**CSci 384: Artificial Intelligence** Spring, 2016

Instructor: Dr. M. E. Kim Date: April 5th, 2016

**Due: 24:00 PM (Midnight), April 19th (Tue.), 2016**

**Home Assignment 4: FOL & Inference in FOL (143/200 points)**

**Q1.** 6/ [10] Decide each sentence is valid (necessarily true). Justify your answer.

1. 1/ (∃*x x* = *x*) → (∀*y* ∃ *z* *y* = *z*)

This sentence is not valid.

x implies nothing on y or z

Valid.

The LHS is valid by itself; so, the whole sentence is valid iff the RHS is valid.

The RHS is valid because for every value of y in any given model, there is a z which that is identical to y, i.e. y itself.

1. 5/ ∀*x* Smart(*x*) ∨ (*x* = *x*)

This sentence is valid.

Either x is x or every x is smart. This has to be true because x=x is always true.

**Q2**.**14/** [40] 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. 3/ [10] No region in South America borders any region in Europe.
   1. ¬ ∃x ~~∀~~∃y (Borders(x, y)) where x = region in SA and y = region in Europe

∧ In(c, SouthAmerica) ∧ In(d, Europe).

Since negation applies to the whole sentence, both quentifiers are existential.

1. 3/ [10] There is a lawyer all of whose customers are doctors.
   1. ∃q ∀p (Customer(p, q) where q = lawyer and p = doctor

∃ p Person(p) ∧ Occupation(p, Lawyer) ∧ ∀ q Customer(q, p) ⇒ Occupation(q, Doctor).

1. 4/[10] 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))
2. 4/ [10] 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 ∃a ∃b Person(x) ∧ Born(*x, UK*) ∧ Person(a) ∧ Person(b) ∧ Parent(a,x) ∧ Parent(b,x) ∧ ( Citizen(*a, UK*) ∨ Resident(a*, UK*)) ∧ ( Citizen(b*, UK*) ∨ Resident(b*, UK*)) ∧ Citizen(*x, UK, birth*)

∀ x Person(x)∧Born(x,UK)∧(∀ y Parent(y, x) ⇒ ((∃ r Citizen(y, UK, r))∨Resident(y,UK))) ⇒ Citizen(x,UK,Birth).

**Q3.** [10] Translate the given FOL sentence into good and natural English without using variables *x*s or *y*s.

∀ *x, y, l* SpeaksLanguage(*x, l*) ∧ SpeaksLanguage(*y, l*) ⇒ Understands(*x, y*) ∧ Understands(*y, x*)

All of one group of people speak a certain language. All of another group of people speak the same language. Thus, all of the first group of people understand the second group and all of the second group understands the first group.

People who speak the same language understand each other.

**Q4.** 0/ [10] Find the **MGU** (most general unifier) in each pair of sentence below or justify why unification is not possible.

1. 0/ Older(Father(*y*), *y*), Older(Father(*x*), John)

This cannot be unified as Father(y)/Father(x) and y/John fail.

{x/y, y/John} ➔ {y/John, x/John}.

1. 0/ Knows(Father(*y*), *y*), Knows(*x, x*)
   1. θ = {x/Father(y), y/x}

No unifier because x can’t be substituted both by y and by Father (y) as y occurs in Father(y)

**Q5.** [10] Convert the following sentence in FOL to the sentences in **CNF**.

∀w { [(P1(w) ∨ P2(w)) ⇒ P3(w)] ∨ [ ∃x ∃y (P3(x,y) ⇒ P4(w, x))]} ∧ [∀w P5(w)].

Standardize Variables: ∀w { [(*¬*P1(w) ∧ *¬*P2(w)) ∨ P3(w)] ∨ [ ∃x ∃y (*¬*P3(x,y) ∨ P4(w, x))]} ∧ [∀z P5(z)]

Skolemization: ∀w { [(*¬*P1(w) ∧ *¬*P2(w)) ∨ P3(w)] ∨ [(*¬*P3(F(w), G(w)) ∨ P4(w, F(w)))]} ∧ [∀z P5(z)]

Drop universal quantifiers: [(*¬*P1(w) ∧ *¬*P2(w)) ∨ P3(w)] ∨ [(*¬*P3(F(w), G(w)) ∨ P4(w, F(w)))] ∧ [P5(z)]

Distribute ∧ over ∨: [*¬*P1(w) ∨ P3(w) ∨ *¬*P3(F(w), G(w)) ∨ P4(w, F(w))] ∧ [*¬*P2(w) ∨ P3(w) ∨ *¬*P3(F(w), G(w)) ∨ P4(w, F(w))] ∧ P5(z)

**Q6. [12/** 15] 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. - ∀x (Horse(X)∨Cow(x)∨Pig(x)) => Mammal(x)

Not in Horn clauses -1

Horse(x) ⇒ Mammal(x), To apply GMP, the LHS should be a conjunction of sentences,

Cow(x) ⇒ Mammal(x), not disjunction.

Pig(x) ⇒ Mammal(x). So, divide it to 3 sentences to the suitable use of GMP because

(A ∨ B ∨ C) → D is equivalent to (A → D) & (B → D) & (C → D)

1. An offspring of a horse is a horse. - ∀x ∀z (Horse(z) ∧ Offspring(x,z)) => Horse(x)
2. Bluebeard is a horse. - Horse(Bluebeard)
3. Bluebeard is Charlie’s parent. - Pig(Bluebeard,Charlie)
4. Offspring and parent are inverse relations. - ∀x ∀z Offspring(x,z) <=> Pig(z,x)

Offspring(x, y) ⇒ Parent(y, x) Not in Horn clause. Divide it to 2 sentences -1

Parent(x, y) ⇒ Offspring(y, x).

1. Every mammal has a parent. - ∀x Mammal(x) ⇒ ∃y Parent(y,x) **-1**

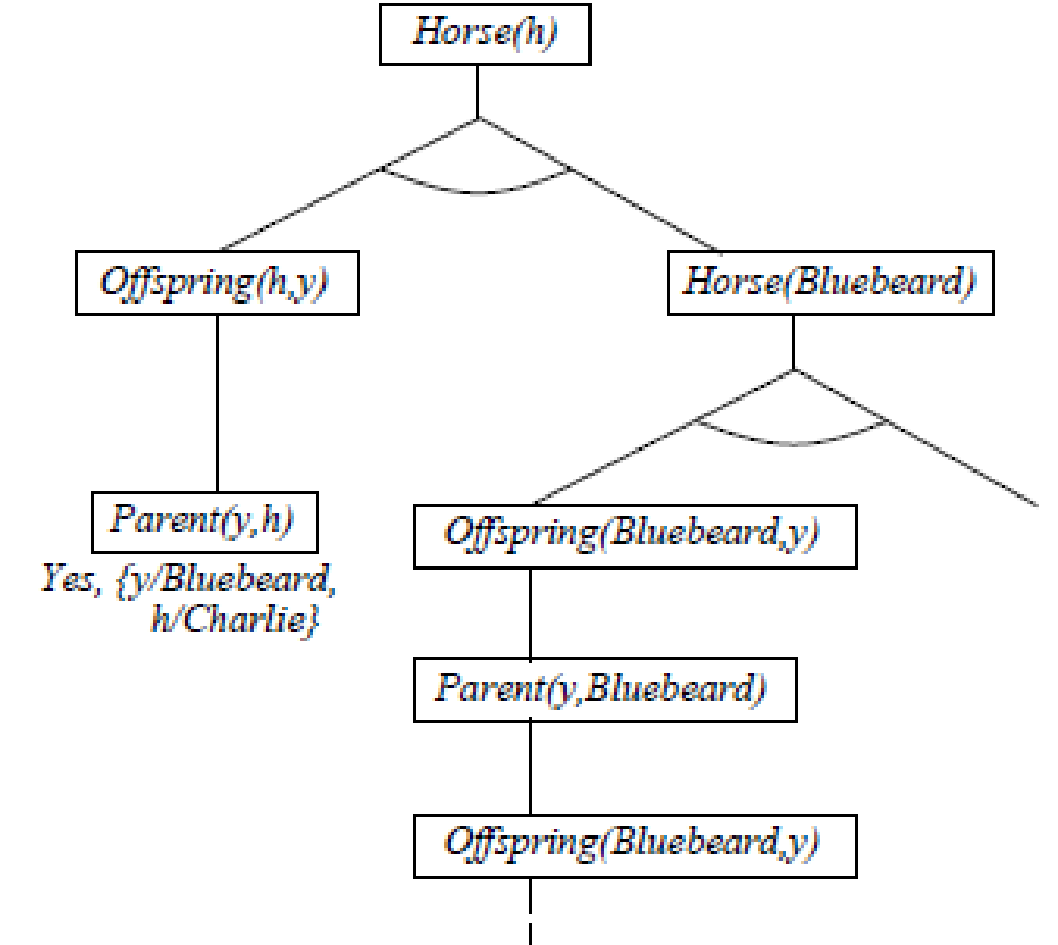
Similarly, the suitable sentence for GMP is the sentence after skolemization for ∃.

Mammal(x) ⇒ Parent(G(x), x)

**Q7. 15/** [20 pt] From the sentences you wrote in Q6, answer the following question using a ***backward-chaining algorithm***.

1. 5/ [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. 

Your tree is not exactly correct. Show the substitutions, too.



1. [5] What do you notice about this domain?
   1. It’s an infinite domain
2. [5] How many solutions for *h* actually follow from your sentences?
   1. There should be 2 solutions, (h/Bluebeard) and (h/Charlie). How?

**Q8. [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] 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)*

∀x Horse(x) => Animal(x)

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

1. [10] Negate the conclusion, and convert the premise and the negated conclusion into conjunctive normal form (CNF).

***¬***Horse(x) ∨ Animal(x)

¬∀x ∀h {[Horse(x) ∧ HeadOf(h, x)] ⇒ ∃y [Animal(y) ∧ HeadOf(h, y)]}

≡ ∃x ∃h ¬{[Horse(x) ∧ HeadOf(h, x)] ⇒ ∃y [Animal(y) ∧ HeadOf(h, y)]}

≡ ∃x ∃h ¬{¬[Horse(x) ∧ HeadOf(h, x)] ∨ ∃y [Animal(y) ∧ HeadOf(h, y)]}

≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ¬∃y [Animal(y) ∧ HeadOf(h, y)]}

≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ∀y ¬[Animal(y) ∧ HeadOf(h, y)]}

≡ ∃x ∃h {[Horse(x) ∧ HeadOf(h, x)] ∧ ∀y [¬Animal(y) ∨ ¬HeadOf(h, y)]}

≡ Horse(G) ∧ HeadOf(H,G) ∧ [¬Animal(y) ∨ ¬HeadOf(H, y)]

1. [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.

***¬***HeadOf(H,y), HeadOf(H,G) == ***¬***Animal(G)

*¬*Animal(G),***¬***Horse(x) ∨ Animal(x) == ***¬***Horse(G)

Horse(G), [¬Horse(x) ∨ Animal(x)] ⊢ Animal(G) , so θ={x/G}

Animal(G), [¬Animal(y) ∨ ¬HeadOf(H, y)] ⊢ ¬ HeadOf(H, G) , so θ={y/G}

¬ HeadOf(H, G), HeadOf(H, G) ⊢ *empty* clause , so θ={x/G, y/G}



**Q9. 31/** [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] write them in their 1st -order logical representations using the following predicates and constants:

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

Constants: Paul, Sophie

∀x GC(x) => F(x)

∀x F(x) => EF(x)

GC(Paul) ∨ GC(Sophie)

***¬***GC(Paul)

∃z EF(z)

1. 1/ [10] Answer the query by ***forward chaining*** method. Draw the proof trees showing the substitutions step by step. Refer to the slide #23 - #25.

GC(x), F(x) == EF(x)

EF(x), ***¬***GC(Paul) == GC(Sophie)

GC(Sophie), ***¬*** GC(Sophie) == empty clause so θ={~~GC~~/Sophie, ~~EF~~/Sophie} GC and EF are predicateds, not variables.

You should do reasoning by forward chaining, not by resolution.

No rule whose LHS is EF(x). So, you can’t start from EF(x). Start from the known facts.



GC(Paul) ∨ GC(Sophie) with ~GC(Paul) ==➔ GC(Sophie)

(≡ ~GC(Paul) → GC(Sophie))

GC(Sophie) with GC(x) → F(x) ==➔ F(Sophie) by θ={x/Sophie}

F(Sophie) with F(y) → EF(y) ==➔ EF(Sophie) by θ={y/Sophie}

1. [10] Convert the sentences in 1) to the definite clauses in CNF, suitable for Knowledge\_Base through Skolemization, etc. if necessary. Refer to the slide #42.

***¬***GC(x) ∨ F(x)

***¬***F(x) ∨ EF(x)

GC(Paul) ∨ GC(Sophie)

¬GC(Paul)

Negation of Query: ¬ EF(z)

1. [10] By applying ***resolution****,* answer the query. Show the steps of proof by drawing the resolution tree. Refer to the figure in slide #43 of Chap. 9.

Negation of query: ***¬*** EF(z)

***¬***GC(Paul), GC(Paul) ∨ GC(Sophie) == GC(Sophie)

GC(Sophie), ***¬***GC(x) ∨ F(x) == F(Sophie) θ={x/Sophie}

F(Sophie) , ***¬***F(x) ∨ EF(x) == EF(Sophie) θ={x/Sophie}

EF(Sophie), ***¬*** EF(z) == empty clause θ={z/Sophie, x/Sophie}



**Q10.** [15] Suppose that the sentence A in Q9 is changed to:

A1. *Some* great chefs are French.

1. [5] Write it in the FOL sentence.

∃x GC(x) => ∧ F(x)

1. [5] Convert 1) to the the definite clause in CNF, suitable for Knowledge\_Base through Skolemization, etc. if necessary.

GC(K), F(K) where ∃x is Skolemized by a constant K.

1. [5] Prove how the same query can be answered (or not). Justify your answer step by step.

***¬***GC(Paul), GC(Paul) ∨ GC(Sophie) == GC(Sophie)

GC(Sophie), ∃x GC(x)

We find that this answers ‘someone enjoys good food’, but not that Sophie enjoys good food. Inference is unsuccessful -- too general answer.

A query is ∃z EF(z) , not ∃x GC(x).

Sophie is a good chef, it’s unknown whether she is French because only some chefs are French.

So, we can’t conclude that she enjoys a good food.

In inference, GC(Sophie) can’t be further resolved with any other clauses because the fact GC(K) is already skolemized with K.