Mathematics for Computer Science Summative - Elis Mostyn

December 8, 2019

1 Discrete Mathematics and Linear Algrebra

1

$2^{n+5} \cdot 3^{4n} + 5^{3n+1} = 37m$
Base Case: n =1
$2^6 \cdot 3^4 + 5^4 = 37m$
$64 \cdot 81 + 625 = 37m$
$5184 + 625 = 37m$
$5809 = 37m$
So for n=1, m = 157. Hence n = 1 holds
Assume that n = k holds, where:
$2^{k+5} \cdot 3^{4k} + 5^{3k+1} = 37m$

Let n = k+ 1 = $2^{k+6} \cdot 3^{4k+4} + 5^{3k+4}$
$124 \cdot (2^{k+5} \cdot 3^{4k} + 5^{3k+1}) + 37 \cdot (2^{k+5} \cdot 3^{4k} \cdot 161) = 37c$
$2^{k+5} \cdot 3^{4k} \cdot 161 + 5^{3k+1} \cdot 124 = 37c$
$2^{k+5} \cdot 3^{4k}(2 \cdot 3^4 - 1) + 5^{3k+1}(5^3 - 1) = 37c$
$(2^{k+6} \cdot 3^{4k+4} - 2^{k+5} \cdot 3^{4k}) + (5^{3k+4} - 5^{3k+1}) = 37c$
$(2^{k+6} \cdot 3^{4k+4} + 5^{3k+4}) - (2^{k+5} \cdot 3^{4k} + 5^{3k+1}) = 37c$
$2^{k+6} \cdot 3^{4k+4} + 5^{3k+4} = 37m$
As n=k+1 holds we can say that:
$2^{n+5} \cdot 3^{4n} + 5^{3n+1}$
Is divisible by 37 for all $n \geqslant 1$

2

To visualise the number of bit strings, we will first provide example numbers.
Let $n = 30$ so there are 30 '0s'
Let $k = 4$ so there are 4 '1s'
No restrictions on position of 1s: $\binom{34}{4}$
At least one zero between each 1 (m=1) so 1s look like:
10 10 10 1
Taking each of the above as an indivdual item, there are 27 other '0s' and 4 groups of '1s'. Hence the combinations are: $\binom{31}{4}$
At least two zeroes between each 1 (m=2) so 1s look like:
100 100 100 1
Again, taking each of the above as an individual item, there are 24 other '0s' and once again 4 groups of '1s'. So the combinations are: $\binom{28}{4}$

Using this example we can see a pattern occuring: where the number of combinations of bit strings is:

$$\binom{NumOf0s+NumOf1s-NumBetween(NumOf1s-1)}{NumOf1s}$$

Which in terms of n,k and m is:

$$\binom{n+k-m\cdot(k-1)}{k}$$

3

To identify the probability that neither of the outcomes appears 3 times in a row, we will first identify the probability of that occuring.
Each of the 2^{10} outcomes can be viewed as a bit string.
Then using a program in python we can see where three of the same outcome occurs:

```
def printOptions():
    counter = 0 #Count from 0
    for i in range(0,1024): #For all of the combinations
       num = str(format(i, '#012b')) #Converts the number into a 10 bit binary string
       counter = check3(num,counter,i) #Check the number for 3 1s or 0s
    print(counter)


def check3(num,counter,i):
    for j in range(0,len(num)-2): #So checks every possible place
        if((num[j] == num[j+1]) and (num[j] == num[j+2])): #If the 3 of the same outcome occurs
            counter += 1 #Increment counter
            print(num + "-" + str(i)) #Print the binary value + the decimal value
            return counter
    print("break") #Number is not valid so print "break"
    return counter
```

Starting at 0000000000, we can count the ranges in which it occurs:

Begin	End	Number of Combinations
0000000000	0010010001	146
0010010111	0010011000	2
0010011100	0010100011	8
0010100111	0010101000	2
0010101110	0010110001	4
0010110111	0011001000	18
0011001110	0011010001	4
0011010111	0011011000	2
0011011100	0100100011	72
0100100111	0100101000	2
0100101110	0100110001	4
0100110111	0101001000	18
0101001110	0101010001	4
0101010111	0101011000	2
0101011100	0101100011	8
0101100111	0101101000	2
0101101110	0110010001	36
0110010111	0110011000	2
0110011100	0110100011	8
0110100111	0110101000	2
0110101110	0110110001	4
0110110111	1001001000	146
1001001110	1001010001	4
1001010111	1001011000	2
1001011100	1001100011	8
1001100111	1001101000	2
1001101110	1010010001	36
1010010111	1010011000	2
1010011100	1010100011	8
1010100111	1010101000	2
1010101110	1010110001	4
1010110111	1011001000	18
1011001110	1011010001	4
1011010111	1011011000	2
1011011100	1100100011	72
1100100111	1100101000	2
1100101110	1100110001	4
1100110111	1101001000	18
1101001110	1101010001	4
1101010111	1101011000	2
1101101110	1111111111	146

Total = 846 Combinations where 3 of either outcomes occurs.
P(No outcome 3 times in a row) = 1 - $\frac{846}{1024}$

P(No outcome 3 times in a row) $=\frac{89}{512}$

4

If variable X is equal to the number of the umbrellas left on the bus. Then:
ExpectedValue(X) $=4 \cdot P(all4left) + 3 \cdot P(exactly3left) + 2 \cdot P(exactly2left) + 1 \cdot P(exactly1left) + 0 \cdot P(noneleft)$
P(all 4 left) = $(0.4 \cdot 0.5 \cdot 0.6 \cdot 0.8)$
P(all 4 left) = 0.096
P(Exactly 3 Left):

Combination	Probability
P1,P2,P3,¬ P4	0.024
P1,P2,P4,¬ P0	0.064
P1,P3,P4,¬P2	0.096
P2,P3,P4,¬P1	0.144
Total	0.328

P(Exactly 2 Left):

Combination	Probability
P1,P2,¬P3,¬P4	0.016
P1,¬P2,P3,¬P4	0.024
P1,¬P2,¬P3,P4	0.064
¬P1,P2,P3,¬P4	0.036
¬P1,P2,¬P3,P4	0.096
¬P1,¬P2,P3,P4	0.144
Total	0.364

P(Exactly 1 Left)

Combination	Probability
P1,¬P2,¬P3,¬P4	0.016
¬P1,P2,¬P3,¬P4	0.024
¬P1,¬P2,P3,¬P4	0.036
¬P1,¬P2,¬P3,P4	0.096
Total	0.172

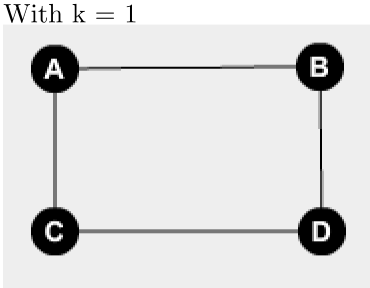
P(None left) = $0.6 \cdot 0.5 \cdot 0.4 \cdot 0.2$
P(None left) = 0.024

Expected Value = $4(0.096) + 3(0.328) + 2(0.364) + 1(0.172) + 0(0.024)$
Expected Value = $0.384 + 0.984 + 0.728 + 0.172 + 0$
Expected Value = 2.268

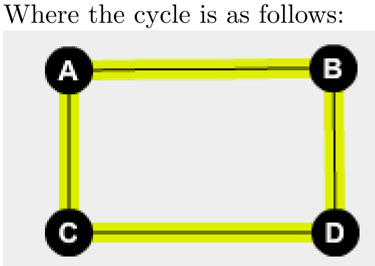
Variance(x) = E(x^2) - (E(x))^2
Variance(x) $=16 \cdot (0.096) + 9 \cdot (0.328) + 4 \cdot (0.364) + 1 \cdot (0.172) + 0 \cdot (0.024) - (2.0268)^2$
Variance(x) = $6.116 - (2.268)^2$
Variance(x) = 0.972

5

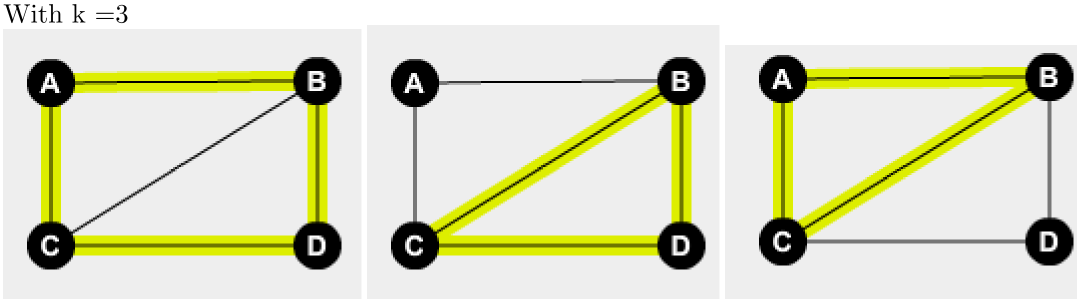
For n=4, there exists a graph with diameter 2:
With k = 1



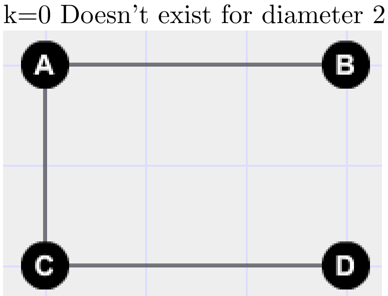
Where the cycle is as follows:



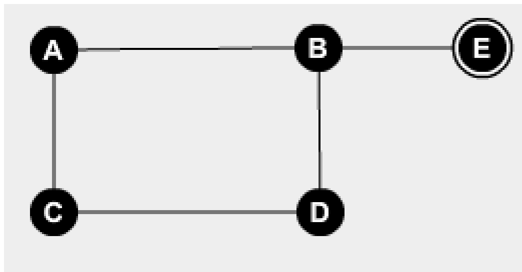
With k =3



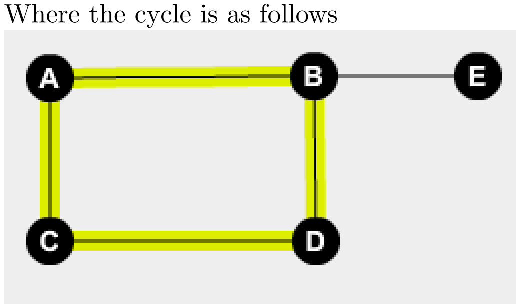
k=0 Doesn't exist for diameter 2 as removing an edge makes the diameter 3.



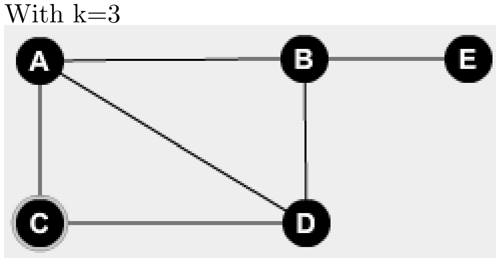
For n = 5, there exists a graph with diameter 3
With k =1



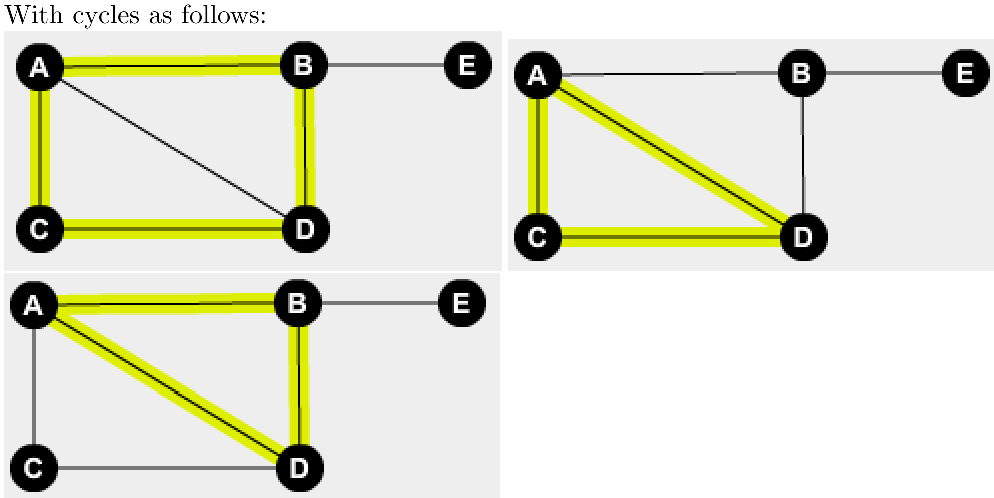
Where the cycle is as follows



With k=3

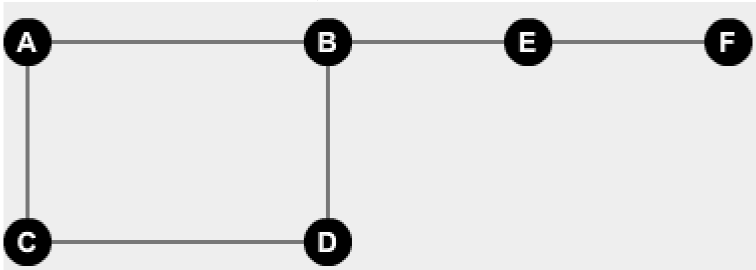


With cycles as follows:

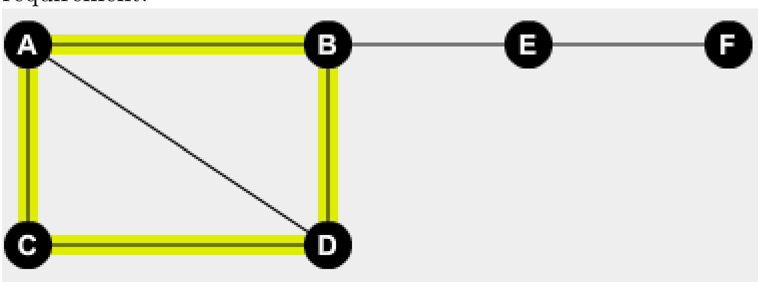


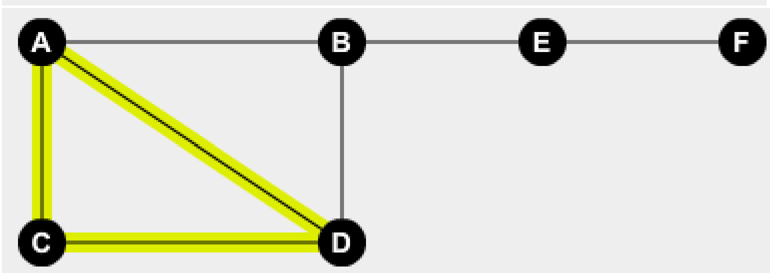
For any n≥4 a graph can be drawn which has a diameter of n-2 and k=1 and k=3 by adding an extra vertex in a chain from "B".

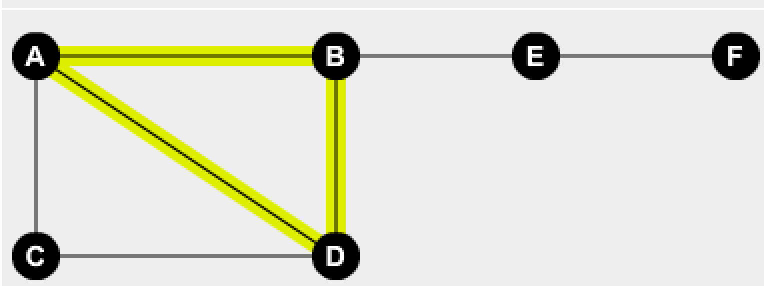
And either having an edge between A & D for k=3 or not for k=1
k=2 cannot exist in a graph because adding another edge to a graph where k=1 holds will add an extra 2 cycles. For example n = 6:

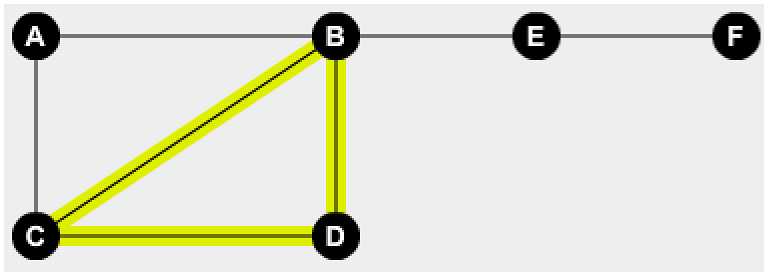
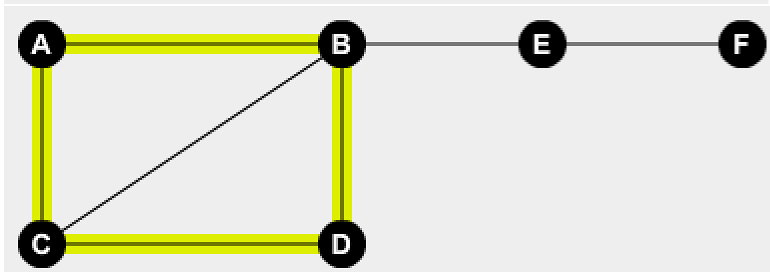
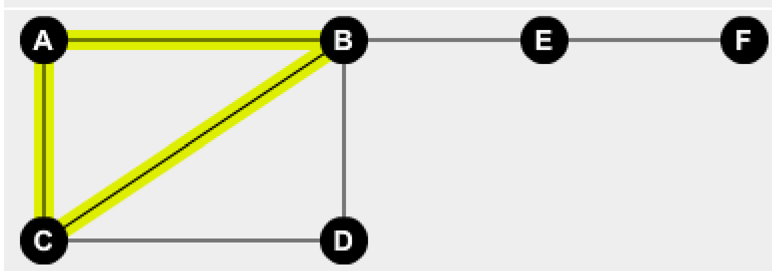
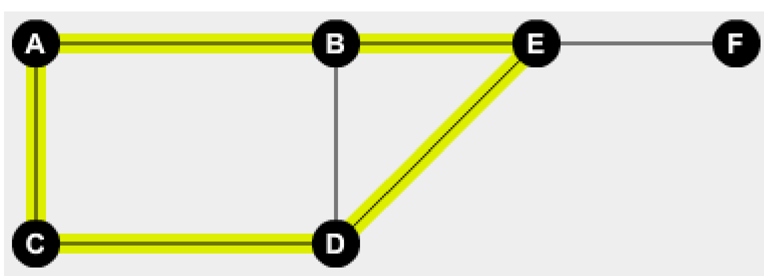
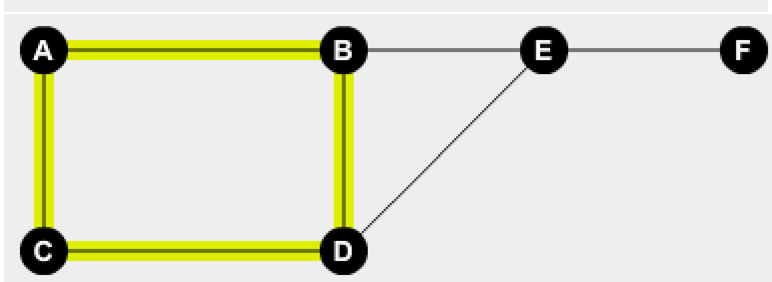


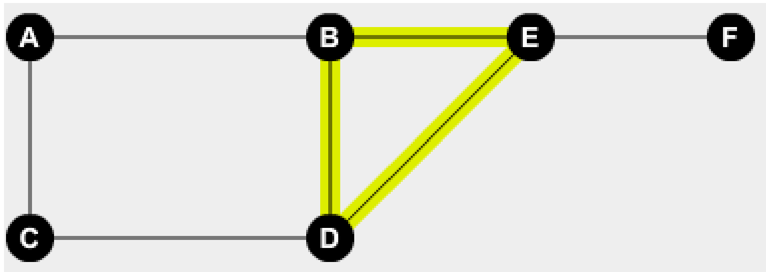
Which holds for k=1
No edge can be added which adds only 1 cycle, even ignoring the n-2 diameter requirement:

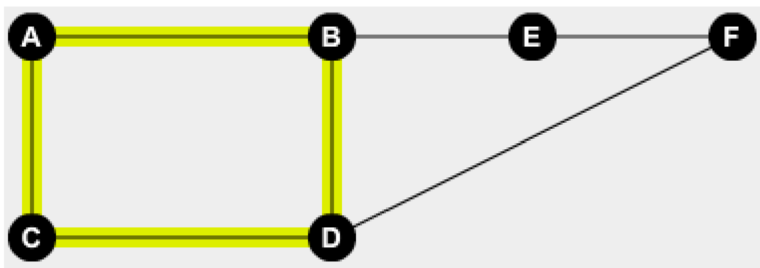


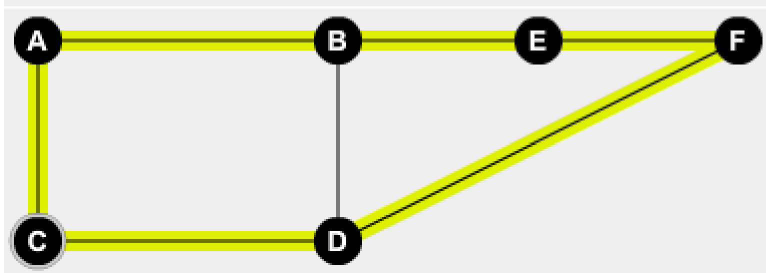


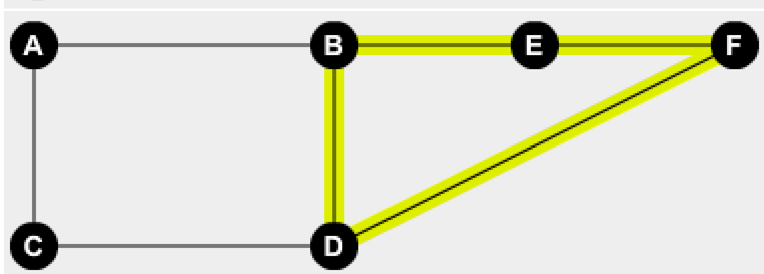



A
B
E
F
C
D

A
B
E
F
C
D

A
B
E
F
C
D

A
B
E
F
C
D

A
B
E
F
C
D









Therefore $k \neq 2$
Furthermore, for n-2 diameter, $1 \geq k \leq 3$ as adding more edges to increase cycles will decrease the diameter.

2 Logic and Discrete Structures

6a

To convert a formula to disjunctive normal form using truth tables,

a	b	c	$\neg a \wedge \neg b$	$(a \wedge b) \implies c$	$\neg(a \wedge b \implies c) \vee \neg a \wedge \neg b$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1
1	1	1	0	1	0

$\varphi = (\neg a \wedge \neg b \wedge \neg c) \vee (\neg a \wedge \neg b \wedge c) \vee (a \wedge b \wedge \neg c)$
Although, due to $\neg c \vee c$
$\varphi = (\neg a \wedge \neg b) \vee (a \wedge b \wedge \neg c)$

6b.

$\neg\varphi = \neg((\neg a \wedge \neg b) \vee (a \wedge b \wedge \neg c))$
$\neg\varphi = (a \vee b) \wedge (\neg a \vee \neg b \vee c)$

7

To prove a set functionally complete we must prove it can represent all the logical operators of the set:
$\{\vee, \wedge, \neg\}$
As that is a set we know to be functionally complete:
If $P \bowtie Q \equiv (P \iff Q) \vee P$
Then the truth table is as follows

P	Q	$P \iff Q$	$(P \iff Q) \wedge P$
0	0	1	0
0	1	0	0
1	0	0	0
1	1	1	1

From this truth table we can see that $P \bowtie Q \equiv P \wedge Q$
As a result we can say that the logical operator $\wedge$ can be produced from the set $\{\vee, \bowtie\}$
Providing truth premises to the set produces this:

P	Q	Operator	Output
1	1	$\vee$	1
1	1	$\bowtie$	1

This is significant because it means that the set is truth preserving. Meaning that it does not produce a false output given a true input. As a result of this, the set cannot represent the $\neg$ operator.
Therefore, the set is not functionally complete.

8a

1) $\neg(a \vee b) \wedge c$	Premise	
2) c	$\wedge e(1)$	
3) $\neg(a \vee b)$	$\wedge\, e(1)$	
4)	a	Assumption
5)	$a \vee b$	$\vee\, i$
6)	$\perp$	$\neg e(3,5)$
7)	b	*Assumption*
8)	$a \vee b$	$\vee\, i(7)$
9)	$\perp$	$\neg e(3,8)$
10) $\neg a$	$\neg i(4-6)$	
11) $\neg b$	$\neg i(7-9)$	
12) $\neg b \wedge c$	$\wedge\, i(2,11)$	
13) $\neg a \wedge (\neg b \wedge c)$	$\wedge\, i(10,12)$	

8b

1)$a \wedge b$	*Premise*	
2) $a \implies (\neg c \wedge \neg b)$	*Premise*	
3)a	$\wedge\, e(1)$	
4) $(\neg c \wedge \neg d)$	$\implies e(3),(2)$	
5) $\neg c$	$\wedge\, e(4)$	
6) $\neg d$	$\wedge\, e(4)$	
7)	$c \vee d$	*Assumption*
8)	$\perp$	$\neg e(5),(6),(7)$
9) $\neg(c \vee d)$	$\neg i(7-8)$	

9

To identify whether φ is a theorem, where:
$\varphi = \neg((\neg t \vee \neg r) \wedge (p \vee p) \wedge (\neg p \vee \neg q) \wedge (r \vee p \vee q \vee \neg r) \wedge (t \vee s) \wedge (\neg r \vee \neg s \vee t) \wedge (q \vee r))$
We must negate φ to find $\neg\varphi$ and apply resolution continually until we have inferred the empty clause or reach a point where we cannot infer any new clauses.

$\neg\varphi = ((\neg t \vee \neg r) \wedge (p \vee p) \wedge (\neg p \vee \neg q) \wedge (r \vee p \vee q \vee \neg r) \wedge (t \vee s) \wedge (\neg r \vee \neg s \vee t) \wedge (q \vee r))$

1) $\neg t \vee \neg r$
2) $p \vee p$
3) $\neg p \vee \neg q$
4) $r \vee p \vee q \vee \neg r$
5) $t \vee s$
6)$\neg r \vee \neg s \vee t$

7)$q \vee r$
8) $\neg r \vee \neg s$ $Resolution - 1, 6$
9) $\neg r \vee \neg t$ $Resolution - 5, 8$
10) $t \vee \neg r$ $Resolution - 5, 6$
11) $\neg r \vee \neg r$ $Resolution - 9, 10$
12) $\neg r$
13) $\neg q$ Resolution - 2,3
14) r Resolution - 7,13
15) ϕ 12,14

As $\neg\varphi$ Produces the empty clause, φ must be always true, hence it is a theorem.