The University of Texas at Dallas
CS 6364
Artificial Intelligence
Fall 2020
Instructor: Dr. Sanda Harabagiu
Grader/ Teaching Assistant: Maxwell Weinzierl

<u>Homework 3: 200 points (30 points extra-credit)</u>
Issued October 12, 2020
Due November 9, 2020 before midnight
Submit only in eLearning

Name _______KAPIL GAUTAM____________________
Student ID _________KXG180032__________________

<u>PROBLEM 1:</u> Inference in Propositional Logic (**55 points**)
a/ A propositional 2-CNF expression is a conjunction of clauses, each containing exactly two literals, e.g.:
(X ˅ Y) ^ (⌐ X ˅ Z) ^ (⌐Y ˅ W) ^ (⌐Z ˅ G) ^ (⌐W ˅ G)

(15 points) Prove using resolution that the above sentence entails G.

<u>Solution 1.a/</u>

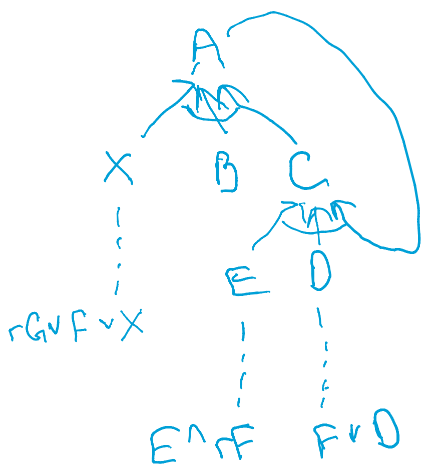
Given sentence S1: (X ˅ Y) ^ (⌐ X ˅ Z) ^ (⌐Y ˅ W) ^ (⌐Z ˅ G) ^ (⌐W ˅ G)
We want to entail α: G i.e., ⌐α: ⌐G
Step1: Using And-Elimination a ^ b gives a or b
S2: X ˅ Y
S3: ⌐ X ˅ Z
S4: ⌐Y ˅ W
S5: ⌐Z ˅ G
S6: ⌐W ˅ G
Step 2: Resolution S2: X ˅ Y and S4: ⌐Y ˅ W
S7: X ˅ W
Step 3: Resolution S7: X ˅ W and S6: ⌐W ˅ G
S8: X ˅ G
Step 4: Resolution S8: G ˅ X and S3: ⌐ X ˅ Z
S9: G ˅ Z
Step 5: Resolution S9: G ˅ Z and S5: ⌐Z ˅ G
S10: G ˅ G , i.e., S10: G
Step 6: Resolution S10: G and ⌐α: ⌐G
NIL
S1 entails G

b/ Given the following Knowledge Base:

1. X ^ B ^ C => A
2. A ^ D ^ E => C
3. B
4. E ^ ⌐ F
5. F ˅ D
6. G ^ ⌐ F => X
7. G

(15 points) Use backward-chaining inference to prove the query A.

<u>Solution 1.b/</u>



S1: X ^ B ^ C => A
S2: A ^ D ^ E => C
S3: B
S4: E ^ ⌐ F
S5: F ˅ D
S6: G ^ ⌐ F => X or S6: ⌐G ˅ F ˅ X
S7: G

Let us perform backward chaining:

S4: E ^ ⌐ F, **S5:** F ˅ D and **S6**: ⌐G ˅ F ˅ X (at-most one literal positive) are not Horn Clauses.

Step1: Suppose A is true
From S1: X ^ B ^ C => A

- X must be true
- B must be true
- C must be true

Step2: From S3: B, we have proof that B is true

Step3: Suppose C is true

From S2: A ^ D ^ E => C

- A is already assumed true
- D must be true
- E must be true

Step4: From S7: G, we have proof that G is true.

Step5: If X true from Step1

From S6: ⌐G ˅ F ˅ X

- Since G is true, ⌐G must be false

S6: ⌐G ˅ F ˅ X is true, regardless of F.

Step5: If D is true from Step3, then S5: F ˅ D true regardless of F.

Step6: If E is true from Step3, then S4 true only if F is False.

E	F	S4: E ^ ⌐ F
False	False	False
False	True	False
True	**False**	**True**
True	True	False

Therefore, the assumption that A is true was correct.

c/ Use propositional logic inference rules to decide which of the following sentences are entailed by the Sentence 1: (X ˅ Y) ^ (⌐ Z ˅ ⌐W ˅ Q):

Sentence 2: (X ˅ Y)

Sentence 3: (X ˅ Y ˅ Z) ^ (Y ^ Z ^ W => Q)

Sentence 4: (X ˅ Y) ^ (⌐W ˅ Q)

To get full credit you need to write if:

i. S1 Entails S2 or not, and if so, which propositional inference rules you have applied to reach the conclusion. Detail the results of each inference rule. **(5 points)**

ii. S1 Entails S3 or not, and if so, which propositional inference rules you have applied to reach the conclusion. Detail the results of each inference rule. **(5 points)**

iii. S1 Entails S4 or not, and if so, which propositional inference rules you have applied to reach the conclusion. Detail the results of each inference rule. **(5 points)**

Solution 1.c/

i. Check S1 entails S2:
S1: $(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$
S2: $(X \vee Y)$, i.e., $\neg$S2: $\neg(X \vee Y)$

$(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$ **|=** $(X \vee Y)$

Step1: Using And-Elimination for S1
C1: $(X \vee Y)$
C2: $(\neg Z \vee \neg W \vee Q)$

Step 2: Resolution for $\neg$S2 and C1
C3: NIL

Thus, S2 is true for all places where S1 is true, i.e., **S1 entails S2**

ii. Check S1 entails S3:
S1: $(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$
S3: $(X \vee Y \vee Z) \wedge (Y \wedge Z \wedge W \Rightarrow Q)$

$(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$ **|=** $(X \vee Y \vee Z) \wedge (Y \wedge Z \wedge W \Rightarrow Q)$

Step1: Eliminate => by replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$
S3: $(X \vee Y \vee Z) \wedge (\neg(Y \wedge Z \wedge W) \vee Q)$

Step2: CNF requires $\neg$ to appear only in literals; we move $\neg$ inwards by repeated application
S3: $(X \vee Y \vee Z) \wedge (\neg Y \vee \neg Z \vee \neg W \vee Q)$

Step3: Using And-elimination on S1: $(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$
C1: $(X \vee Y)$
C2: $(\neg Z \vee \neg W \vee Q)$

Step4: S3: $(C1 \vee Z) \wedge (\neg Y \vee C2)$
Wherever C1 is true regardless of Z, $(C1 \vee Z)$ is true.
Similarly, wherever C2 is true regardless of Y, $(\neg Y \vee C2)$ is true
Therefore, S3 will be true.
Thus, S3 is true for all places where S1 is true, i.e., **S1 entails S3**

iii. Check S1 entails S4:
S1: $(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$
S4: $(X \vee Y) \wedge (\neg W \vee Q)$

$(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$ |= $(X \vee Y) \wedge (\neg W \vee Q)$

Step1: Using And-elimination on S1: $(X \vee Y) \wedge (\neg Z \vee \neg W \vee Q)$
C1: $(X \vee Y)$
C2: $(\neg Z \vee \neg W \vee Q)$

Step2: S4: $C1 \wedge (\neg W \vee Q)$
Step3: From C2: $(\neg Z \vee \neg W \vee Q)$, if $(\neg W \vee Q)$ is true regardless of Z, C2 is true
Thus, S4 is true for all places where S1 is true, i.e., **S1 entails S4**

d/ Demonstrate whether the following sentences are valid, satisfiable or neither. Motivate and detail your demonstrations.

Sentence 1: ((Smart ᵛ Beautiful) => (Interesting ᵛ Boring)) ⇔ ((Smart => Interesting) ᵛ (Beautiful => boring)) **(5 points)**
Sentence 2: (Tall ᵛ Gorgeous) ᵛ ⌐ (Tall => Gorgeous) **(5 points)**

Solution 1.d/

Sentence 1: ((Smart ᵛ Beautiful) => (Interesting ᵛ Boring)) ⇔ ((Smart => Interesting) ᵛ (Beautiful => boring))

A: ((Smart ᵛ Beautiful) => (Interesting ᵛ Boring))
Step1: Eliminate => by replacing α => β with ⌐α ᵛ β
A: (⌐(Smart ᵛ Beautiful) ᵛ (Interesting ᵛ Boring))
Step2: CNF requires ⌐ to appear only in literals; we move ⌐ inwards by repeated application
A: ((⌐Smart ^ ⌐Beautiful) ᵛ (Interesting ᵛ Boring))
Step3: Apply the distributivity law (αᵛ(β^γ)) Ξ ((α ᵛ β) ^ (α ᵛ γ))
A: ((⌐Smart ᵛ Interesting ᵛ Boring) ^ (⌐Beautiful ᵛ Interesting ᵛ Boring))

B: ((Smart => Interesting) ᵛ (Beautiful => Boring))
Step1: Eliminate => by replacing α => β with ⌐α ᵛ β
B: ((⌐Smart ᵛ Interesting) ᵛ (⌐Beautiful ᵛ Boring))
B: (⌐Smart ᵛ Interesting ᵛ ⌐Beautiful ᵛ Boring)

Sentence 1: A ⇔ B
Step1: Eliminate **⇔** by replacing α ⇔ β with (α => β) ^ (β => α)
Sentence 1: (A => B) ^ (B=>A)
Step2: Eliminate => by replacing α => β with ⌐α ᵛ β
Sentence 1: (⌐A ᵛ B) ^ (⌐B ᵛ A)

A: ((⌐Smart ᵛ Interesting ᵛ Boring) ^ (⌐Beautiful ᵛ Interesting ᵛ Boring))
B: (⌐Smart ᵛ Interesting ᵛ ⌐Beautiful ᵛ Boring)
Sentence 1: (⌐A ᵛ B) ^ (⌐B ᵛ A)

Smart	Beautiful	Boring	Interesting	⌐Smart ᵛ ⌐Beautiful	Interesting ᵛ Boring	A	B	Sentence 1
F	F	F	F	T	F	T	T	T
F	F	F	T	T	T	T	T	T
F	F	T	F	T	T	T	T	T
F	F	T	T	T	T	T	T	T
F	T	F	F	T	F	F	T	F
F	T	F	T	T	T	T	T	T
F	T	T	F	T	T	T	T	T

F	T	T	T	T	T	T	T	T
T	F	F	F	T	F	T	T	T
T	F	F	T	T	T	T	T	T
T	F	T	F	T	F	T	T	T
T	F	T	T	T	T	T	T	T
T	T	F	F	F	F	F	F	T
T	T	F	T	F	T	T	T	T
T	T	T	F	F	T	T	T	T
T	T	T	T	F	T	T	T	T

The sentence 1 is satisfiable, since most of the values of the truth-table are true.

<u>Sentence 2:</u> (Tall ᵛ Gorgeous) ᵛ ⌐ (Tall => Gorgeous)

A: Tall ᵛ Gorgeous

B: ⌐(Tall => Gorgeous)

Step1: Eliminate => by replacing α => β with ⌐α ᵛ β

B: ⌐(⌐Tall ᵛ Gorgeous)

Step2: CNF requires ⌐ to appear only in literals; we move ⌐ inwards by repeated application

B: ⌐⌐Tall ^ ⌐Gorgeous

Step3: Double negation removal

B: Tall ^ ⌐Gorgeous

Sentence2: A ᵛ B

Tall	Gorgeous	A: Tall ᵛ Gorgeous	B: Tall ^ ⌐Gorgeous	Sentence2: A ᵛ B
F	F	F	F	F
F	T	T	F	T
T	F	T	T	T
T	T	T	F	T

The sentence is satisfiable, since most of the values of the truth-table are true.

<u>PROBLEM</u> <u>2:</u> (**25 points**) Logic Representations

a/ According to political pundits, a person who is a radical (R) is electable (E) if he/she is conservative (C), but otherwise is not electable. Which of the following are correct representations in propositional logic of this assertion?

i. (R ^ E) ⇔C

ii. R => (E ⇔ C)

iii. R => ((C => E) ᵛ ⌐ E)

Explain why. **(15 points)**

Solution 2.a/

Propositional logic of assertion:

Q: (R ^ C => E) ^ (R ^ ⌐C => ⌐E)

Step1: Eliminate => by replacing α => β with ⌐α ˅ β

Q: (⌐(R ^ C) ˅ E) ^ (⌐(R ^ ⌐C) ˅ ⌐E)

Step2: CNF requires ⌐ to appear only in literals; we move ⌐ inwards by repeated application

Q: (⌐R ˅ ⌐C ˅ E) ^ (⌐R ˅ ⌐⌐C ˅ ⌐E)

Step3: Removing double negation

Q: (⌐R ˅ ⌐C ˅ E) ^ (⌐R ˅ C ˅ ⌐E)

i. S1: (R ^ E) ⇔C

Step1: Eliminate ⇔ by replacing α ⇔ β with (α => β) ^ (β => α)

S1: ((R ^ E) => C) ^ (C => (R ^ E))

Step2: Eliminate => by replacing α => β with ⌐α ˅ β

S1: (⌐ (R ^ E) ˅ C) ^ (⌐C ˅ (R ^ E))

Step3: CNF requires ⌐ to appear only in literals; we move ⌐ inwards by repeated application

S1: (⌐R ˅ ⌐E ˅ C) ^ (⌐C ˅ (R ^ E))

Step4: Apply the distributivity law (α˅(β^γ)) Ξ ((α ˅ β) ^ (α ˅ γ))

S1: (⌐R ˅ ⌐E ˅ C) ^ (⌐C ˅ R) ^ (⌐C ˅ E)

Step5: Rearrange terms

S1: (⌐R ˅ C ˅ ⌐E) ^ (⌐C ˅ R) ^ (⌐C ˅ E)

Not the same representation of Q: (⌐R ˅ ⌐C ˅ E) ^ (⌐R ˅ C ˅ ⌐E)

ii. S2: R => (E ⇔ C)

Step1: Eliminate ⇔ by replacing α ⇔ β with (α => β) ^ (β => α)

S2: R => ((E => C) ^ (C => E))

Step2: Eliminate => by replacing α => β with ⌐α ˅ β

S2: R => ((⌐E ˅ C) ^ (⌐C ˅ E))

Step3: Eliminate => by replacing α => β with ⌐α ˅ β

S2: ⌐R ˅ ((⌐E ˅ C) ^ (⌐C ˅ E))

Step4: Apply the distributivity law (α˅(β^γ)) Ξ ((α ˅ β) ^ (α ˅ γ))

S2: (⌐R ˅ ⌐E ˅ C) ^ (⌐R ˅ ⌐C ˅ E)

Step5: Rearrange terms

S2: (⌐R ˅ ⌐C ˅ E) ^ (⌐R ˅ C ˅ ⌐E)

Same representation of Q: (⌐R ˅ ⌐C ˅ E) ^ (⌐R ˅ C ˅ ⌐E)

iii. S3: R => ((C => E) ˅ ⌐ E)

Step1: Eliminate => by replacing α => β with ⌐α ˅ β

S3: R => ((⌐C ˅ E) ˅ ⌐ E)

Step2: Eliminate => by replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$

S3: $\neg R \vee (\neg C \vee E \vee \neg E)$

Step3: $E \vee \neg E \equiv$ True

S3: $\neg R \vee (\neg C \vee$ True)

S3: $\neg R \vee \neg C$

Not the same representation of Q: $(\neg R \vee \neg C \vee E) \wedge (\neg R \vee C \vee \neg E)$

b/ Unification: For each pair of literals, find the Most General Unifier and the Most General Common Substitution Instance:

(**2 points**) {P(x), P(A)}

Most General Unifier θ: {x/A}

Most General Common Substitution Instance P(A)

(**4 points**) {P[f(x), y, g(y)], P[f(x), z, g(x)]}

θ_1: {y/z} and θ_2: {y/x}

Occur-check fails, need to Standardize Apart

{P[f(x), y, g(w)], P[f(x), z, g(x)]}

Most General Unifier θ: {y/z; w/x}

Most General Common Substitution Instance P[f(x), z, g(x)]

(**4 points**) {P[f(x, g(A,y)), g(A,y)], P[f(x,z),z]}

Most General Unifier θ: {z/g(A,y)}

Most General Common Substitution Instance P[f(x, g(A,y)), g(A,y)]

<u>PROBLEM 3:</u> First-Order Logic (FOL) representations (**40 points**)

Write in FOL the following statements by defining first your vocabulary (i.e. predicates, constants, variables, functions, etc):

Solution 3:

Vocabulary used:

Fall(x), Winter(x), Spring(x), Leaf(x), LoseLeaves(x,y), Color(x,y), Tree(x), Flower(x), FadePoint(x,y), Nice(x), Smells(x,y), LeftInVase(x), Animal(x), BuyFlower(x, y), Kill(x,y), Poet(x), WritePoem(x,y), Sensitive(x), Insured(x), Agent(x), Policy(x), Sells(x,y,z), Born(x,y), Barber(x), InHouse(x), Shave(x,y), Parent(x,y), Citizen(x,y,z), Man(x), Person(x)

Constants used:

Red, Lovely, John, Nice, US

1. (**2 points**) *Some leaves turn red each Fall*.
   ∃x ∀y Leaf(x) ^ Fall(y) => Color(x,Red)

2. (**3 points**) *Some trees lose all their leaves when winter comes.*
   $\exists x\ \forall y\ \forall z\ Tree(x) \wedge Leaf(y) \wedge Winter(z) \Rightarrow LoseLeaves(x,y)$

3. (**2 points**) *Flowers are always nice and they smell lovely*.
   $\forall x\ Flower(x) \Rightarrow Nice(x) \wedge Smell(x, Lovely)$

4. (**2 points**) *One flower does not bring Spring.*
   $\exists x\ \exists y\ Flower(x) \wedge \neg Spring(y)$

5. (**2 points**) *Every flower fades at some point.*
   $\forall x\ Flower(x) \Rightarrow \exists y\ FadePoint(x,y)$

6. (**2 points**) *Only one flower is left in the vase.*
   $\exists x\ Flower(x) \wedge LeftInVase(x) \wedge [\forall y\ Flower(y) \wedge LeftInVase(y) \Rightarrow (x=y)]$

7. (**2 points**) *Every person that buys flowers is sensitive.*
   $\forall x\ \forall y\ Person(x) \wedge BuyFlower(x, y) \Rightarrow Sensitive(x)$

8. (**3 points**) *Poets are sensitive but they do not buy flowers, they write beautiful poems.*
   $\forall x\ Poet(x) \Rightarrow Sensitive(x) \wedge [\exists y\ \neg BuyFlower(x,y)] \wedge [\exists z\ WritePoem(x,z)]$

9. (**2 points**) *No poet will kill an animal.*
   $\forall x\ \forall y\ Poet(x) \wedge Animal(y) \Rightarrow \neg Kill(x, y)$

10. (**3 points**) *There is an agent who sells policies only to people who are not insured.*
    $\exists x\ Agent(x) \wedge [\forall y\ \forall z\ Policy(y) \wedge Sell(x, y, z) \Rightarrow Person(z) \wedge \neg Insured(z)]$

11. (**2 points**) *There is a barber who shaves all men in town who do not shave themselves.*
    $\exists x\ \forall y\ Barber(x) \wedge Man(y) \wedge \neg Shave(y, y) \Rightarrow Shave(x, y)$

12. (**10 points**) *A person born outside the US, one of who has at least one parent who is a US citizen by birth is a US citizen by descent.*
    $\forall x\ Person(x) \wedge \neg Born(x, US) \wedge [\exists y\ Parent(y,x) \wedge Citizen(y, US, Birth)] \Rightarrow Citizen(x,US,Descent)$

13. (**2 points**) *There is a flower that smells nice in the house.*
    $\exists x\ Flower(x) \wedge Smells(x, Nice) \wedge InHouse(x)$

14. (**3 points**) *John bought only two flowers.*
    $\exists x\ \exists y\ BuyFlower(John, x) \wedge BuyFlower(John, y) \wedge \neg(x = y) \wedge [\forall z\ BuyFlower(John, z) \Rightarrow (x=z) \vee (y=z)]$

PROBLEM 4: Refutation in First-Order Logic (**80 points**)

The purpose of this assignment is to give you experience in proving facts with the resolution method and in exposing you to Prover9, an automatic theorem prover that can help you devise your refutations.
Consider the following helpful pointers for using Prover9:

Installation (https://www.cs.unm.edu/~mccune/mace4/gui/v05.html)
For linux users, install python-wxtools also.

Help Manual (https://www.cs.unm.edu/~mccune/prover9/manual/2009-02A/)

Simple tutorial (www.cs.utsa.edu/~bylander/cs5233/prover9-intro.pdf)

You are asked to solve the following puzzle.
1. Anyone who rides a Harley is a rough character.
2. Every biker rides [something that is] either a Harley or a BMW.
3. Anyone who rides any BMW is a yuppie.
4. Every yuppie is a lawyer.
5. Any nice girl does not date anyone who is a rough character.
6. Mary is a nice girl, and John is a biker.
7. (Conclusion) If John is not a lawyer, then Mary does not date John.

Solution 4:
i. (**14 points)** Represent these clauses in first order logic, using only these predicates:

Harley(x) , *Rides(x,y)* , *Rough(x)* , *Biker(x)* , *BMW(x)* , *Yuppie(x)* , *Lawyer(x)* , *Nice(x)* , *Date(x,y)* .

1. Anyone who rides a Harley is a rough character.
   S1: ∀x ∀y [Harley(y) ^ Rides(x,y) => Rough(x)]

2. Every biker rides [something that is] either a Harley or a BMW.
   S2: ∀x [Biker(x) => [∃y Rides(x,y) ^ (Harley (y) ˅ BMW(y))]]

3. Anyone who rides any BMW is a yuppie.
   S3: ∀x ∀y [BMW(y) * Rides(x,y) => Yuppie(x)]

4. Every yuppie is a lawyer.
   S4: ∀x [Yuppie(x) => *Lawyer*(x)]

5. Any nice girl does not date anyone who is a rough character.
   S5: ∀x ∀y [Rough(x) ^ Nice(y) => ⌐*Date*(y,x)]

6. Mary is a nice girl, and John is a biker.
   S6: Nice(Mary) ^ Biker(John)

7. (Conclusion) If John is not a lawyer, then Mary does not date John.
   Q: ⌐Lawyer(John) => ⌐Date(Mary, John)

ii. (**14 points)** Convert the logic sentences to clause form, skolemizing as necessary.

S1: ∀x ∀y [Harley(y) ^ Rides(x,y) => Rough(x)]
Step1: Eliminate => by replacing α => β with ⌐α ˅ β
S1: ∀x ∀y [⌐(Harley(y) ^ Rides(x,y)) ˅ Rough(x)]
Step2: Move ⌐ inwards by repeated application
S1: ∀x ∀y [⌐Harley(y) ˅ ⌐Rides(x,y)) ˅ Rough(x)]
Step3: Eliminate ∀ quantifiers
S1: ⌐Harley(y) ˅ ⌐Rides(x,y)) ˅ Rough(x)

S2: ∀x [Biker(x) => [∃y Rides(x,y) ^ (Harley (y) ˅ BMW(y))]]
Step1: Eliminate => by replacing α => β with ⌐α ˅ β
S2: ∀x [⌐Biker(x) ˅ [∃y Rides(x,y) ^ (Harley (y) ˅ BMW(y))]]
Step2: Eliminate ∃ quantifiers, y is in scope of x variable
Skolemize function: y = f(x)
Step3: Skolemize the sentence
S2: ∀x [⌐Biker(x) ˅ Rides(x,f(x)) ^ (Harley (f(x)) ˅ BMW(f(x)))]
Step4: Eliminate ∀ quantifiers
S2: ⌐Biker(x) ˅ (Rides(x,f(x)) ^ (Harley (f(x)) ˅ BMW(f(x))))
Step5: Apply the distributivity law (α ˅ (β ^ γ)) Ξ ((α ˅ β) ^ (α ˅ γ))
S2: (⌐Biker(x) ˅ Rides(x,f(x))) ^ (⌐Biker(x) ˅ Harley (f(x)) ˅ BMW(f(x)))

Clauses from S2:
X1: ⌐Biker(x) ˅ Rides(x,f(x))
X2: ⌐Biker(x) ˅ Harley (f(x)) ˅ BMW(f(x))

S3: ∀x ∀y [BMW(y) ^ Rides(x,y) => Yuppie(x)]
Step1: Eliminate => by replacing α => β with ⌐α ˅ β
S3: ∀x ∀y [⌐(BMW(y) ^ Rides(x,y)) ˅ Yuppie(x)]
Step2: Move ⌐ inwards by repeated application
S3: ∀x ∀y [⌐BMW(y) ˅ ⌐Rides(x,y) ˅ Yuppie(x)]
Step3: Eliminate ∀ quantifiers
S3: ⌐BMW(y) ˅ ⌐Rides(x,y) ˅ Yuppie(x)

S4: ∀x [Yuppie(x) => *Lawyer*(x)]

Step1: Eliminate => by replacing α => β with ⌐α ˅ β

S4: ∀x [⌐Yuppie(x) ˅ *Lawyer*(x)]

Step2: Eliminate ∀ quantifiers

S4: ⌐Yuppie(x) ˅ *Lawyer*(x)

S5: ∀x ∀y [Rough(x) ^ Nice(y) => ⌐*Date*(y,x)]

Step1: Eliminate => by replacing α => β with ⌐α ˅ β

S5: ∀x ∀y [⌐(Rough(x) ^ Nice(y)) ˅ ⌐*Date*(y,x)]

Step2: Move ⌐ inwards by repeated application

S5: ∀x ∀y [⌐Rough(x) ˅ ⌐Nice(y) ˅ ⌐*Date*(y,x)]

Step3: Eliminate ∀ quantifiers

S5: ⌐Rough(x) ˅ ⌐Nice(y) ˅ ⌐*Date*(y,x)

S6: Nice(Mary) ^ Biker(John)

Step1: Apply and-eminination

X3: Nice(Mary)

X4: Biker(John)

Q: ⌐Lawyer(John) => ⌐Date(Mary, John)

Step1: Apply Contraposition

Q: Date(Mary, John) => Lawyer(John)

Step2: Eliminate => by replacing α => β with ⌐α ˅ β

Q: ⌐Date(Mary, John) ˅ Lawyer(John)

Goal Clause: ⌐Q: Date(Mary, John) ^ ⌐Lawyer(John)

Q1: Date(Mary, John)

Q2: ⌐Lawyer(John)

Thus, the clauses are:

S1: ⌐Harley(y) ˅ ⌐Rides(x,y)) ˅ Rough(x)

X1: ⌐Biker(x) ˅ Rides(x,f(x))

X2: ⌐Biker(x) ˅ Harley (f(x)) ˅ BMW(f(x))

S3: ⌐BMW(y) ˅ ⌐Rides(x,y) ˅ Yuppie(x)

S4: ⌐Yuppie(x) ˅ *Lawyer*(x)

S5: ⌐Rough(x) ˅ ⌐Nice(y) ˅ ⌐*Date*(y,x)

X3: Nice(Mary)

X4: Biker(John)

Q1: Date(Mary, John)

Q2: ⌐Lawyer(John)

iii. (**42 points)** Prove <u>by hand</u> whether the conclusion is true by using resolution refutation (i.e. negate the conclusion and show its unsatisfiability with the rest of the knowledge base). Make sure to document the substitutions you use.

S1: ⌐Harley(y) ᵛ ⌐Rides(x,y)) ᵛ Rough(x)
X1: ⌐Biker(x) ᵛ Rides(x,f(x))
X2: ⌐Biker(x) ᵛ Harley (f(x)) ᵛ BMW(f(x))
S3: ⌐BMW(y) ᵛ ⌐Rides(x,y) ᵛ Yuppie(x)
S4: ⌐Yuppie(x) ᵛ *Lawyer*(x)
S5: ⌐Rough(x) ᵛ ⌐Nice(y) ᵛ ⌐*Date*(y,x)
X3: Nice(Mary)
X4: Biker(John)

Q1: Date(Mary, John)
Q2: ⌐Lawyer(John)

Step1: Resolve **X2**: ⌐Biker(x) ᵛ Harley (f(x)) ᵛ BMW(f(x)) and **S1**: ⌐Harley(y) ᵛ ⌐Rides(x,y)) ᵛ Rough(x) for eliminating Harley(f(x)), θ: {y/f(x)}
X5: ⌐Biker(x) ᵛ BMW(f(x)) ᵛ ⌐Rides(x,f(x)) ᵛ Rough(x)

Step2: Resolve **X4**: Biker(John) and **X5**: ⌐Biker(x) ᵛ BMW(f(x)) ᵛ ⌐Rides(x,f(x)) ᵛ Rough(x) for eliminating Biker(John), θ:{x/John}
X6: BMW(f(John)) ᵛ ⌐Rides(John,f(John)) ᵛ Rough(John)

Step3: Resolve **X6:** BMW(f(John)) ᵛ ⌐Rides(John,f(John)) ᵛ Rough(John) and S3: ⌐BMW(y) ᵛ ⌐Rides(x,y) ᵛ Yuppie(x) for eliminating BMW(f(John)), θ: {y/f(John)}
X7: ⌐Rides(John,f(John)) ᵛ Rough(John) ᵛ ⌐Rides(x, f(John)) ᵛ Yuppie(x)

Step4: Resolve **X7:** ⌐Rides(John,f(John)) ᵛ Rough(John) ᵛ ⌐Rides(x, f(John)) ᵛ Yuppie(x) and **S4**: ⌐Yuppie(x) ᵛ *Lawyer*(x) for eliminating Yuppie(x)
X8: ⌐Rides(John,f(John)) ᵛ Rough(John) ᵛ ⌐Rides(x, f(John)) ᵛ *Lawyer*(x)

Step5: Resolve **X8** ⌐Rides(John,f(John)) ᵛ Rough(John) ᵛ ⌐Rides(x, f(John)) ᵛ *Lawyer*(x) and **S5**: ⌐Rough(x) ᵛ ⌐Nice(y) ᵛ ⌐*Date*(y,x) for Rough(John), θ: {x/John}
X9: ⌐Rides(John,f(John)) ᵛ *Lawyer*(John) ᵛ ⌐Nice(y) ᵛ ⌐*Date*(y,John)

Step6: Resolve **X9:** ⌐Rides(John,f(John)) ᵛ *Lawyer*(John) ᵛ ⌐Nice(y) ᵛ ⌐*Date*(y,John) and **X3**: Nice(Mary) for eliminating Nice(Mary), θ: {y/Mary}
X10: ⌐Rides(John,f(John)) ᵛ *Lawyer*(John) ᵛ ⌐*Date*(Mary,John)

Step6: Resolve **X10**: ⌐Rides(John,f(John)) ᵛ *Lawyer*(John) ᵛ ⌐*Date*(Mary,John) and **Q2:** ⌐Lawyer(John) for eliminating Lawyer(John)
X11: ⌐Rides(John,f(John)) ᵛ ⌐*Date*(Mary,John)

Step7: Resolve **X11**: ⌐Rides(John,f(John)) ˅ ⌐*Date*(Mary,John) and **Q1**: Date(Mary, John) for eliminating Date(Mary, John)
X12: ⌐Rides(John,f(John))

Step8: Resolve **X1**: ⌐Biker(x) ˅ Rides(x,f(x)) and **X4**: Biker(John) for eliminating Biker(John), θ: {x/John}
X13: Rides(John,f(John))

Step9: Resolve **X12:** ⌐Rides(John,f(John)) and **X13**: Rides(John,f(John))
X14: NIL
The query is proved.

iv. (**20 points)** Use Prover9 to perform automatically the refutation. Submit a report with three parts:

I. Assumptions and goal;

Assumptions:

all x all y (Harley(y) & Rides(x,y) -> Rough(x)).
all x exists y (Biker(x) -> Rides(x,y) & (Harley(y) | BMW(y))).
all x all y (BMW(y) & Rides(x,y) -> Yuppie(x)).
all x (Yuppie(x) -> Lawyer(x)).
all y all x (Rough(x) & Nice(y) -> -Date(y,x)).
Nice(Mary) & Biker(John).

Goals:

-Lawyer(John) -> -Date(Mary,John).

II. The input and output of prover9 (The input of prover 9 should be in plain text)

input -> **prover9_1_input.txt (attached)**
output -> **prover9_1_output.txt (attached)**

III. Conclusion

The query is proved using the assumptions and the goals:
If John is not a lawyer, then Mary does not date John.

Extra-credit: (**30 points)** Use Prover9 to automatically perform the refutation of the following:

The Pigs and Balloons Puzzle

(1) All, who neither dance on tight ropes nor eat penny-buns, are old.
(2) Pigs, that are liable to giddiness, are treated with respect.
(3) A wise balloonist takes an umbrella with him.

(4) No one ought to lunch in public who looks ridiculous and eats penny-buns.
(5) Young creatures, who go up in balloons, are liable to giddiness.
(6) Fat creatures, who look ridiculous, may lunch in public, provided that they do not dance on tight ropes.
(7) No wise creatures dance on tight ropes, if liable to giddiness.
(8) A pig looks ridiculous, carrying an umbrella.
(9) All, who do not dance on tight ropes, and who are treated with respect are fat.

Show that no wise young pigs go up in balloons.

-Lewis Carroll, Symbolic Logic,

Submit a report with three parts:

I. Assumptions and goal;

Assumptions:

all x (-dance(x) | -buns(x) -> old(x)).
all x (pigs(x) & giddiness(x) -> respect(x)).
all x exists y (wise(x) & balloonist(x) -> umbrella(x,y)).
all x (ridiculous(x) & buns(x) -> -lunch(x)).
all x (young(x) & up(x) -> giddiness(x)).
all x (fat(x) & ridiculous(x) & -dance(x) -> lunch(x)).
all x (wise(x) & giddiness(x) -> -dance(x)).
all x all y (pigs(x) & umbrella(x,y) -> ridiculous(x)).
all x (-dance(x) & respect(x) -> fat(x)).
all x (young(x) <-> -old(x)).

Goals:

all x (wise(x) & young(x) & pigs(x) -> -up(x)).

II. The input and output of prover9 (The input of prover 9 should be in plain text)

input -> **prover9_2_input.txt (attached)**
output -> **prover9_2_output.txt (attached)**

III. Conclusion

The query is proved using the assumptions and the goals:
No wise young pigs go up in balloons.