Session 1 – Solutions

1 FO

1.1 Translation of natural language to first-order logic

1. $\neg Happy(John) \wedge \neg Happy(Peter)$
2. $\neg(Happy(John) \wedge Happy(Peter))$
3. $Happy(John) \Rightarrow Happy(Peter)$
4. $\forall x, y : (Man(x) \wedge Veg(y)) \Rightarrow Likes(x, y)$
5. $\forall x, y : (Man(x) \wedge \neg Butcher(x) \wedge Veg(y)) \Rightarrow Likes(x, y)$
6. $\exists x : Man(x) \wedge Butcher(x) \wedge Veg(x)$
7. $\neg\exists x : Man(x) \wedge Butcher(x) \wedge Veg(x)$
8. $\forall x, y : (Man(x) \wedge Woman(y) \wedge Veg(y)) \Rightarrow \neg Likes(x, y)$
9. $\neg\forall x : Men(x) \wedge Veg(x) \Rightarrow Happy(x)$
10. $\forall x : (\exists y : Butcher(y) \wedge Likes(x, y)) \Rightarrow Man(x)$
11. $\exists x : Butcher(x) \wedge \forall y : (Veg(y) \Rightarrow Likes(x, y)$.
12. $\forall y : Veg(y) \Rightarrow Likes(John, y)$
13. $\forall y : Veg(y) \Rightarrow \neg Likes(John, y)$
14. $\forall x : Butcher(x) \Rightarrow \neg Veg(x)$
15. $\forall x : Butcher(x) \Rightarrow \exists y : Veg(y) \wedge Likes(y, x)$

1.2 Intermezzo

1. Intuitively, `John` is an object and happy is a property, which correspond to a set of objects (the ones that have that property). This suggests to make `happy` into a unary predicate and `John` an identifier/constant.
2. A first attempt would give something as `John(43)` both for age and shoe size. Other wrong attempts include `John(shoeSize(43))`, `John(age(43)`, however, this gives rise to what then shoeSize(43) denotes.

3. Two sets with the same elements are equal. Or differently said, different sets have different elements. If `John` and `Bob` are sets (unary predicates), then they are mathematically speaking identical. This is of course not the case.

4. This is not directly possible since first order logic does not allow for predicates over predicates (second-order logic does allow this). If we model it with `John(Friend(Bob))`, we still have some asymmetries to resolve and we need ask what kind of object `Friend(Bob)` is. It is a term without demonstrable denotation in the world.

5. First order does not allow quantification over sets (or unary predicates).

Each of these 5 are a sufficient reason why `John(happy)` is a bad representation/modelling.

Some advices when choosing a vocabulary

- Divide your world into categories, types, relations and functies. Check over which types you need to quantify. These types should be basis types in FO.

- Choose symbols that have a clear and intuitive denotation in the application domain. Then, terms formed with your symbols will also have a clear denotation. (`Friend(Bob)` in contrast does not have this clear denotation)

- For basis symbols, decide whether they should be identifiers or symbolic constants. For example, in the application domain, we know who is denoted by the symbol `John`, and this is different from the one who is denoted by the symbol `Bob`; they are identifiers. But if there is a symbol `Murderer` and we do not know who it is, although we might know some things (e.g. (s)he is right-handed). This is a symbolic constant.

- When you represent a kind of objects as a set, relation or function, keep in mind with the following: verify that two objects that both represent a set, cannot contain the same elements. If this is possible, the representation is not good.

- Try to choose your vocabulary as simple as possible: keep the number of arguments low.

1.3 Quantors

1. True, take $y = x$

2. True, take $y = x$

3. True, take $y = 0$

4. False, there is no biggest natural number

5. False, there is no natural number strictly smaller than $y = 0$
6. True, take $y = x + 1$
7. False, we can take $y = x$
8. False, we can take $y = x$
9. True, this is equivalent to saying $\exists y : \exists x : x > y$, this is true with $x = 1, y = 0$
10. True, this is equivalent to saying $\exists y : \exists x : x > y$, this is true with $x = 1, y = 2$
11. False, this is equivalent to $\forall x : \forall y : x < y$, which fails for $x = y$
12. False, this is equivalent to $\forall x : \forall y : x > y$, which fails for $x = y$

2 Structures

For a given vocabulary, a structure over that vocabulary is an assignment of values to the symbols in the vocabulary. This is a mathematical abstraction of the state of affairs. For the state of affairs implied by the statements given below, write a structure, abstracting this state of affairs, over the following vocabulary:

- Person/1
- Age/2
- Friends/2
- Oldest/0:

An example of a structure over this vocabulary:

- Person = {Bert; Ernie}
- Age = {(Bert, 20); (Ernie, 21)}
- Friends = {}
- Oldest = Ernie

Create a structure satisfying all the following statements at once:

- An is 16 years old and friends with Pete, who is older.
- Everyone who has friends is a person
- Fred, who is 14, does not have any friends, he is still a person though.
- Betty has two friends, she is younger than both of them.

- Every person has an age.
- The earth is older than any person.
- No one is friends with someone who is not friends with them.
- The objects discussed here are the only ones we know anything about.
- When deciding who is the oldest, only people are compared.

Solution :

- Person = {An, Pete, Fred, Betty}
- Age = {(An, 16), (Pete, 20), (Fred, 14), (Betty, 15), (earth, 4.5×10^9)}
- Friends = {(An, Pete), (Pete, An), (Betty, An), (An, Betty), (Betty, Pete), (Pete, Betty)}
- Oldest : Pete