# 1.1

Check whether the following formulas are tautologies:

(p ∧ q → r) ∧ (r ∨ p → s) → ((q → ¬s) → (p → ¬q))

Negate the formula

¬ [(p ∧ q → r) ∧ (r ∨ p → s) → ((q → ¬s) → (p → ¬q))]

CNF

Transform A → B as ¬A v B

¬[(¬p v ¬ q v r) ^ ( ¬ r ^ ¬ p v s ) → (( ¬ q v ¬ s ) → (¬p v ¬q))]

¬[(¬p v ¬ q v r) ^ ( ¬ r ^ ¬ p v s ) → ( ¬ ( ¬ q v ¬ s ) v (¬p v ¬q))]

¬ [ ¬ ((¬p v ¬ q v r) ^ ( ¬ r ^ ¬ p v s )) v ( ¬ ( ¬ q v ¬ s ) v (¬p v ¬q)) ]

((¬p v ¬ q v r) ^ ( ¬ r ^ ¬ p v s )) ^ ¬ ( ¬ ( ¬ q v ¬ s ) v (¬p v ¬q))

(¬p v ¬ q v r) ^ ( ¬ r ^ ¬ p v s ) ^ ( ¬ q v ¬ s ) ^ ¬(¬p v ¬q)

(¬p v ¬ q v r) ^ ( (¬ r ^ ¬ p) v s ) ^ ( ¬ q v ¬ s ) ^ (p ^ q)

(¬p v ¬ q v r) ^ ( ¬r v s) ^ ( ¬p v s) ^ ( ¬ q v ¬ s ) ^ p ^ q

After CNF

I have 6 clause

1. ¬p v ¬q v r
2. ¬r v s
3. ¬p v s
4. ¬q v ¬s
5. p
6. q

RESOLUTION

Apply resolution between C and E, gives:

1. s

Apply resolution between F and D, gives:

1. ¬s

I apply resolution between G and H, give me the empty clause, so it is UNSAT

Since it is unsat, the formula is a tautology because it was negated at the beginning

# 1.2

Check whether the following formulas are tautologies:

¬(p ∧ (q ∨ ((¬p ∨ ¬r) ∧ r)) ∧ (q → ¬p))

First I negate the formula

p ∧ (q ∨ ((¬p ∨ ¬r) ∧ r)) ∧ (q → ¬p)

Rewrite A → B as ¬ A v B

(p ∧ (q ∨ ((¬p ∨ ¬r) ∧ r)) ∧ ( ¬q v ¬p)

Apply distributive rules

Since (¬r ^ r) is impossible I can remove it

p ∧ (q ∨ (¬p ^ r)) ∧ ( ¬q v ¬p)

Apply distributive rules

p ∧ (q ∨ ¬p) ^ (q v r) ∧ ( ¬q v ¬p)

1. p
2. q v ¬p
3. q v r
4. ¬q v ¬p

CDCL

(p[1] | C0 | \* )

Propagate

(p[1], q[2] | C0 | \* )

Go to conflict

(p[1], q[2] | C0 | ¬q[4])

Resolution between 2 and 4

Since I go to conflict without decided literal I get the empty clause, it is UNSAT, so the formula is a tautology

# 1.3

Check whether the following formulas are tautologies:

((p → q ∨ r)) → (p ∨ q ∨ r)

Negate the formula

¬ ((p → q ∨ r)) → (p ∨ q ∨ r))

Rewrite A → B as ¬A v B

¬ (( ¬ p v q ∨ r)) → (p ∨ q ∨ r)

Rewrite A → B as ¬A v B

¬ (( ¬ ( ¬ p v q ∨ r)) v (p ∨ q ∨ r))

(¬ p v q ∨ r) ^ ¬ (p ∨ q ∨ r)

(¬ p v q ∨ r) ^ (¬p ^ ¬q ^ ¬r)

I have 4 clauses

1. ¬ p v q ∨ r
2. ¬p
3. ¬q
4. ¬r

RESOLUTION:

Apply resolution between 1 and 3, gives

1. ¬ p ∨ r

Apply resolution between 4 and 5, gives

1. ¬p

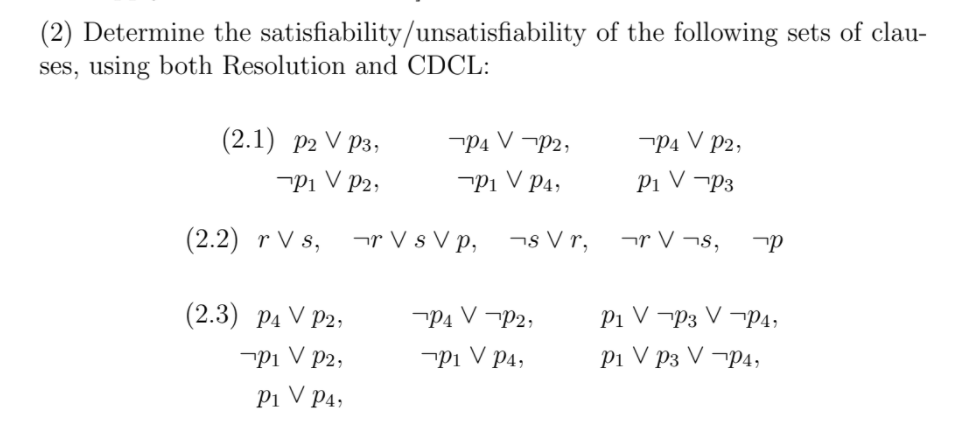
The solution is ¬p, so the formula is SAT and it is not a tautology

CDCL

(¬p[2], ¬q[3], ¬r[4] | C0 | \*)

Propagate

(¬p[2], ¬q[3], ¬r[4], ¬p[1] | C0 | \*)



**2.1**

CDCL

1. p2 v p3
2. ¬p4 v ¬p2
3. ¬p4 v p2
4. ¬p1 v p2
5. ¬p1 v p4
6. p1 v ¬p3

( p2 [d] | C0 | \* )

Propagate

( p2[d], ¬p4[2] | C0 | \* )

Propagate

( p2[d], ¬p4[2], ¬p1[5] | C0 | \* )

Propagate

( p2[d], ¬p4[2], ¬p1[5], ¬p3[6] | C0 | \* )

( p2[d], ¬p4[2], ¬p1[5], ¬p3[6], p2[1] | C0 | \* )

*SATISFIABLE*

RESOLUTION

1. p2 v p3
2. ¬p4 v ¬p2
3. ¬p4 v p2
4. ¬p1 v p2
5. ¬p1 v p4
6. p1 v ¬p3

Apply resolution between 2 and 3

1. ¬p4

Apply resolution between 7 and 5

1. ¬p1

Apply resolution between 8 and 6

1. ¬p3

Apply resolution between 9 and 1

1. p2

Since all literals have been evaluated and I had no conflict, the formula is SAT with value of clause 7,8,9,10

# 2.2

1. r ∨ s
2. ¬r ∨ s ∨ p
3. ¬s ∨ r
4. ¬r ∨ ¬s
5. ¬p

CDCL

( ¬p[5] | C0 | \* )

Decide

( ¬p[5], s[d] | C0 | \* )

Decide

( ¬p[5], s[d], r[3] | C0 | \* )

Decide

( ¬p[5], s[d], r[3] | C0 | \* )

Conflict

( ¬p[5], s[d], r[3] | C0 | ¬r∨¬s[4] )

Resolution between 3 and 4

( ¬p[5], ¬s[6] | C0 | \* )

Decide

( ¬p[5], ¬s[6], r[1] | C0 | \* )

Conflict

( ¬p[5], ¬s[6], r[1] | C0 | ¬r∨s∨p[2] )

Conflict without decided literals, UNSAT

RESOLUTION

1. r ∨ s
2. ¬r ∨ s ∨ p
3. ¬s ∨ r
4. ¬r ∨ ¬s
5. ¬p

Resolution between 2 and 5

1. ¬r ∨ s

Resolution between 1 and 6

1. s

Resolution between 7 and 3

1. r

Resolution between 8 and 4

1. ¬s

Resolution between 7 and 9, give empty clause, so UNSAT

# 2.3

1. p4 ∨ p2
2. ¬p4 ∨ ¬p2
3. p1 ∨ ¬p3 ∨ ¬p4
4. ¬p1 ∨ p2
5. ¬p1 ∨ p4
6. p1 ∨ p3 ∨ ¬p4
7. p1 ∨ p4

CDCL

decide

(p4[d] | C0 | \* )

propagate

(p4[d], ¬p2[2] | C0 | \* )

propagate

(p4[d], ¬p2[2], ¬p1[4] | C0 | \* )

propagate

(p4[d], ¬p2[2], ¬p1[4], p3[6] | C0 | \* )

Conflict

(p4[d], ¬p2[2], ¬p1[4], p3[6] | C0 | [3] )

Resolution

[6] [3]

------------

p1 v ¬p4

Apply backjumping

p1 v ¬p4

(p4[d], ¬p2[2], ¬p1[4] | C0 | p1 v ¬p4)

Conflict

[4] p1 v ¬p4

———

p2 v ¬p4

Conflict

(p4[d], ¬p2[2] | C0 | p2 v ¬p4)

[2] p2 v ¬p4

————

¬p4

(p4[d] | C0 | ¬p4)

Apply back jumping

Learned clause

¬p4 [8]

Search state

( ¬p4[8] | C0 [8] | \*)

Propagate

( ¬p4[8], p2[1] | C0 [8] | \*)

Propagate

( ¬p4[8], p2[1], ¬p1[5] | C0 [8] | \*)

Conflict

( ¬p4[8], p2[1], ¬p1[5] | C0 [8] | [7] )

Since I got to conflict without decided literal it is unsat

RESOLUTION

1. p4 ∨ p2
2. ¬p4 ∨ ¬p2
3. p1 ∨ ¬p3 ∨ ¬p4
4. ¬p1 ∨ p2
5. ¬p1 ∨ p4
6. p1 ∨ p3 ∨ ¬p4
7. p1 ∨ p4

Resolution between 1 and 2

1. UNSAT

# 3.1 FIX!!!!

All suspects are passengers on the Orient Express

Some of the suspects have been questioned.

Some suspects are guilty.

-----------------------------------

Some Orient Express passengers who have been questioned are guilty

∀x(S(x) → P(x))

∃x(S(x) ^ Q(x))

∃x(S(x) ^ G(x))

----------------------

∃xP(x)^Q(x)^G(x)

Negate the thesis

¬(∃x P(x)^Q(x)^G(x))

¬∃x A(x) is equal to ∀x ¬A(x)

∀x(¬P(x) v ¬Q(x) v ¬G(x))

∀x(S(x) → P(x))

∃x(S(x) ^ Q(x))

∃x(S(x) ^ G(x))

∀x (¬P(x) v ¬Q(x) v ¬G(x))

initialise:

S(c) → P(c)

S(c) ^ Q(c) or S(f(c)) ^ Q(c) ????

S(c) ^ G(c)

¬P(c) v ¬Q(c) v ¬G(c)

CNF:

¬S(c) v P(c)

S(c) ^ Q(c)

S(c) ^ G(c)

¬P(c) v ¬Q(c) v ¬G(c)

We have 6 clouse

1. ¬S(c) v P(c)
2. S(c)
3. Q(c)
4. S(c)
5. G(c)
6. ¬P(c) v ¬Q(c) v ¬G(c)

Since clause 2 and clause 4 are the same I can remove one

1. ¬S(c) v P(c)
2. S(c)
3. Q(c)
4. G(c)
5. ¬P(c) v ¬Q(c) v ¬G(c)

CDCL:

(S(c)[2], Q(c)[3], G(c)[4] | Co | \*)

Propagate

(S(c)[2], Q(c)[3], G(c)[4], P(c)[1] | Co | \*)

Conflict

(S(c)[2], Q(c)[3], G(c)[4], P(c)[1] | Co | [5])

Since I go in conflict without decided literal, it is UNSAT, so the formula is correct

# 3.2

All cats are felines.

No felines are vegetarian.

----------------

If Tom is a vegetarian, he is not a cat

Rewrite the sentences

∀x (C(x) → F(x))

∀x ¬( F(x) ^ V(x))

-------------------

V(t) → ¬C(t)

Negate the thesis

∀x (C(x) → F(x))

∀x ¬( F(x) ^ V(x))

¬ (V(t) → ¬C(t))

Skolemise:

C(t) → F(t)

¬(F(t) ^ V(t))

¬ (V(t) → ¬C(t))

Rewrite in CNF

¬C(t) v F(t)

¬F(t) v ¬V(t)

V(t) ^ C(t)

4 clauses

1. ¬C(t) v F(t)
2. ¬F(t) v ¬V(t)
3. V(t)
4. C(t)

RESOLUTION:

Apply resolution between clause 2 and clause 3, gives

1. ¬F(t)

Apply resolution between clause 1 and 4, gives

1. F(t)

Apply resolution between clause 5 and 6, give the empty clause, so UNSAT. Therefore the formula is correct

CDCL:

(V(t)[3], C(t)[4] | C0 | \*)

Propagate

(V(t)[3], C(t)[4], ¬F(t)[2] | C0 | \*)

Conflict

(V(t)[3], C(t)[4], ¬F(t)[2] | C0 | [1])

Since I go to conflict without decided literal, I get UNSAT, so the formula is correct

# 3.3 RIFARE

Maria only talks to those who play with her or play with someone she likes.

Laura only plays with little girls and Maria doesn’t like little girls.

Maria is not a little girl.

----------------------------

Mary doesn’t talk to Laura

∀x∀y (T(m,x) → (P(x,m) v (P(x,y)^L(m,y))))

∀z((P(l,z) → S(z)) ^ (S(z) → ¬L(m,z))) S for short

¬S(m)

------------------------

¬ T(m,l)

Negate the thesis

T(m,x) → (P(x,m) v (P(x,y)^L(m,y)))

(P(l,z) → S(z)) ^ (S(z) → ¬L(m,z)) S for short

¬S(m)

T(m,l)

Rewrite in CNF

¬T(m,x) v (P(x,m) v P(x,y)) ^ (P(x,m) v L(m,y))

becomes

(¬T(m,x) v P(x,m) v P(x,y)) ^ (¬T(m,x) v P(x,m) v L(m,y))

(¬ P(l,z) v S(z)) ^ (¬ S(z) v ¬L(m,z))

¬S(m)

T(m,l)

There are 6 clauses

1. ¬T(m,x) v P(x,m) v P(x,y)
2. ¬T(m,x) v P(x,m) v L(m,y)
3. ¬ P(l,z) v S(z)
4. ¬ S(z) v ¬L(m,z)
5. ¬S(m)
6. T(m,l)

3 variables, x,y,z

2 constant m,l

1. x=m, y=m, z=m
   1. ¬T(m,m) v P(m,m) v P(m,m)
   2. ¬T(m,m) v P(m,m) v L(m,m)
   3. ¬ P(l,m) v S(m)
   4. ¬ S(m) v ¬L(m,m)
   5. ¬S(m)
   6. T(m,l)
2. x=m, y=m, z=l
   1. ¬T(m,m) v P(m,m) v P(m,m)
   2. ¬T(m,m) v P(m,m) v L(m,m)
   3. ¬ P(l,l) v S(l)
   4. ¬ S(l) v ¬L(m,l)
   5. ¬S(m)
   6. T(m,l)
3. x=m, y=l, z=m
4. x=l, y=m, z=m
5. x=m, y=l, z=l
6. x=l, y=l, z=m
7. x=l, y=m, z=l
8. x=l, y=l, z=l
9. x=m, y=m, z=m
   1. ¬T(m,m) v P(m,m)
   2. ¬T(m,m) v P(m,m) v L(m,m)
   3. ¬ P(l,m) v S(m)
   4. ¬ S(m) v ¬L(m,m)
   5. ¬S(m)
   6. T(m,l)

CDCL for 1.

(¬S(m)[e], T(m,l)[f] | C0 | \* )

propagate

(¬S(m)[e], T(m,l)[f], ¬ P(l,m)[c] | C0 | \* )

decide

(¬S(m)[e], T(m,l)[f], ¬ P(l,m)[c], ¬P(m,m)[decided] | C0 | \* )

propagate

(¬S(m)[e], T(m,l)[f], ¬ P(l,m)[c], ¬P(m,m)[decided], ¬T(m,m)[a] | C0 | \* )

decide

(¬S(m)[e], T(m,l)[f], ¬ P(l,m)[c], ¬P(m,m)[decided], ¬T(m,m)[a], L(m,m)[decide] | C0 | \* )

SAT, since is sat, the formula is non a tautology

# 3.4

He who is lucky or knows how to bluff wins at poker and vice versa

To know how to bluff you need to be smart

Donald is not smart

--------------------

Guy wins at poker if and only if he’s lucky

(L(x) v B(x)) → W(x)

W(x) → ( L(x) v B(x))

B(x) → S(x)

¬S(d)

---------------

W(x) → L(x) ^ L(x) → W(x)

Negate the thesis

(L(x) v B(x)) → W(x)

W(x) → ( L(x) v B(x))

B(x) → S(x)

¬S(d)

---------------

¬(W(x) → L(x) ^ L(x) → W(x))

Rewrite in CNF I

(¬L(x) ^ ¬B(x)) v W(x)

¬W(x) v (L(x) v B(x))

¬B(x) v S(x)

¬S(d)

((W(x) ^ ¬L(x)) v (L(x) ^ ¬W(x))

Rewrite in CNF II

(¬L(x) v W(x)) ^ (¬B(x) v W(x))

¬W(x) v L(x) v B(x)

¬B(x) v S(x)

¬S(d)

((W(x) ^ ¬L(x)) v (L(x) ^ ¬W(x))

Rewrite in CNF III

(¬L(x) v W(x)) ^ (¬B(x) v W(x))

¬W(x) v L(x) v B(x)

¬B(x) v S(x)

¬S(d)

((W(x) v L(x) ^ ¬W(x))) ^ (¬L(x) v (L(x) ^ ¬W(x))

Rewrite in CNF IV

¬L(x) v W(x)

¬B(x) v W(x)

¬W(x) v L(x) v B(x)

¬B(x) v S(x)

¬S(d)

W(x) v (L(x) ^ ¬W(x))

¬L(x) v (L(x) ^ ¬W(x))

Rewrite in CNF V

¬L(x) v W(x)

¬B(x) v W(x)

¬W(x) v L(x) v B(x)

¬B(x) v S(x)

¬S(d)

W(x) v L(x)

W(x) v ¬W(x)

¬L(x) v L(x)

¬L(x) v ¬W(x)

Initialise

1. ¬L(d) v W(d)
2. ¬B(d) v W(d)
3. ¬W(d) v L(d) v B(d)
4. ¬B(d) v S(d)
5. ¬S(d)
6. W(d) v L(d)
7. W(d) v ¬W(d)
8. ¬L(d) v L(d)
9. ¬L(d) v ¬W(d)

CDCL

(¬S(d)[5] | C0 | \*)

Propagate

(¬S(d)[5], ¬B(d)[4] | C0 | \*)

Decide

(¬S(d)[5], ¬B(d)[4], ¬W(d)[d] | C0 | \*)

Propagate

(¬S(d)[5], ¬B(d)[4], ¬W(d)[d], ¬L(d)[1] | C0 | \*)

Propagate

(¬S(d)[5], ¬B(d)[4], ¬W(d)[d], ¬L(d)[1] | C0 | \*)

SAT, so the formula is not correct

**4**

Determine the satisfiability/unsatisfiability of the following set of literals

using congruence closure algorithm:

h(b, g(a)) = b

h(b, b) = c

b = g(a)

b ≠ c

g(a) = d

h(b, d) = b

h(b, b) = c

b = d

b ≠ c

Block 1:

g(a), d

Block 2:

h(b, d), b

Block 3:

h(b, b), c

Block 4:

b, d

Since b = d (block 4), merge block 1 and block 2

Block 1:

g(a), h(b, d), b, d

Block 2:

h(b, b), c

Since g(a) = d, merge block 1 and block 3

Block 1:

g(a), d, b, h(b, d)

Block 2:

h(b, b), c

Since b = d (block 1), h(b,b)=h(b,d), merge block 1 and block 2

Block 1:

g(a), d, b, h(b, d), h(b, b), c

Since b ≠ c UNSAT

# 5

Determine the satisfiability/unsatisfiability of the following set of literals using Fourier-Motzkin algorithm:

y − 1 < x,

3x + y < 2y,

0 < x

y<x+1

y>3x

x>0

Remove y

3x<x+1

x>0

Remove x

x<1/2

x>0

0<1/2

SAT

# 6

Determine the satisfiability/unsatisfiability of the following set of literals in integer difference logic (IDL):

x </ y,

y − u ≤ −2,

x − z ≤ −2,

z − y ≤ 3,

u − z /≥ −1

x ≥ y

y − u ≤ −2

x − z ≤ −2

z − y ≤ 3

u - z< -1

y - x ≤ 0

y − u ≤ −2

x − z ≤ −2

z − y ≤ 3

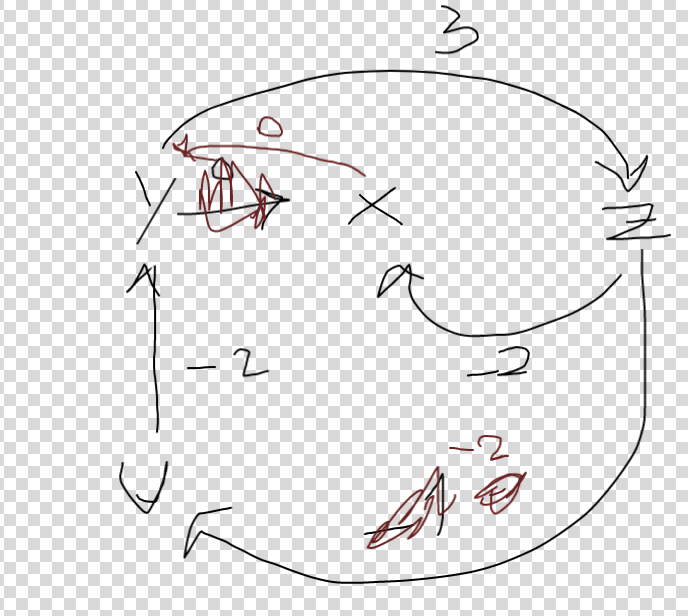
u - z ≤ -2

x --(0)→y

u --(-2)→y

y--(3)→z--(-2)-->x

z--(-2)-->u



-2+(-2)+3 ≥0 UNSAT