^* cbL_A \cup (a \cup cb)^* c) \cup b$

$= \underline{b(a \cup cb)^* cbL_A} \cup b(a \cup cb)^* c \cup \underline{bbL_A} \cup \underline{bb(a \cup cb)^* cbL_A} \cup bb(a \cup cb)^* c \cup b$

$= (b(a \cup cb)^* cb \cup bb \cup bb(a \cup cb)^* cb)^* (b(a \cup cb)^* c \cup bb(a \cup cb)^* c \cup b)$

Q2:

$S \to Aaa \mid B$
$A \to aS$
$B \to b$

$S \to Aaa \to aSaa \to aAaaaa \to aaSaaaa$

$S \to b \to b$

$aSaa \to aBaa \to abaa$

$aaSaaaa \to aaBaaaa \to aabaaaa$

$\therefore L(G) = \{ a^i b a^j \mid j \geq i \geq 0 \}$

- To prove that $L(G)$ is not regular, let's assume that $L(G)$ is a regular language.
- Hence, there exists a dfa $\underset{\sim}{D}$ that has n states that accepts the language $L(G)$. $L(G) = L(\underset{\sim}{D})$
- Consider $X = w \cdot v = a^i b a^j$, where $w = a^i$, and $v = b a^j$
- Since $|w| = n$, we can apply the pumping lemma:
  $w = w_1 w_2 w_3$, such that $|w_2| \geq 1$
- So, $\tau(q_0, w) = \tau(q_0, w_1 (w_2)^s w_3)$, where $s \geq 0$ is $\in L(G)$
- let's assume $s = 0$:
  - we have $\tau(q_0, w) = \tau(q_0, w_1 w_3)$, but $|w_1 w_3| = n - |w_2| < n$
  - Therefore, $\tau(q_0, w_1 w_3) \notin L(G)$

Since $\tau(q_0, w) \in L(G)$, and $\tau(q_0, w_1 w_3) \notin L(G)$, therefore, we have a contradiction.

Thus, $L(G)$ is <u>NOT</u> regular.

Q3: $[\overline{(01^*)^*} \cap 1^*01^*] \equiv [\overline{(01^*)^* \cup \overline{(1^*01^*)}}]$

	0	1	
0	1	–	0
1	–	–	1

	0	1	
0	–	2	0
2	–	–	1

	0	1	
0	–	3	0
3	–	–	1

	0	1	
0	4	–	0
4	–	–	1

	0	1	
0	–	5	0
5	–	–	1

$\downarrow *$

	0	1	
0	–	2	1
2	–	2	1

$\downarrow *$

	0	1	
0	–	3	1
3	–	3	1

$\downarrow *$

	0	1	
0	–	5	1
5	–	5	1

	0	1	
0	1	–	0
1	–	2	0
2	–	–	1

$\downarrow *$

	0	1	
0	1	–	1
1	–	2	0
2	–	–	1

	0	1	
→0	4	3	0
3	4	3	0
4	–	5	1
5	–	5	1

$\downarrow$ dfa

	0	1		
→0	4	3	0	1
4	–	5	1	0
3	4	3	0	1
–	–	–	0	1
5	–	5	1	0

flip

	0	1	
0	1	—	1
1	—	2	0
2	1	—	1

∪

	0	1	
0	4	3	1
4	—	5	0
3	4	3	1
—	—	—	1
5	—	5	0

Assume ϕ = —

=

	0	1	
→ 0	14	3	1
1	ϕ	2	0
2	1	ϕ	1
4	ϕ	5	0
3	4	3	1
ϕ	ϕ	ϕ	1
5	ϕ	5	0

1ϕ →

		0	1	
→ A	0	14	3	1
B	14	ϕ	25	0
C	3	4	3	1
D	ϕ	ϕ	ϕ	1
E	25	1ϕ	5ϕ	1
F	4	4ϕ	5	0
G	1ϕ	ϕ	2ϕ	1
H	5ϕ	ϕ	5ϕ	1
I	5	ϕ	5	0
J	2ϕ	1ϕ	ϕ	1

Hypt renames →

	0	1	
A	B	C	0
B	D	E	1
C	F	C	0
D	D	D	0
E	G	H	0
F	D	I	1
G	D	J	0
H	D	H	0
I	D	I	1
J	G	D	0

0			1
ACDEGHJ			BFI
ADEGHJ		C	
A	DEGHJ	C	
A	DEGHJ	C	
1	2	3	4

reduced

	0	1
.	4	3
2	2	2
3	4	3
4	2	2

Q4.

$S \rightarrow Sb \mid Ba$

$B \rightarrow cBBd \mid S \mid \varepsilon$

1) Eliminate $B \rightarrow \varepsilon$

$S \rightarrow Sb \mid Ba$

$B \rightarrow cBBd \mid cBd \mid cd \mid S$

2) Eliminate $B \rightarrow S$ (replace every B with S)

$S \rightarrow Sb \mid Ba \mid Sa$

$B \rightarrow cBBd \mid cSBd \mid cBSd \mid cSSd \mid cBd \mid cSd \mid cd$

$S \rightarrow S X_b \mid B X_a \mid S X_a$

$B \rightarrow X_c BB X_d \mid X_c SB X_d \mid X_c BS X_d \mid X_c SS X_d \mid X_c B X_d \mid X_c S X_d \mid X_c X$

$\rightarrow S \rightarrow S X_b \mid B X_a \mid S X_a$

$B \rightarrow X_c B_1 \mid X_c B_3 \mid X_c B_4 \mid X_c B_6 \mid X_c B_2 \mid X_c B_5 \mid X_c X_d$

$B_1 \rightarrow B B_2$

$B_2 \rightarrow B X_d$

$B_3 \rightarrow S B_2$

$B_4 \rightarrow B B_5$

$B_5 \rightarrow S X_d$

$B_6 \rightarrow S B_5$

$X_a \rightarrow a$

$X_b \rightarrow b$

$X_c \rightarrow c$

$X_d \rightarrow d$

Q5:

S → AS | A

A → SSS | aba

1) Eliminate S → A (replace S with A)

S → AS | AA

A → SSS | ASS | SAS | SSA | AAS | SAA | ASA | AAA | aba

2) replace S with AS | AA in A

A → ASSS | AASS | ASS | ASAS | AAAS | ASSA | AASA | AAS

ASAA | AAAA | ASA | AAA | aba

3) Eliminate left recursion:

A → aba | abaA'.

A' → SSS | ASS | SS | SAS | AAS | SSA | ASA | AS | SAA | AAA | SA | AA |

SSSA' | ASSA' | SSA' | SASA' | AASA' | SSAA' | ASAA' | ASA' | SAAA' |

AAAA' | SAA' | AAA'

4) Find the production of S:

S → abaS | abaA'S | abaA | abaA'A

Q6: $L = \{0^n 1^n 0^{n+1} \mid n \geq 1\}$

- Assume that is a Context free language, then $\exists$ a $G(N, T, P, S)$ in CNF such that $L = L(G)$.
- Consider the case where $n = 2^m$.
- So, $Z = 0^{2^m} 1^{2^m} 1^{2m+1}$, $|Z| > 2^m$ and $Z \in L$, then, by Pumping lemma for CFL, we have, $Z = uvwxy$, and $|zv| \geq 1$ and $uv^i w z^i y \in L$, $\forall i \geq 0$.

Case 1: v and x has only left 0's:

- we increase the number of left 0's, while the number right 0's remains the same.
- for $i = 2$, # left 0's $\geq$ # right 0's. (Contradiction)

Case 2: v and x has only 1's.

- for $i = 2$, we increase the number of 1's, while The number of right 0's remains the same.
- # 1's $\geq$ # right 0's. (Contradiction)

Case 3: v and x has only right 0's:

- for $i = 0$, we decrease the number of right 0's, while of 1's remains the same.
- # right 0's $\leq$ # 1's (Contradiction)

Case 4: v or x has 1's and right 0's, but no left 0's:

- for $\delta = 0$, we decrease the number of right 0's, while, the number of left 0's remains the same.

- # left 0's $\geq$ # right 0's. (contradiction)

Case 5: v or x has NO 1's

- for $\delta = 0$, we decrease the number of right 0's, while the number of 1's remains the same.

- # right 0's $\leq$ # 1's (contradiction)

Case 6: v or x has NO right 0's:

- for $\delta = 2$, we increase the number of 1's, while the number of right 0's remains the same.

- # 1's $\geq$ # right 0's. (contradiction)

Case 7: v contains more than one left 0's, 1's, right 0's.

- for $\delta = 2$, we get the 0 following 0. (contradiction)

Case 8: x contains more than one left 0's, 1's, right 0's.

- for $\delta = 2$, we get the 0 following 0.

Q7:

a) is CF_A Countable?

- Since CF_A is a context free language, we know that there must be at least one Pda that accepts CF_A. And we know that pda is a finite automaton. Hence, $\boxed{CF_A \text{ is Countable.}}$ (1)

b) is $NOTCF_A$ Countable?

- We know that CF_A is a context free language, We also know that $NOTCF_A$ consists of all languages that are not context free.
- Therefore, we don't know if $NOT CF_A$ is context free or not. It may be very complex, or it may be a problem or infinite.
- Also, let A^* indicates all possible combinations of the fixed alphabet:

$\therefore A^*$ is Countable infinite

$\therefore 2^{A^*}$ is also countable infinite.

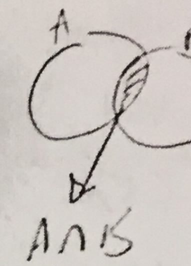
$\therefore CF_A$ is countable

$\therefore 2^{A^*} - CF_A = NOTCF_A$, which is infinite.

Thus, we cannot account for the languages represented by $NOTCF_A$, so $\boxed{\text{It's NOT Countable.}}$ (2)

c) Is $REG_A \cap NOTREG_A$ Countable?

- Intersection states that an element is in $A \cap B$ iff:
  1) The element is in A AND
  2) The element is in B



- But since $NOTCF_A$ consists of all languages that are not in CF_A, no element can be in both A and B.

- And by the difinition of intersection, $REG_A \cap NOTREG_A$ will yield to an empty set.
- The empty set is considered to be a finite set with cardinality of zero. $|\{\emptyset\}| = 0$ which is a countable number.

3

- Therefore $REG_A \cap NOTREG_A$ is Countable

Q8: $L = \{0^n 1^n 0^{n+1} \mid n \geq 1\}$

left 0's right 0's.

Number left 0's is equal to number of 1's, and <u>less</u>
The the number of right 0's.

1) Configure the Turing Machine from the left, find the first left 0 and change it to 0'.

2) Then, move right until you find a 1, and change it to 1'.

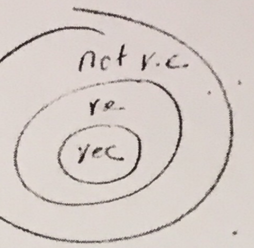
3) Then move left until you find the first 0', then stop and right until you find the first 0.

4) Repeat the process until 0' is the next state from 1 and then find all 0 and change them to 0'.

	0	1	0'	1'	B
q_0	$(q_1, 0', R)$	–	–	–	–
q_1	$(q_1, 0, R)$	$(q_2, 1', R)$	–	–	–
q_2	–	$(q_3, 1', L)$	–	–	–
q_3	$(q_3, 0, L)$	–	$(q_4, 0', R)$	$(q_3, 1', L)$	–
q_4	$(q_4, 0, R)$	$(q_5, 1', R)$	–	$(q_4, 1', R)$	–
q_5	$(q_6, 0', L)$	$(q_5, 1, R)$	$(q_5, 0', R)$	–	q_f
q_6	–	$(q_6, 1, L)$	$(q_6, 0', L)$	$(q_7, 1', L)$	–
q_7	$(q_7, 0, L)$	–	$(q_8, 0', R)$	$(q_7, 1', L)$	–
q_8	$(q_4, 0', R)$	–	–	–	–
q_f	Accepting state!				

Scanned by CamScanner

Q4:



i) L_1 and L_2 are <u>both</u> recursive : (i) ONLY!!

Q1: $L_1 - L_2$ contains a given fixed word w?

$L_1 - L_2$ is recursive because recursive languages are closed under set difference. A TM for deciding if a word is in a recursive language is recursive. which means always halt for a given fixed word w belongs L_1 but not L_2.

Q2: $L_1 - L_2$ empty? $L_1 - L_2 = \emptyset$?

$L_1 - L_2 = \emptyset$ if $L_1 = L_2$. To decide $L_1 - L_2 = \emptyset$, the TM has to run for ever to enumerate all the possibilities with no guarantee to stop. Then, $L_1 - L_2$ is not re

Q3: $L_1 \cap L_2$ contains a given fixed word w?

Since both L_1 and L_2 is recursive, TM for $L_1 \cap L_2$ can always answer yes for a fixed word w in $L_1 \cap L_2$ and say no when $w \notin (L_1 \cap L_2)$. Also, recursive languages are closed under intersection, therefore $L_1 \cap L_2$ is recursive

Q4: $L_1 \cap L_2$ empty? $L_1 \cap L_2 = \emptyset$?

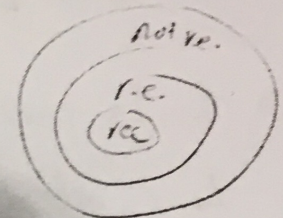
TM for $L_1 \cap L_2$, simulate input word w for $L_1 \cap L_2$ when $w \in L_1 \cap L_2$, and $w \notin \emptyset$, the TM halts and answer yes, but for $L_1 \cap L_2 = \emptyset$, The TM has to go through all cases for $L_1 \cap L_2$ and runs for ever and say no

$\models L_1 \cap L_2 = \emptyset$ is not

ii) L_1 is recursive, but L_2 is r.e. but not recursive.

Q1: $L_1 - L_2$ contains a given fixed word w?

The problem is $\boxed{\text{recursive}}$, both closed under set difference.



Q2: $L_1 - L_2$ is empty? $L_1 - L_2 = \emptyset$? if $L_1 = L_2$

∵ L_1 is recursive and L_2 is r.e. but not recursive,
There exist a TM to prove whether the language contains a word or not, therefore, the problem
$\boxed{\text{is r.e. but not recursive.}}$

Q3: $L_1 \cap L_2$ contains a given fixed word?

since both L_1 and L_2 are closed under intersection, and when $w \in L_1$ and $w \in L_2$, the TM for $L_1 \cap L_2$ always answers yes on input w.
∴ $\boxed{L_1 \cap L_2 \text{ is r.e. but not recursive.}}$

Q4: $L_1 \cap L_2$ is empty? $L_1 - L_2 = \emptyset$?

∵ L_1 is recursive and L_2 is r.e. but not recursive,
Then exists a TM to prove whether the language contains a word or not, therefore the problem is
$\boxed{\text{r.e. but not recursive}}$