Homework 2
CS 474

Alexandra Levinshteyn

October 15th, 2024

Problem 1

We have the following propositional formula:

$$(p \wedge (p \implies q)) \implies q.$$

In order to prove that the above formula is valid, we want to prove that its negation

$$\neg\left((p \wedge (p \implies q)) \implies q\right)$$

is unsatisfiable. We will use the Tseytin transformation to transform the negation into CNF form.

Consider all subformulas and introduce a variable for each one:

$$\begin{cases} x_1 \iff ((p \wedge (p \implies q)) \implies q) \\ x_2 \iff (p \wedge (p \implies q) \\ x_3 \iff (p \implies q) \end{cases}.$$

We can rewrite this as

$$\begin{cases} x_1 \iff (x_2 \implies q) \\ x_2 \iff (p \wedge x_3) \\ x_3 \iff (p \implies q) \end{cases}.$$

The equations that we want are then

$$\begin{cases} \neg x_1 \\ x_1 \implies (x_2 \implies q) \\ (x_2 \implies q) \implies x_1 \\ x_2 \implies (p \wedge x_3) \\ (p \wedge x_3) \implies x_2 \\ x_3 \implies (p \implies q) \\ (p \implies q) \implies x_3 \end{cases}.$$

We can rewrite these as

$$\begin{cases} \neg x_1 \\ \neg x_1 \vee (\neg x_2 \vee q) \\ \neg\left(\neg x_2 \vee q\right) \vee x_1 \\ \neg x_2 \vee (p \wedge x_3) \\ \neg(p \wedge x_3) \vee x_2 \\ \neg x_3 \vee (\neg p \vee q) \\ \neg(\neg p \vee q) \vee x_3 \end{cases}.$$

To get to the correct form, we rewrite these as

$$\begin{cases} \neg x_1 \\ \neg x_1 \vee \neg x_2 \vee q \\ (x_2 \wedge \neg q) \vee x_1 \\ (\neg x_2 \vee p) \wedge (\neg x_2 \vee x_3) \\ \neg p \vee \neg x_3 \vee x_2 \\ \neg x_3 \vee \neg p \vee q \\ (p \wedge \neg q) \vee x_3 \end{cases}.$$

Finally, separating some equations, we get the following set of clauses Γ that is equisatisfiable to the negation of our given formula:

$$\begin{cases} \neg x_1 \\ \neg x_1 \vee \neg x_2 \vee q \\ x_2 \vee x_1 \\ \neg q \vee x_1 \\ \neg x_2 \vee p \\ \neg x_2 \vee x_3 \\ \neg p \vee \neg x_3 \vee x_2 \\ \neg x_3 \vee \neg p \vee q \\ p \vee x_3 \\ \neg q \vee x_3 \end{cases}.$$

This can be written as

$$\begin{cases} \{\neg x_1\} \\ \{\neg x_1, \neg x_2, q\} \\ \{x_2, x_1\} \\ \{\neg q, x_1\} \\ \{\neg x_2, p\} \\ \{\neg x_2, x_3\} \\ \{\neg p, \neg x_3, x_2\} \\ \{\neg x_3, \neg p, q\} \\ \{p, x_3\} \\ \{\neg q, x_3\} \end{cases}.$$

Thus, the negation of our original formula is satisfiable if and only if Γ is satisfiable. Note that the original formula is valid if and only if its negation is not satisfiable. Therefore, the original formula is valid if and only if Γ is not satisfiable.

We then have the following resolution refutation for Γ:

1. $\{\neg x_1\}$ (First clause from above)
2. $\{x_2, x_1\}$ (Third clause from above)
3. $\{x_2\}$ (Resolvent of 1 and 2 withy respect to x_1)
4. $\{\neg q, x_1\}$ (Fourth clause from above)
5. $\{\neg q\}$ (Resolvent of 1 and 4 with respect to x_1)
6. $\{\neg x_2, p\}$ (Fifth clause from above)
7. $\{p\}$ (Resolvent of 3 and 6 with respect to x_2)
8. $\{\neg x_3, \neg p, q\}$ (Eighth clause from above)
9. $\{\neg x_3, q\}$ (Resolvent of 7 and 8 with respect to p)
10. $\{\neg x_3\}$ (Resolvent of 5 and 9 with respect to q)
11. $\{\neg x_2, x_3\}$ (Sixth clause from above)
12. $\{\neg x_2\}$ (Resolvent of 10 and 11 with respect to x_3)
13. $\{\}$ (Resolvent of 3 and 12 with respect to x_2)

Since we end up with the empty clause, Γ is not satisfiable, so the negation of our original formula is not satisfiable. Therefore, the original formula is valid.

The proof/interaction with the resolution tool is shown below. I inputted the formulas I got for the CNF form and then followed my same process until I reached the empty clause.

```
Apply resolution rules by providing clause numbers and the literal on which you want to resolve, e.g.
1:{x, y} 2:{!x, z}
>> 1 2 x
1:{y, z}
Enter 'b' to backtrack, and 'done' to indicate that you have saturated resolution steps.
Enter 'help' to display this message.

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3}
>> 1 3 x1

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2}
>> 1 4 x1

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q}
>> 11 6 x2

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q} 13:{x3}
>> b

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q}
>> 11 5 x2

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q} 13:{p}
>> 13 8 p
>> 6 15 x3

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q} 13:{p} 14:{!x3, q} 15:{!x3} 16:{!x2}
>> 11 16 x2

1:{!x1} 2:{!x1, !x2, q} 3:{x1, x2} 4:{!q, x1} 5:{!x2, p} 6:{!x2, x3} 7:{!p, !x3, x2} 8:{!p, !x3, q} 9:{p, x3} 10:{!q, x3} 11:{x2} 12:{!q} 13:{p} 14:{!x3, q} 15:{!x3} 16:{!x2} 17:{}
>> done
Enter the name of the file where you want to save your proof. If you press Enter, it will be saved to proof.json
```

The link to the resulting json (which has been retitled "CS 474 Homework 2 Problem 1 Proof.json") is here:

https://github.com/AlexandraLevinshteyn/Alexandra-Levinshteyn-CS-474/blob/main/CS%20474%20Homework%202/CS%20474%20Homework%202%20Problem%201%20Proof.json

Problem 2

Part (a)

We have the following formula φ:

$$(q \vee \neg r) \wedge (\neg p \vee r) \wedge (\neg q \vee r \vee p) \wedge (p \vee q \vee \neg q) \wedge (\neg r \vee q).$$

We can consider the following set of clauses Γ:

$$\begin{cases} q \vee \neg r \\ \neg p \vee r \\ \neg q \vee r \vee p \\ p \vee q \vee \neq q \\ \neg r \vee q \end{cases}.$$

We write these as follows and then resolve:

1. $\{q, \neg r\}$
2. $\{\neg p, r\}$
3. $\{\neg q, r, p\}$
4. $\{p, q, \neg q\} \equiv T$ (so we no longer have to worry about it)
5. $\{\neg r, q\} \equiv \{q, \neg r\}$ (so it's equivalent to 1 and we don't have to worry about it)
6. $\{q, \neg p\}$ (Resolution of 1 and 2 with respect to r)
7. $\{\neg r, r, p\} \equiv T$ (Resolution of 1 and 3 with respect to q, so we no longer have to worry about it)
8. $\{q, \neg q, p\} \equiv T$ (Resolution of 1 and 3 with respect to r, so we no longer have to worry about it)
9. $\{\neg q, r, r\} \equiv \{\neg q, r\}$ (Resolution of 2 and 3 with respect to p)
10. Resolving 1 and 6 is impossible
11. $\{\neg r, r\} \equiv T$ (Resolution of 1 and 9 with respect to q, so we no longer have to worry about it)
12. $\{q, \neg q\} \equiv T$ (Resolution of 1 and 9 with respect to r, so we no longer have to worry about it)
13. Resolving 2 and 6 is impossible
14. Resolving 2 and 9 is impossible
15. $\{r, p, \neg p\} \equiv T$ (Resolution of 3 and 6 with respect to q, so we no longer have to worry about it)
16. $\{\neg q, r, q\} \equiv T$ (Resolution of 3 and 6 with respect to p, so we no longer have to worry about it)
17. Resolving 3 and 9 is impossible
18. Resolving 6 and 9 leads to 2

We can resolve no further.

This leads us to the following exhaustive set of clauses:

1. $\{q, \neg r\}$
2. $\{\neg p, r\}$
3. $\{\neg q, r, p\}$
4. $\{q, \neg p\}$
5. $\{\neg q, r\}$

which is the following formula ψ:

$$(q \vee \neg r) \wedge (\neg p \vee r) \wedge (\neg q \vee r \vee p) \wedge (q \vee \neg p) \wedge (\neg q, r).$$

A set of clauses Γ (representing φ) has a resolution refutation if and only if Γ is unsatisfiable. We have just shown that the resulting exhaustive set of clauses from Γ does not contain an empty clause. Since Γ has no resolution refutation, φ is satisfiable.

Part (b)

My Z3 work is shown below.

```
%pip install z3-solver

from z3 import *
import numpy as np
```

```
Requirement already satisfied: z3-solver in /usr/local/lib/python3.10/dist-packages (4.13.3.0)
```

```
[13] def find_solution(formula):
       # create a SAT instance
       s = Solver()
       s.add(formula)
       # Check for satisfiability and return model if possible
       if s.check() == sat:
         return s.model()
       else:
         return []
```

```
[14] # Problem 2 Part (b) Satisfiability
     p = Bool("p")
     q = Bool("q")
     r = Bool("r")
     phi = And(Or(q, Not(r)), Or(Not(p), r), Or(Not(q), r, p), Or(p, q, Not(q)), Or(Not(r), q))
     sol = find_solution(phi)
     print("Potential Solution: " + str(sol))
     print("As there is a potential solution, phi is satisfiable.")
```

```
Potential Solution: [q = False, p = False, r = False]
As there is a potential solution, phi is satisfiable.
```

```
[23] # Problem 2 Part (b) Equality

     psi = And(Or(q, Not(r)), Or(Not(p), r), Or(Not(q), r, p), Or(q, Not(p)), Or(Not(q), r))
     prove(phi == psi)
     print("As the above prove call prints 'proved' and provides no counterxample, phi is logically equivalent to psi.")
```

```
proved
As the above prove call prints 'proved' and provides no counterxample, phi is logically equivalent to psi.
```

The link to the ipynb is here:

https://github.com/AlexandraLevinshteyn/Alexandra-Levinshteyn-CS-474/blob/main/CS%20474%20Homework%202/CS%20474%20Homework%202%20Code.ipynb

Problem 3

As a reminder, these are the clauses we are working with from Problem 1:

$$\begin{cases} \neg x_1 \\ \neg x_1 \vee \neg x_2 \vee q \\ x_2 \vee x_1 \\ \neg q \vee x_1 \\ \neg x_2 \vee p \\ \neg x_2 \vee x_3 \\ \neg p \vee \neg x_3 \vee x_2 \\ \neg x_3 \vee \neg p \vee q \\ p \vee x_3 \\ \neg q \vee x_3 \end{cases}.$$

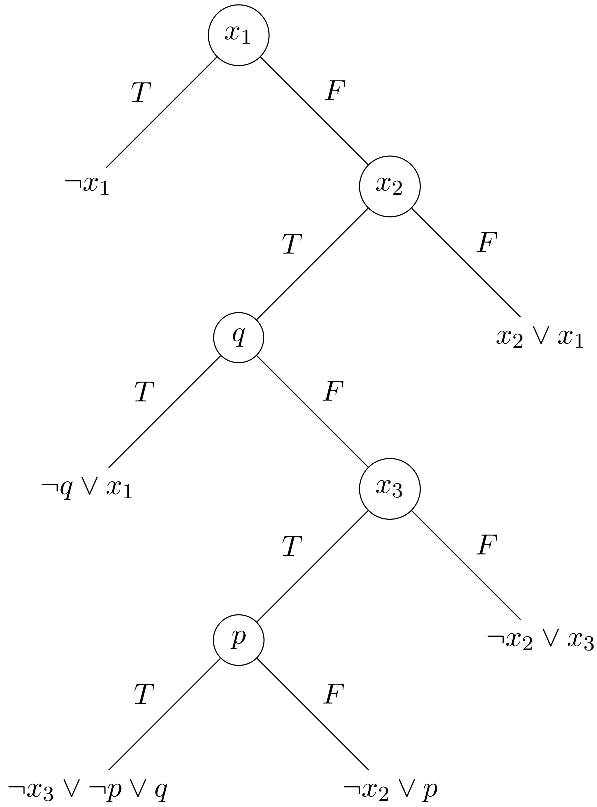
This can be written as

$$\begin{cases} \{\neg x_1\} \\ \{\neg x_1, \neg x_2, q\} \\ \{x_2, x_1\} \\ \{\neg q, x_1\} \\ \{\neg x_2, p\} \\ \{\neg x_2, x_3\} \\ \{\neg p, \neg x_3, x_2\} \\ \{\neg x_3, \neg p, q\} \\ \{p, x_3\} \\ \{\neg q, x_3\} \end{cases}.$$

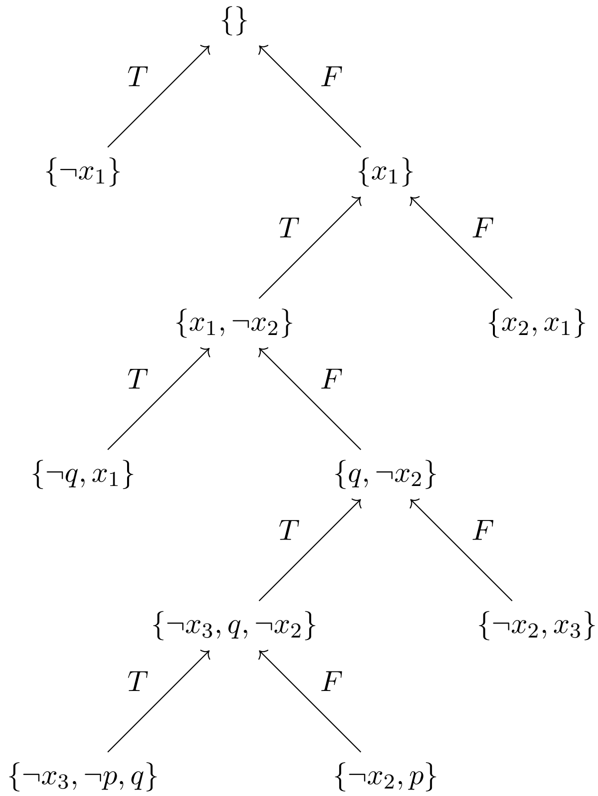
I will fix the variable order $x_1 > x_2 > q > x_3 > p$. This leads to the following falsifications:

1. If x_1 is true, then $\neg x_1$ is false.
2. If x_1 is false and x_2 is false, then $x_2 \vee x_1$ is false.
3. If x_1 is false, x_2 is true, and q is true, then $\neg q \vee x_1$ is false.
4. If x_1 is false, x_2 is true, q is false, and x_3 is false, then $\neg x_2 \vee x_3$ is false.
5. If x_1 is false, x_2 is true, q is false, x_3 is true, and p is true, then $\neg x_3 \vee \neg p \vee q$ is false.
6. If x_1 is false, x_2 is true, q is false, x_3 is true, and p is false, then $\neg x_2 \vee p$ is false.

We get the resulting decision tree:



Following the procedure given in the instructions for converting a decision tree to a proof of resolution gives us the following:



This is equivalent to the following resolution proof:

1. $\{\neg x_3, \neg p, q\}$ (Original clause)
2. $\{\neg x_2, p\}$ (Original clause)
3. $\{\neg x_3, q, \neg x_2\}$ (Resolvent of 1 and 2 with respect to p)
4. $\{\neg x_2, x_3\}$ (Original clause)
5. $\{q, \neg x_2, \neg x_2\} \equiv \{q, \neg x_2\}$ (Resolvent of 3 and 4 with respect to x_3)
6. $\{\neg q, x_1\}$ (Original clause)
7. $\{x_1, \neg x_2\}$ (Resolvent of 5 and 6 with respect to q)
8. $\{x_2, x_1\}$ (Original clause)
9. $\{x_1, x_1\} \equiv \{x_1\}$ (Resolvent of 7 and 8 with respect to x_2)
10. $\{\neg x_1\}$ (Original clause)
11. $\{\}$ (Resolvent of 9 and 10 with respect to x_1)

Since we manage to get the root to be an empty clause, the CNF formula is indeed not satisfiable.