Fundamentals of Computing
Coursework 1

Birkbeck, University of London

Baran Buluttekin
ID No. 13153116

I have read and understood the sections of plagiarism in the College Policy on assessment offences and confirm that the work is my own, with the work of others clearly acknowledged. I give my permission to submit my report to the plagiarism testing database that the College is using and test it using plagiarism detection software, search engines or meta-searching software.

1. (a) Truth table for F.

We can use $A \to B = \neg A \vee B$ to transfom the F.

In this case F:

$$F = \neg(B \to A) \vee ((A \wedge B \wedge C) \vee (\neg B \wedge C))$$

$$F = \neg(\neg B \vee A) \vee ((A \wedge B \wedge C) \vee (\neg B \wedge C))$$

We can transfom one more tim with De Morgan laws $\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$ then;

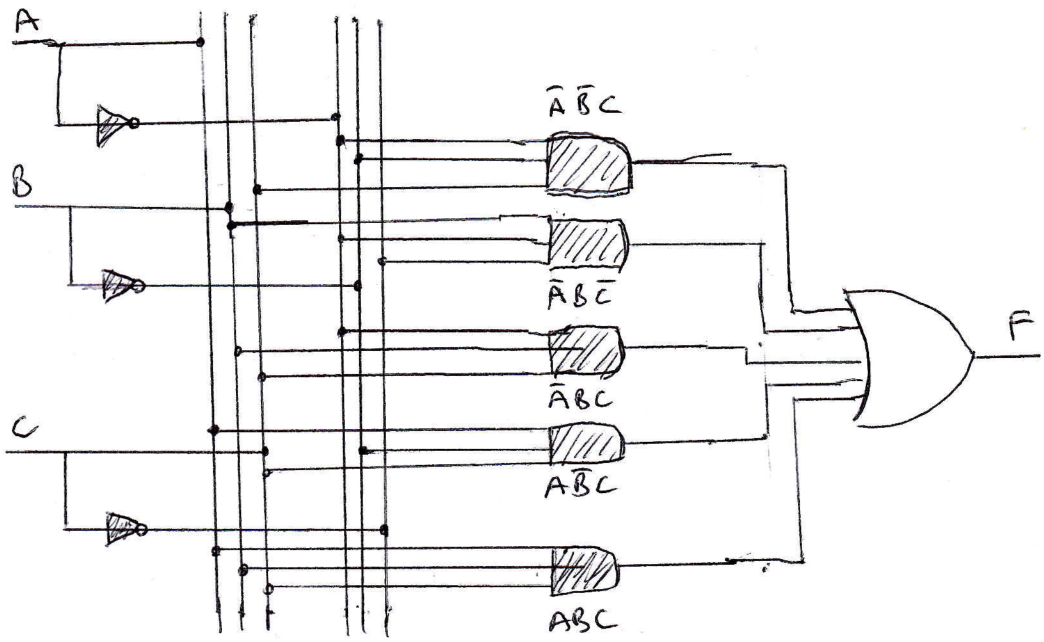
$$F = \underbrace{(B \wedge \neg A)}_{1} \vee (\underbrace{(A \wedge B \wedge C)}_{2} \vee \underbrace{(\neg B \wedge C)}_{3})$$

I divided and labelled sub logic as 1, 2, 3 to make it easier in the truth table.

A	B	C	¬A	¬B	1	3	2	2∨3	F
0	0	0	1	1	0	0	0	0	$\boxed{0}$
0	0	1	1	1	0	1	0	1	1
0	1	0	1	0	1	0	0	0	1
0	1	1	1	0	1	0	0	0	1
1	0	0	0	1	0	0	0	0	0
1	0	1	0	1	0	1	0	1	1
1	1	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	1	1	1

(b) Function F can be realised as a disjunction of five conjunctions:

$$(\neg A \wedge \neg B \wedge C) \vee (\neg A \wedge B \wedge \neg C) \vee (\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee (A \wedge B \wedge C)$$



(c) We can costruct a truth table to examine the argument. We can call additional argument N.

$$N = (C \rightarrow A) \wedge (A \rightarrow B)$$

A	B	C	$C \rightarrow A$	$A \rightarrow B$	N	F	$F \wedge N$
0	0	0	1	1	1	0	0
0	0	1	0	1	0	1	0
0	1	0	1	1	1	1	1
0	1	1	0	1	0	1	0
1	0	0	1	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	1	1	1	0	0
1	1	1	1	1	1	1	1

As we can see B is 1 where the argument is true.

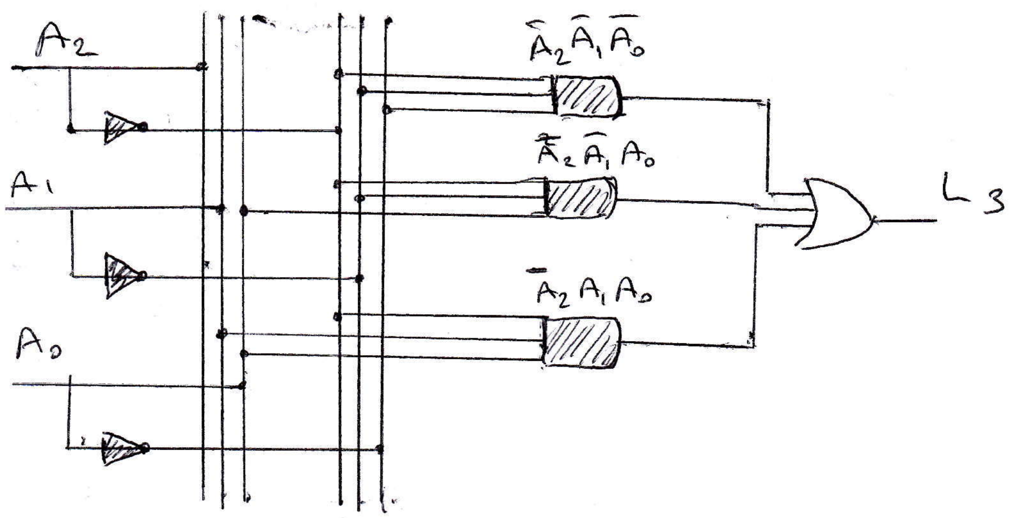
We can conclude that argument is logically correct.

2. Truth table for A_2, A_1, A_0 is:

A_2	A_1	A_0	Less than 3
0	0	0	1
0	0	1	1
0	1	0	1
1	0	1	0
0	1	1	0
1	0	0	0
1	1	0	0
1	1	1	0

Function for less than 3:

$$L_3 = (\neg A_2 \wedge \neg A_1 \wedge \neg A_0) \vee (\neg A_2 \wedge \neg A_1 \wedge A_0) \vee (\neg A_2 \wedge A_1 \wedge A_0)$$



3. Functions from the statements:

$$A \to B \vee C$$
$$\neg B \to A \wedge D$$
$$D \to B \vee C$$

we can transform them with De Morgan laws.

1. $\neg A \vee (B \vee C)$
2. $B \vee (A \wedge D)$
3. $\neg D \vee (B \vee C)$

The argument is not correct. We can construct a boolean table and we can find that.

A	B	C	D	$A \to B \vee C$	$\neg B \to A \wedge D$	$D \to B \vee C$
1	0	1	1	1	1	1

holds true. Therefore we can say not all $B = 1$ satisfies the three condition, which is required to say that argument is correct.

4. (a) N = 1100 0001 1011 0000 0000 0000 0000 0000

As a first alternative, we can use the formula in page 10 of the FoC-2.pdf. We will get following computation.

$$N = -2^{31} + 2^{30} + 2^{24} + 2^{23} + 2^{21} + 2^{20}.$$

Alternatively, we can calculate it with inverting the bits and adding 1.

$$\begin{array}{r} 0011\ 1110\ 0100\ 1111\ 1111\ 1111\ 1111\ 1111 \\ 1 \\ \hline 0011\ 1110\ 0101\ 0000\ 0000\ 0000\ 0000\ 0000 \end{array}$$

Then, $-N = 2^{29} + 2^{28} + 2^{27} + 2^{26} + 2^{25} + 2^{22} + 2^{20}$

(b) $2^{31} + 2^{30} + 2^{24} + 2^{23} + 2^{21} + 2^{20}$

(c) $S = 1$

$E = 10000011 = 131 - 127 = 4$

$F = 011\ 0000\ 0000\ 0000\ 0000\ 0000$

$$N = (-1)^S \cdot 2^{Exp - Bios} \cdot (1 + F)$$

$$= -1 \cdot 2^4 \cdot 1.011_2$$

$$= -1.10110_2$$

$$= -22_{10}$$

5. (a) First we will find unsigned binary equal of the number by continuously.

	Remainder
$107/2 = 53$	1
$53/2 = 26$	1
$26/2 = 13$	0
$13/2 = 6$	1
$6/2 = 3$	0
$3/2 = 1$	1
$1/2 = 0$	1

$107_{10} = 1101011_2$

we can write as 32 bit binary as;

0000 0000 0000 0000 0000 0000 0110 1011

substract 1 from the number;

0000 0000 0000 0000 0000 0000 0110 1010

invert the bits;

1111 1111 1111 1111 1111 1111 1001 0101

(b) From section (a) we know $107_{10} = 1101011_2$

Since the number is negative S should be -1.

$1101011_2 = 1.101011 \times 2^6$ then

$E = 127 + 6 = 133_{10} = 10000101_2$

$-107 = 1100\ 0010\ 1101\ 0110\ 0000\ 0000\ 0000\ 0000$

(c) We can convert -15.375_{10} to binary.

$15_{10} = 1111_2$ $\quad 0,375_{10} = 0.011_2$

$-15.375_{10} = -1111,011_2 = -1,111011_2 \times 2^3$

For IEEE 754 $\quad S = -1 \quad E = 3 + 127 = 130_{10} = 10000010_2$

$-15,375 = 1100\ 0001\ 0111\ 0110\ 0000\ 0000\ 0000\ 0000$

6. (a)

$$f(x) = \begin{cases} 3^{-x}, & x < 0 \\ 2^{x}, & x \geqslant 0 \end{cases}$$

(b) $f(x) = |x|$

(c)

$$f(x) = \begin{cases} -2x-1, & x < 0 \\ 0, & x = 0 \\ 2x, & x \geqslant 0 \end{cases}$$

(d) $f(x) = 123$

7. (a) for $x = 2$ it's not defined as it will give division by zero. Not a function

(b) It is onto but not one-to-one because

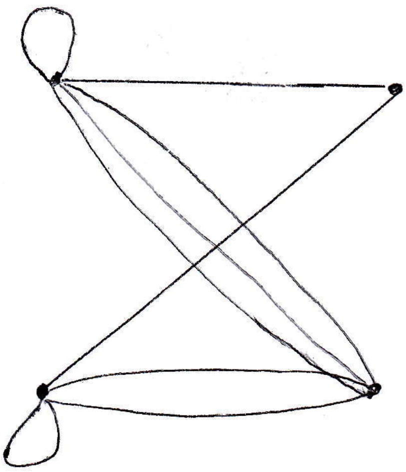
$f(m,n) = f(n,m)$

(c) Also onto but not one-to-one because

$f(m,n) = f(-m,-n)$

8.

(a)



(b) G_1 and G_2 graphs are isomorphic and G_3 is not. G_2 and G_1 share 3 node subgraph property but this property is not present in the G_3 graph.

9. (a) input: bb

$(s, bb), (p, bb), (q, b), (p, b), (q, \varepsilon)$ accepted

$(s, bb), (q, b), (p, b), (q, \varepsilon)$ accepted

input: aa

$(s, aa), (s, a), (s, \varepsilon)$ stuck

$(s, aa), (p, a),$ stuck

input: ab

$(s, ab), (s, b), (q, \varepsilon)$ accepted

$(s, ab), (p, ab)$ stuck

$(s, ab), (s, b), (p, b) (q, \varepsilon)$ accepted

input: aba

$(s, aba), (s, ba), (p, ba), (q, a), (q, \varepsilon)$ accepted

$(s, aba), (p, aba)$ stuck

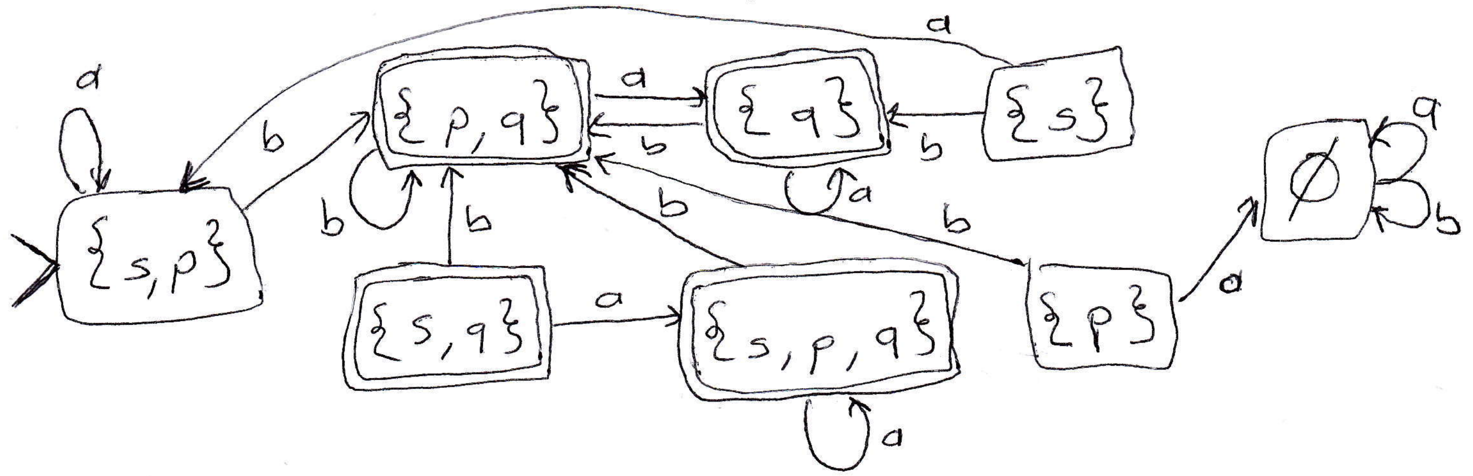
$(s, aba), (s, ba), (q, a), (q, \varepsilon)$ accepted

input: ε

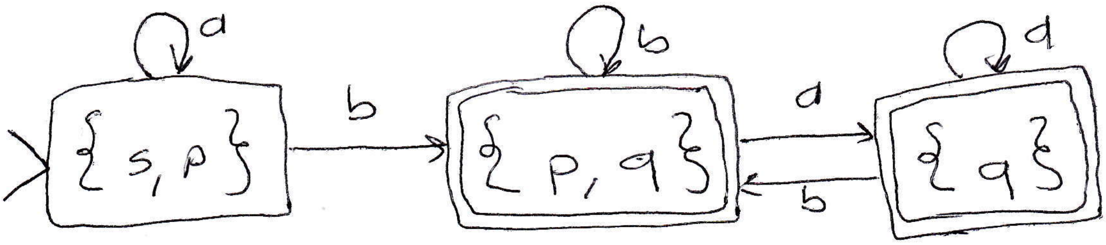
(s, ε) stuck

$(s, \varepsilon), (p, \varepsilon)$ stuck (Not accepted)

9. (b) The new states are $\emptyset$, $\{s\}$, $\{p\}$, $\{q\}$, $\{s,p\}$, $\{s,q\}$, $\{p,q\}$, $\{s,p,q\}$. The initial state is $\{s,p\}$. The favorible states are $\{q\}$, $\{s,q\}$, $\{p,q\}$ and $\{s,p,q\}$



if we remove unreachable states, we will obtain:

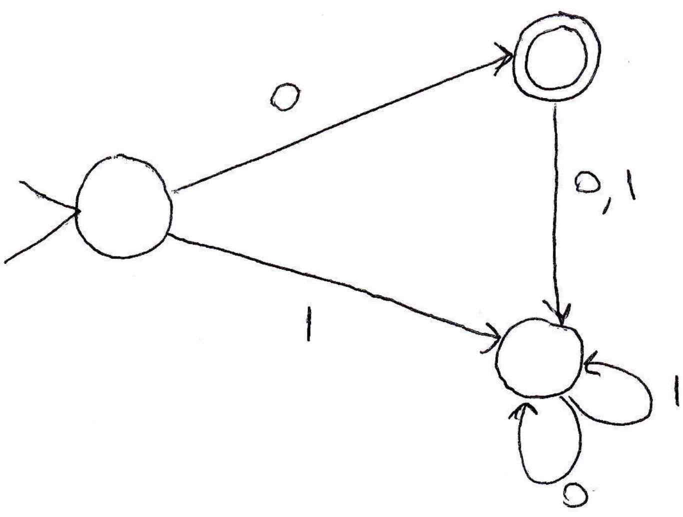


(c) Regular expression: $L(A) = L[a^* b(a \cup b)^*]$

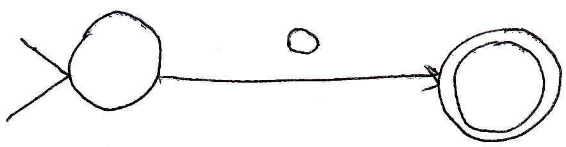
(d) Context-free:

$S \to AbB$, $A \to \varepsilon$, $A \to Aa$, $B \to \varepsilon$, $B \to Ba$, $B \to Bb$

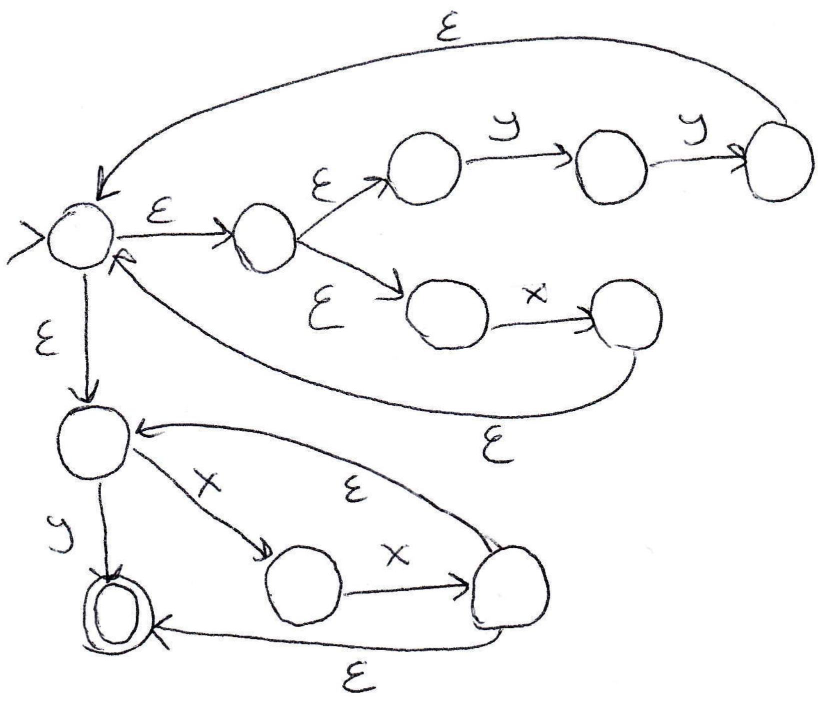
10. (a)



(b)



11.



12. For aabcc, $i=k$ and the grammar,

(a)

$S \to aSc$

$S \to X$

$X \to b$

$X \to \varepsilon$

If there was a case for $i=j$ then,

$S \to Sc$

$S \to A$

$A \to aAb$

$A \to \varepsilon$

(b) This language does not satisfies pumping lemma rules and not a regular language. It doesn't hold $|xyv| \leq P$ rule for context-free pumping lemma rule.

$\{ a^i b^j a^i \mid i \geq 0, j \geq 3 \}$

14. (i)

input : 10

$(s, \triangleright \underline{1}0), (q, \triangleright 1\underline{0}), (q, \triangleright 10\underline{\sqcup}), (p, \triangleright 1\underline{0}), (p, \triangleright 10\underline{\sqcup}),$

$(h, \triangleright 10\underline{0})$

input : 111

$(s, \triangleright \underline{1}11), (q, \triangleright 1\underline{1}1), (q, \triangleright 11\underline{1}), (q, \triangleright 111\underline{\sqcup}), (p, \triangleright 11\underline{1}),$

$(h, \triangleright 11\underline{0})$

input : 110

$(s, \triangleright \underline{1}10), (q, \triangleright 1\underline{1}0), (q, \triangleright 11\underline{0}), (q, \triangleright 110\underline{\sqcup}), (p, \triangleright 11\underline{0}),$

$(p, \triangleright 110\underline{\sqcup}), (h, \triangleright 110\underline{0})$

(ii) According to machine if input starts with 0 it halts. If the first input is 1 then machine reads the next inputs untill it finds $\sqcup$ (empty) symbol, if previous symbol of empty input is 1, it changes it to 0 and halts. Otherwise it will change ~~empty~~ empty string to 0 and halts. In any case where the first item in the tape is empty symbol, machine will go to infinite loop and does not stops.

It computes $f(x) = \begin{cases} 2x & \text{for } x \text{ is even} \\ x-1 & \text{for } x \text{ is odd.} \end{cases}$

13.