Nikolay Pavlovich Laptev
Problem Set #8

Exercise #1

(a)

Proof:
Assume that L_1 is context free. Let n be the "magic number" of the Pumping Lemma

Consider $L_1 = 0^n 1^{2n} 0^n$ in L_1. Then ∃ subdivision z = uvwxy, 0 < |vwx| ≤ n, $vx \neq \varepsilon \neq w, uv^i wx^i y \in L_1$ for all $i \geq 0$.

Words in L_1 contain EXACTLY 3 blocks of numbers (1 block of 0s, 1 block of 1s and 1 block of 0s again). (Also the middle block is twice as long as its two neighboring blocks)

The length condition $|vwx| \leq n$ implies for v,x. v starts in one block and x must lie in the same or the following block. v or x is nonempty.

So if we increment i, then our z would look like uvvwxxy which increases the size of at least one block, but leaves size of at least one block unchanged, hence uvvwxxy is not in L_1.

All cases lead to a contradiction so L_1 is not context-free.

(b)
Intuition:
Similarly to the part *(a)* we cannot *keep track* of more than two variables, hence is the problem. (At first I tried proving the problem with $L_2 = 0^n 1^n 1^n 0^n$ however that does not seem to work. Because we can still pump (0,1) and the resulting string will still be in the language L_2.

Proof:
Assume that L_2 is context free. Let n be the "magic number" of the Pumping Lemma

Consider $L_2 = 0^n 1^n 0^n 1^n 0^n 1^n$ in L_2. Then ∃ subdivision z = uvwxy, 0 < |vwx| ≤ n, $vx \neq \varepsilon \neq w, uv^i wx^i y \in L_1$ for all $i \geq 0$.

Words in L_2 contain EXACTLY 6 consecutive blocks of numbers (1 block of 0s, 1 block of 1s,1 block of 0s, 1 block of 1s, 1 block of 0s and lastly 1s again).

The length condition $|vwx| \leq n$ implies for v,x. v starts in one block and x must lie in the same or the following block. v or x is nonempty. So if we increase any one block or any two blocks we leave at least 2 other blocks unchanged, which produces a string which is not in L_2. Notice if we would have picked $L_2 = 0^n 1^n 1^n 0^n$ then it would be possible to

increase 0s and also increase 1s in x or wx, thus the resulting string would be in the language.

So if we increment i, then our z would look like uvvwxxy which increases the size of at least one block, but leaves size of at least one block unchanged, hence uvvwxxy is not in L_2.

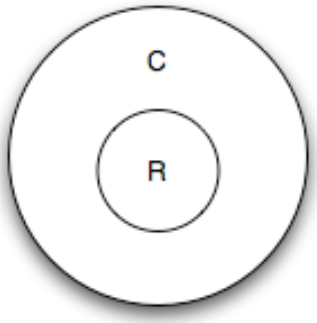
All cases lead to a contradiction so L_2 is not context-free.

Exercise #2

(a) *Never*
We know that is L is a CFL, then so is L^R, however we know that L is not a context free language.

Suppose L^R is context free then $\left(L^R\right)^R = L$ must also be context free, however we know that L is not context free, therefore we have a contradiction.

(b) *Always*
$C - R$ implies that the resulting language is no bigger than C, and C is context free, therefore C-R must also be context free.



(c) *Sometimes*
Note: Finite state languages are also context free.

1) $L \cap \varepsilon = L$ here L is not context free

2) $L \cap \sum^* = \sum^*$ which is finite state and therefore context free.

Exercise 3

(a)

$S \rightarrow 0X001000$

$X \rightarrow 0X00B \mid 1$

$B0 \rightarrow 0B$

$B1 \rightarrow 1000$

How it works: We start off in S from the case when i=1, then as we keep on adding 0s in the first block we also add two zeroes in the second as well as non-terminal B which will keep track of how many 0s we need in the 3rd block. We then move all B's to the right, and convert them to 3 0s when they touch the 2nd 1.

(b)

$S \rightarrow AX \mid BY$

$A \rightarrow 0AC \mid 0 \mid 1 \mid AD$

$B \rightarrow 0BC \mid 1 \mid 1BD$

$DX \rightarrow 1X$

$CX \rightarrow 0X$

$DY \rightarrow 1Y$

$CY \rightarrow 0Y$

$X \rightarrow 0$

$Y \rightarrow 1$

$D0 \rightarrow 0D$

$D1 \rightarrow 1D$

$C0 \rightarrow 0C$

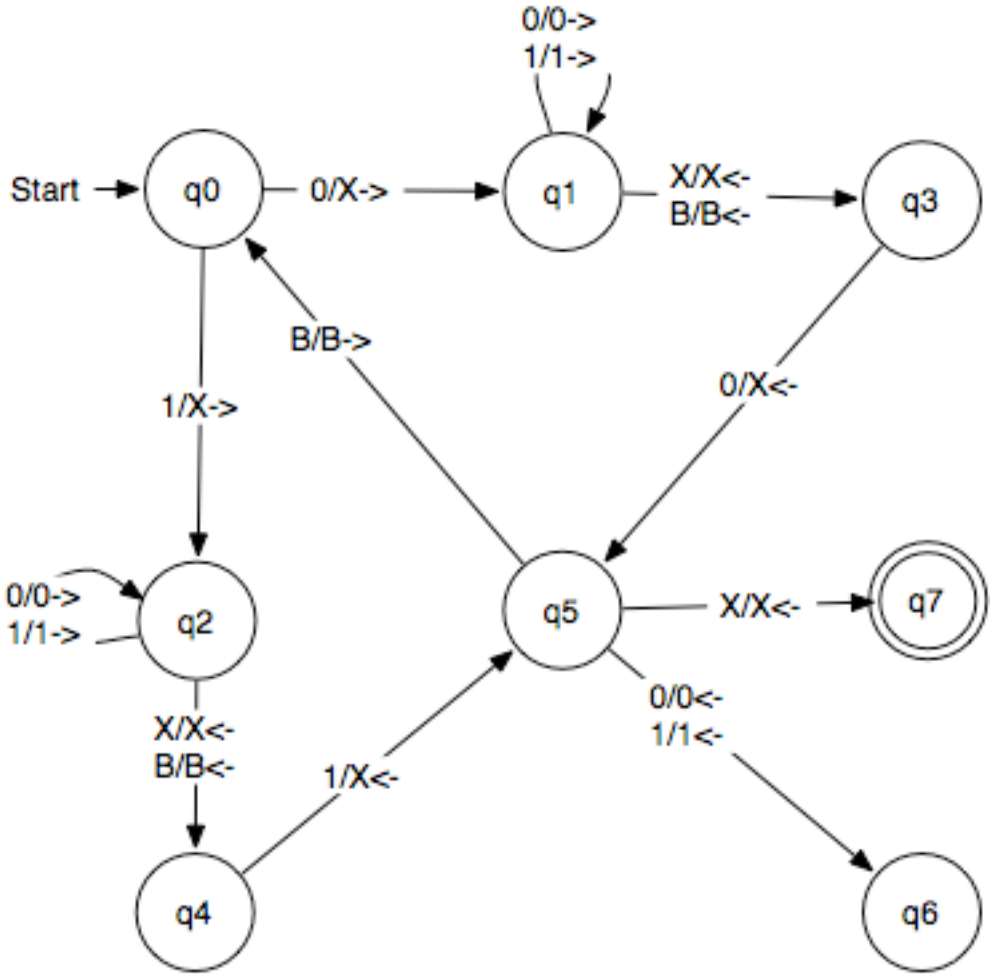
$C1 \rightarrow 1C$

Idea: We know we can easily generate ww^R very easily. So in the grammar above we generate precisely that but we generate w^R part as non-terminals when then rearrange to make ww.

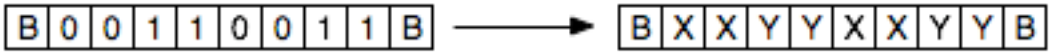
Exercise 4

(a) *Description:* We read the first character and then travel to the very end of the tape and see if the last character matches the first character. We check by replacing them with a new character "X". Then we do the same for 2nd and 2nd to last characters. We terminate in an accepting state when to the left and to the right we have "Xs".

State	0	1	X	B
q_0	(q_1,X,R)	(q_2,X,R)	-	-
q_1	$(q_1,0,R)$	$(q_1,1,R)$	(q_3,X,L)	(q_3,B,L)
q_2	$(q_2,0,R)$	$(q_2,1,R)$	(q_4,X,L)	(q_4,B,L)
q_3	(q_5,X,L)	$(q_6,1,L)$	-	-
q_4	$(q_6,0,L)$	(q_5,X,L)	-	-
q_5	$(q_6,0,L)$	$(q_6,1,L)$	(q_7,X,L)	(q_0,X,R)
q_6	$(q_6,0,L)$	$(q_6,1,L)$	(q_0,X,R)	(q_0,X,R)
(ACCEPT) q_7	-	-	-	-



(b)



1) We start from the left and move to the right. We see a 0, we switch a state and continue moving right until we reach “X” or “B”
2) Once we see “X” or “B” we move left until we see a 0, in which case we mark it “X”. and continue moving left until we reach our previously marked “X”
3) (Also along the way we must make sure that we see block of 0s followed by block of 1s only 2 times)
4) We keep on marking 0s with Xs until we have no more zeroes to mark in which case we are done with 0s and now we need to do a similar procedure to 1s.
5) We mark the left-most 1 with Y and move right until we reach B. We mark the right-most 1 with Y as well, then we move left (skipping all Xs which we marked before.
6) We accept when there are no more 0s or 1s left and all of them have been marked with X or a Y and we made sure that all blocks appear in the correct order.

Exercise 5

(a) *Decidable* – If we let R=(1*(0+1)*0) and we construct a machine M=L(M)-R (by using cross product and reversing the accept states of L(M)) we can test for emptiness of M and if it is empty then L(M) is contained in R and vice versa.

(b) *Undecidable* – We know that containment is undeciadable for CFGs.

(c) *Undecidable* – Well if L(T) is decidable then halting problem would also be decidable. In other words if L(T) = $\sum^*$ is decidable then we can certainly reduce the halting problem to precisely this (i.e. If we know that turing machine which accepts infinite alphabet is decidable then we know when it will halt/accept even if it takes infinite amount of time, therefore halting problem can be decided in this way), however we know that halting problem is undecidable, therefore a contradiction.