**CSci 384: Artificial Intelligence Spring, 2020**

**Derek Trom Date: March 12th , 2020**

**Due: by the end of day, March 28th, 2020 (Sat.)**

**Home Assignment 4: FOL (190/200 points)**

For any inference with the given story, you can also program it if you prefer it: i.e. story in KB ⊨ query? In such a case, submit your program and its output image with your HW file.

**Q1.** [10/10] Decide each sentence is valid (necessarily true). Justify your answer.

1. (*x x* = *x*)  (*y*  *z* *y* = *z*)
   1. This is valid because the LHS (there is an x that x is itself) is valid and the RHS is valid because for every y there is a z that equals y.
2. *x* Smart(*x*)  (*x* = *x*)
   1. Valid because every x is smart(LHS) or x is x. x=x is a tautology so it will be true always as an inference.

**Q2**.[32/32] For the given English sentence, write them in First-Order Logical sentence which is both syntactically valid and express the meaning correct, using the following predicates and function.

Predicates: In(*x, y*) – A country *x* is in the region *y*.

Borders(*x, y*) - *x* borders *y*.

Person(*x*) - *x* is a person.

HasSSN(*x, y*) - *x* has a social security number *y*.

Occupation(*p, o*) – Person *p* has occupation *o*.

Customer(*p, q*) - Person *p* is a customer of person *q*.

Born(*x, y*) - Person *x* was born in the country *y*.

Parent(*x, y*) - Person *x* is a parent of *y*.

Citizen(*x, y, z*) - Person *x* is a citizen of a country y by *z*.

Resident(*x, y*) - Person *x* is a resident of country *y*.

Constant: SouthAmerica, Europe, UK, Birth, Doctor, Lawyer.

1. [8/8] No region in South America borders any region in Europe.
   1. Where x = region in SA and
   2. ¬ ∃x ∀∃y (Borders(x, y)) ∧ In(c, South America) ∧ In(d, Europe)

¬[∃ c, d In(c, SouthAmerica) ∧ In(d, Europe) ∧ Borders(c, d)].

∀ c, d [In(c, SouthAmerica) ∧ In(d,Europe)] ⇒ ￢Borders(c, d)].

∀ c In(c, SouthAmerica) ⇒ ∀ d In(d,Europe) ⇒ ￢Borders(c, d).

1. [8/8] There is a lawyer all of whose customers are doctors.
   1. ∃p Person(p) ∧ Occupation(p, Lawyer) ∧ ∀ Customer(q, p)  Occupation(q, Doctor)
2. [8/8] No two people have the same social security number.
   1. ¬ ∃x¬ ∃y¬ ∃z Person(x) ∧ Person(y) ∧¬(x=y) ∧ (HasSSN(x,z) ∧ HasSSN(y,z))
3. [8/8] A person born in the UK, each of whose parents is a UK citizen or a UK resident, is a UK citizen by birth.
   1. ∀x Person(x) ∧ Born(x, UK) ∧(∀y Parent(y, x) ((∃z Citizen(y, UK, z)) ∨ Resident(y, UK)))  Citizen(x, UK, Birth)

**Q3.** [8/8] Translate the given FOL sentence into *good and natural English* that is understandable in the common sense, **without using variables *x’*s or *y’*s.**

∀ *x, y, l* SpeaksLanguage(*x, l*)  SpeaksLanguage(*y, l*)  Understands(*x, y*)  Understands(*y, x*)

1. For all people who speak the same language, they understand each other.

**Q4.** [8/10] Write the following sentence in the FOL sentence, using the predicates Radical(x), Electable(x), Conservative(x):

A person who is radical is electable if (s)he is conservative, but otherwise is not electable.

1. ∃p (Radical(p)  Conservative(p))  Electable(p)

∀x Radical(x) 🡪( (Electable(x) 🡨 Conservative(x)) ∧ (¬ Electable(x) 🡨 ¬ Conservative(x)) )

|-- ∀x Radical(x) 🡪 (Electable(x) 🡨 Conservative(x)) ∧ (Electable(x) 🡪 Conservative(x))

|-- ∀x Radical(x) 🡪 (Electable(x) ↔ Conservative(x))

x Radical(x) → (Electable(x) ↔ Conservative(x))

**Q5.** [12/12] Find the **MGU** (most general unifier) in each pair of sentence below or justify why unification is not possible.

1. [6/6] Knows(Father(*y*), *y*), Knows(*x, x*)
   1. The MGU does not exist because unification fails at y with Father(y).
2. [6/6] Wines(*x, y*) vs. Wines(*y, x*)
   1. Rename Wines(y,x) with Wines (a, b) then MGU = {x/a, y/b}

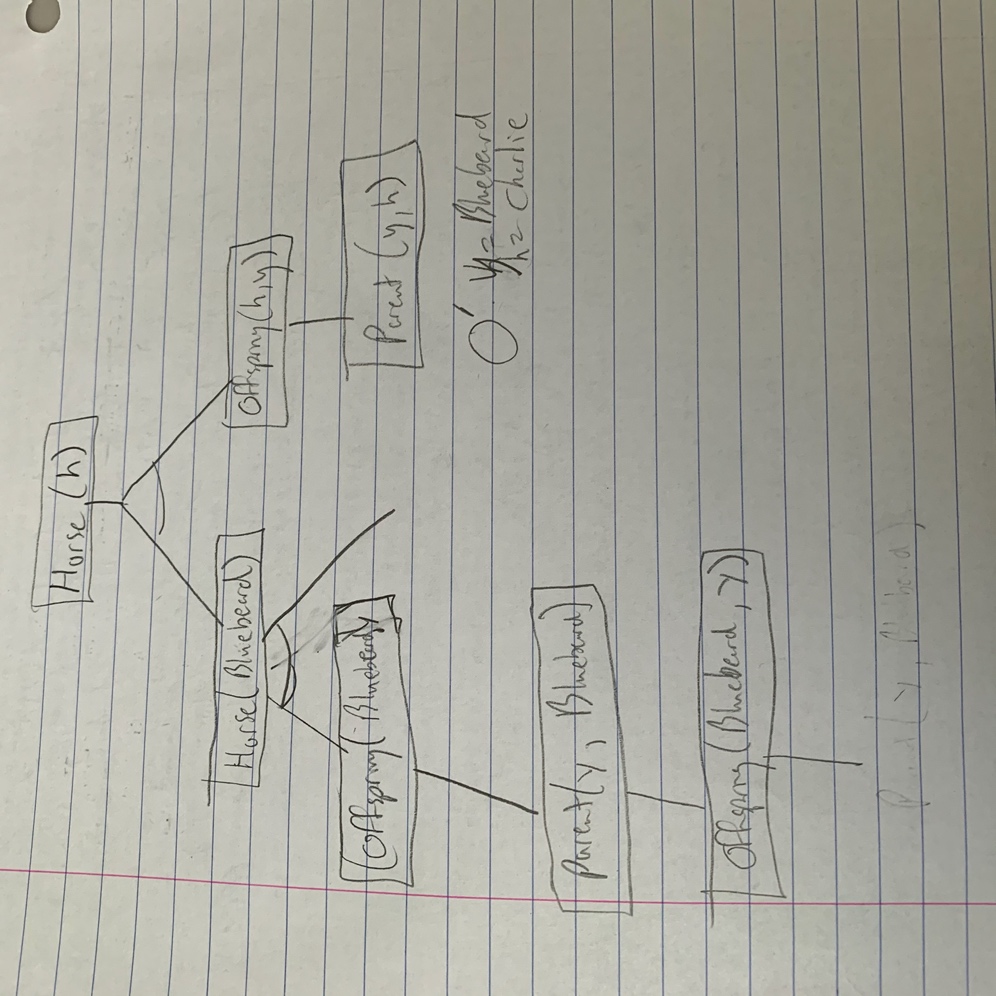
**Q6.** [18/18] Write down the following sentences in the 1st -order logical representations, suitable for their use with ***Generalized Modus Ponens***, i.e. in Horn clauses. Do NOT convert them in CNF.

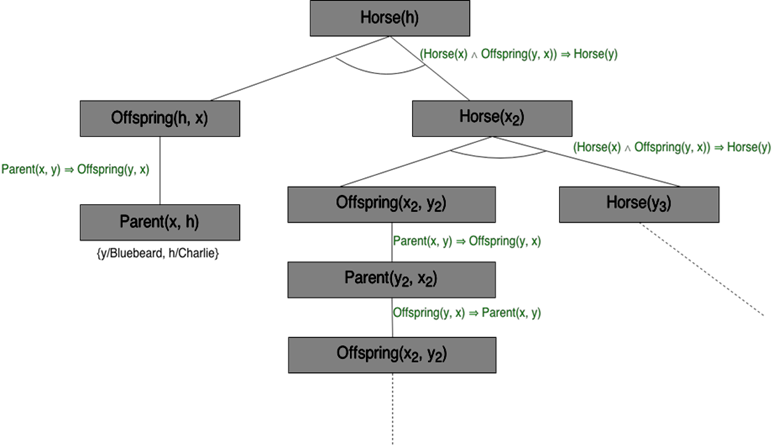
1. Horses, cows, and pigs are mammals.
   1. Horse(x)  Mammal(x)
   2. Cow(x)  Mammal(x)
   3. Pig(x) Mammal(x)
2. An offspring of a horse is a horse.
   1. ∀x ∀z (Horse(z) ∧ Offspring(x,z))  Horse(x)
3. Bluebeard is a horse.
   1. Horse(Bluebeard)
4. Bluebeard is Charlie’s parent.
   1. Parent(Bluebeard, Charlie)
5. Offspring and parent are inverse relations.
   1. Parent(x,y)  Offspring(y, x)
   2. Offspring(x,y)  Parent (y,x)
6. Every mammal has a parent.
   1. Mammal(x)  Parent(f(x), x)

**Q7.** [20/20 pt] From the sentences you wrote in Q6, answer the following question using a ***backward-chaining algorithm***.

1. [10/10] Draw the proof tree generated by a backward chaining algorithm for the query

 *h,*  *Horse(h),* where clauses are matched in the order given.





1. [5/5] What do you notice about this domain?
   1. This could keep going on forever.
2. [5/5] How many solutions for *h* actually follow from your sentences?
   1. 2 because Horse(h) branches out two branches. h/Bluebeard and h/Charlie will be filled in on the tree.

**Q8.** [28/30] From ”Horses are animals”, it follows that ”The head of a horse is the head of an animal.”.

Demonstrate that this inference is valid by carrying out the following steps:

1. [10/10] Translate the premise and the conclusion into the language of 1st -order logic. Use three predicates:

*HeadOf(h,x)* - meaning *h* is the head of *x*.

*Horse(x)*

*Animal(x)*

1. ∀x Horse(x) => Animal(x)
2. ∀h ∀x (Horse(h) ∧ HeadOf(x,h)) => ∃a (Animal(a) ∧ HeadOf(x, a))

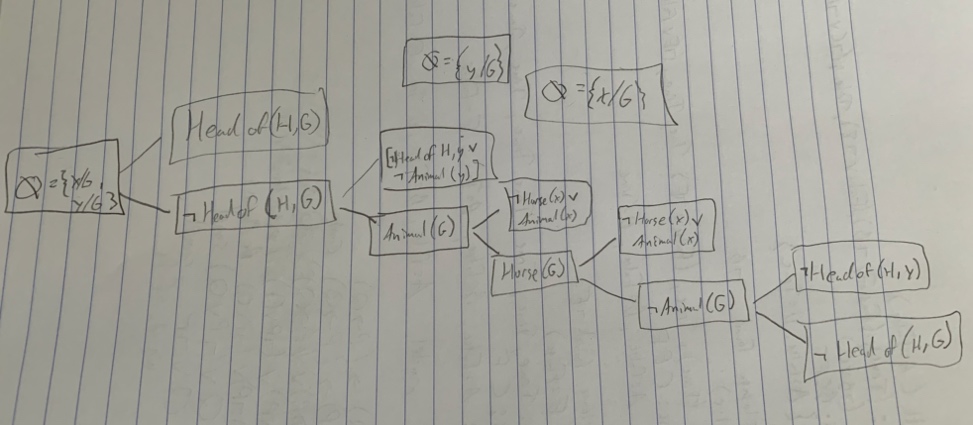
1. [9/10] Negate the conclusion, and convert the premise and the negated conclusion into conjunctive normal form (CNF).
   1. ¬Horse(x) ∨ Animal(x)
   2. ¬∀x ∀h {[Horse(x) ∧ HeadOf(h, x)] ⇒ ∃y [Animal(y) ∧ HeadOf(h, y)]}
   3. ≡ ∃x ∃h ¬{[Horse(x) ∧ HeadOf(h, x)] ⇒ ∃y [Animal(y) ∧ HeadOf(h, y)]}
   4. ≡ ∃x ∃h ¬{¬[Horse(x) ∧ HeadOf(h, x)] ∨ ∃y [Animal(y) ∧ HeadOf(h, y)]}
   5. ≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ¬∃y [Animal(y) ∧ HeadOf(h, y)]}
   6. ≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ∀y ¬[Animal(y) ∧ HeadOf(h, y)]}
   7. ≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ∀y [¬Animal(y) ∨ ¬HeadOf(h, y)]}

Since x and h are existentially quantified, skolemize them by Skolem constants G and H, respectively.

So, the CNF form is:

Horse(G)  Headof(H, G)  (Animal(x)  Headof(H, y)) with 3 clauses:

**Horse(G), HeadOf(H, G), ~ Animal(x) V ~ HeadOf(H, y).**

1. [9/10] Use ***resolution*** to show that the conclusion follows from the premise. Draw the *proof tree of resolution,* showing the *substitutions*; see Figure 9.11-12 in the textbook.
   1. ***¬***HeadOf(H,y), HeadOf(H,G) == ***¬***Animal(G)
   2. *¬*Animal(G),***¬***Horse(x) ∨ Animal(x) == ***¬***Horse(G)
   3. Horse(G), [¬Horse(x) ∨ Animal(x)] ⊢ Animal(G) , so θ={x/G}
   4. Animal(G), [¬Animal(y) ∨ ¬HeadOf(H, y)] ⊢ ¬ HeadOf(H, G) , so θ={y/G}
   5. ¬ HeadOf(H, G), HeadOf(H, G) ⊢ *empty* clause , so θ={x/G, y/G}
   6. 

Resolve A3 and A4:

HeadOf(H,G) and ¬Animal(y) ∨ ¬HeadOf(H, y) ⊢ ¬Animal(G) with ={y/G}

Now, a new fact A5: ¬Animal(G)

Resolve A5 with A1:

¬Animal(G) and ¬Horse(x) ∨ Animal(x) ⊢ ¬ Horse(G) with ={x/G}

Now, a new fact A6: ¬ Horse(G)

Resolve A6 with A2:

¬ Horse(G) and Horse(G) ⊢ { } Contradiction!

So, ∀*x, h Horse(x)* ∧ *HeadOf(h, x) ⇒* ∃*y Animal(y)* ∧ *HeadOf(h, y)*  is true with animal y, ={y/G}, i.e. The head of a horse is the head of an animal.

**Q9.** [34/40] Given sentences (A – E) below,

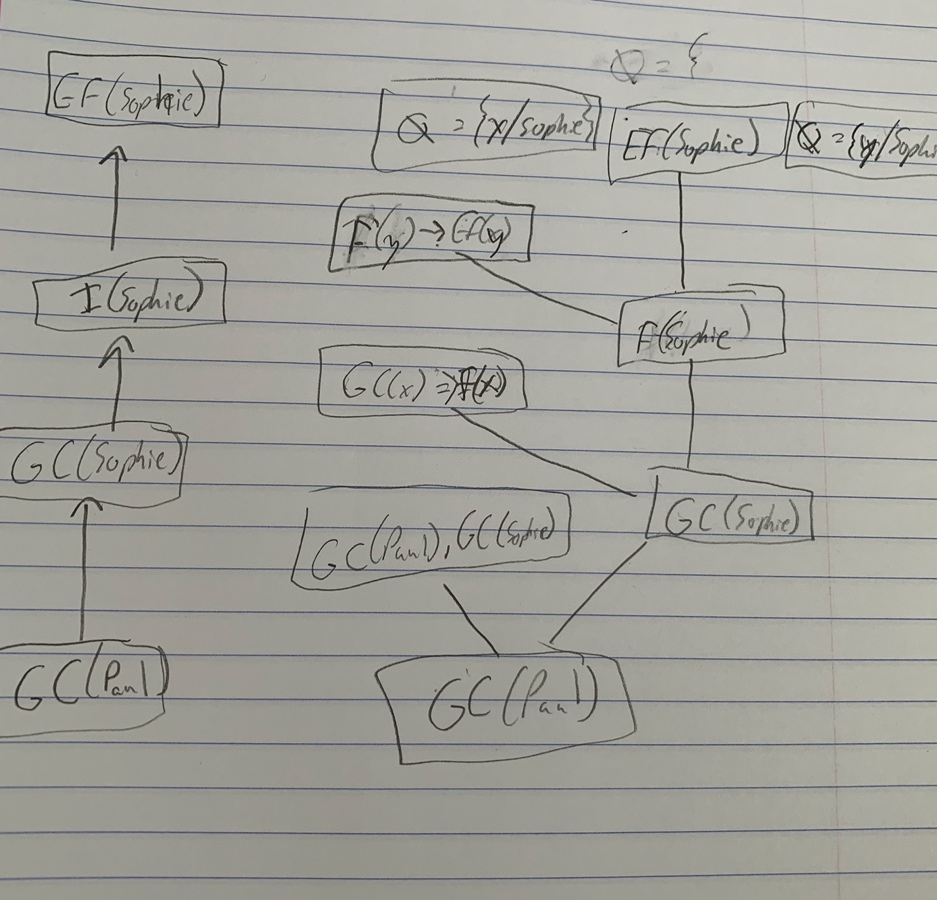
* 1. All great chefs are French.
  2. All Frenches enjoy good food.
  3. Paul or Sophie is a great chef.
  4. Paul is not a great chef.
  5. Query: Who enjoys a good food?

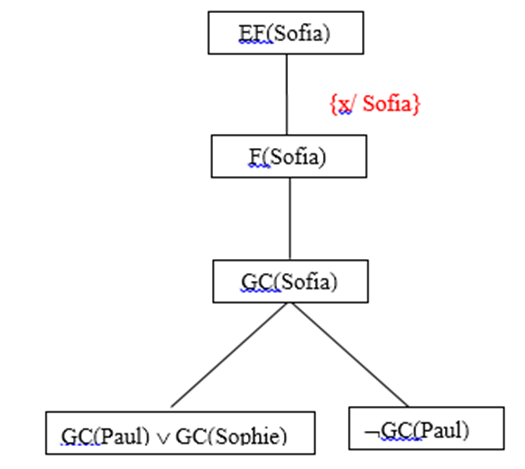
1. [10/10] write them in their FOL sentences using the following predicates and constants:

Predicates: GC – great chef(s), F – French, EF – enjoy good food,

Constants: Paul, Sophie

* 1. ∀x GC(x) => F(x)
  2. ∀x F(x) => EF(x)
  3. GC(Paul) ∨ GC(Sophie)
  4. ***¬***GC(Paul)
  5. ∃z EF(y) // z EF(z) ?

1. [8/10] Answer the query by ***forward chaining*** method. Draw the proof trees showing the substitutions step by step. Refer to the slides #27 - #29.
   1. 
   2. Thus Sophie enjoys good food. By ={x/Sophie} and ={y/Sophie}



1. [8/10] Convert the sentences in 1) to the definite clauses in CNF, suitable for Knowledge\_Base through Skolemization, etc. if necessary. Refer to the slides #46 - #47 and #49 - #52.
   1. ******GC(x)  F(x)
   2. F(x)  EF(x)
   3. GC(Paul)  GC(Sophie)

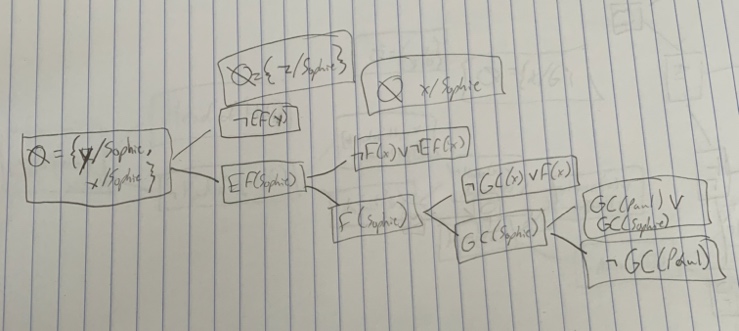
∀x [GC(x)  F(x)] ≡ ¬GC(x)  F(x)

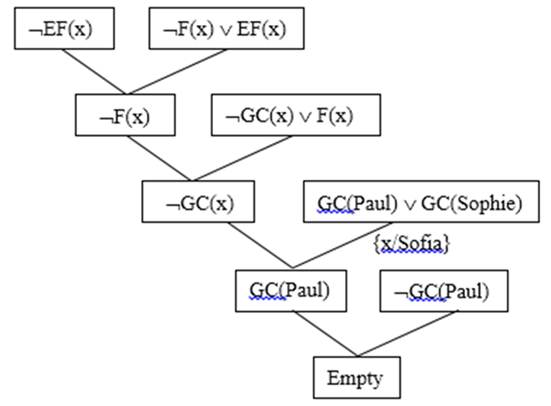
∀y [F(y)  EF(y)] ≡ ¬F(y)  EF(y)

GC(Paul)  GC(Sophie)

¬GC(Paul)

Negation of Query:  EF(z)

1. [8/10] By applying ***resolution****,* answer the query. Show the steps of proof by drawing the resolution tree. Refer to the figures in the slides #48 & #53 of Chap. 9.
   1. Negation of query: ***¬*** EF(y)
   2. ***¬***GC(Paul), GC(Paul) ∨ GC(Sophie) == GC(Sophie)
   3. GC(Sophie), ***¬***GC(x) ∨ F(x) == F(Sophie) ={x/Sophie}
   4. F(Sophie) , ***¬***F(x) ∨ EF(x) == EF(Sophie) ={x/Sophie}
   5. EF(Sophie), ***¬*** EF(y) == empty clause ={y/Sophie, x/Sophie}
   6. Thus Sophie enjoys good food.
   7. 



**Q10.** [20/20] Suppose that the sentence A in Q9 is changed to:

A1. *Some* great chefs are French.

1. [6/6] Write it in the FOL sentence.
   1. ∃x GC(x) ∧ F(x)
2. [6/6] Convert 1) to the definite clause in CNF, suitable for Knowledge\_Base through Skolemization, etc. if necessary.
   1. GC(K), F(K) where ∃x is Skolemized by a constant K.
3. [8/8] Prove how the same query can be answered (or not). Justify your answer step by step.
   1. The query cannot be answered because when
      1. ¬ EF(x) is resolved with ¬ EF(y) ∨¬F(y) yields ¬F(y) with ={x/y}
      2. ¬I(Y) is resolved with I(K) yields null with{y/K}
      3. Thus it is not answered because K is not in KB.