First Order Logic

Yashovardhan Srivastava

25 May,2021

Introduction

In this unit,we will use a different representation language to illustrate concepts of logic and knowledge.We saw how propositional logic works and how inference could be derived from it,now we examine **First Order Logic -** which is more expressive than propositional logic and is a sufficiently good model to represent the real world.

Nature of Representation Languages

Propositional Logic like many programming languages are **declarative** in nature.A declarative language lacks general mechanisms for deriving facts from other facts.Programs and systems are designed in way that details of the program/system are derived by the programmer from their own knowledge in the domain.Knowledge and Inference are separate.Propositional lacks the power to concisely describe an environment with many objects and rules are repetitive and are far from what could be expressed in natural language.First Order Logic makes it possible to describe sentences similar to how we do using natural language.

Natural Language

Natural Languages are very expressive in nature.In Linguistics,natural languages are viewed as a declarative knowledge representation language.The modern view of language is that it serves as a medium of **communication** rather than **representation.**The semantics of a sentence depend on sentence and the context in which it is spoken,which is a big hindrance if we consider how are we going to represent context itself.Natural languages also suffer from ambiguity,which is a problem for representational languages.In the next section we will discuss First Order Logic which borrows many ideas from natural language while avoiding its drawbacks.

First Order Logic

First order Logic,as described above,draws many of its concepts from natural language.When we look at the syntax of the natural language,the most obvious elements are nouns and noun phrases that refer to **objects** and verbs and verb phrases refer to **relations** among objects.The language of first order logic is built around objects and their relations.The objects and their relations to model real-life scenarios is of fundamental importance in many branches of science including mathematics,and theoretical computer science(see OOP).

Characterization of Logic

We now discuss characterization of logic and differences between propositional logic and first order logic:

- One way in which both of the logic differ is in **ontological commitment** made by each language.Ontological Commitment refers to what the language assumes to the nature of reality.For propositional logic, assumes that there are certain facts about the world that are either true or they are either false.These facts either hold up or not. First Order logic on the other hand assumes that the world has objects with certain relations that hold up or not.

- A logic can also be characterized by its **epistemological commitments**.

  Epistemological Commitment refers to the possible states of knowledge that it allows with respect to each fact.In both propositional logic and first order logic, a sentence represents a fact that an agent believes to be true,or false or it has no opinion on the fact.There are therefore,three possible states of knowledge regarding any sentence.For systems using probability theory,agents have a degree of belief(between 0 and 1) for any particular state.

Syntax and Semantics

We start this section by specifying more precisely the possible worlds of first order logic and later various elements of the language.

Models in First Order Logic

Models of a logical language constitute the formal structures that constitute the possible worlds under consideration.They link the vocabulary of the sentences to the elements of the possible world.In propositional logic model linked propositional symbols to predefined truth values,in first-order logic models link objects and their relations.The objects in the model may be related in many ways.It can be a unary relation or properties .Objects might also be related to exactly one object,which are best considered as functions.Strictly speaking,models in first-order logic require **total functions**,*i.e* there must be a value for every input.

Symbols and Interpretations

In this section,we will discuss the syntax of first-order logic.The basic syntactic element of first order logic are the symbols that for objects,relations and functions,therefore the three types of symbols are **constant symbols**-which represent objects,**predicate symbols** -which represent relations among objects and **function symbols** that stand for function.Each proposition symbol comes with an **arity** that fixes the number of arguments.As in propositional logic,every model must provide some information to determine whether a sentence is false or not,which is referred to as **interpretation**.A model,therefore must include set of objects and interpretation that maps constant symbols to **objects**,predicate symbols to **relations** on those objects and function symbols to **functions** on those objects.

Terms

A **term** is a logical expression that refers to an object.Constant Symbols are therefore terms.It is to be noted that it is not convenient for all objects(constants) to have a distinct identifier. For example: It would make no difference if we named the spectacles in 'The Old man's spectacles' with a distinct identifier.That is what function symbols are for instead of this ,we might use *Spectacles(Old Man)*.It is important to remember that function symbols (with or without parameters) are not subroutines.In our above example,we can't form a subroutine that returns *Spectacles* without ever defining what it was.λ **expressions** are useful for providing notation to function and predicate symbols.

Atomic Sentences

Now that we have terms for referring to objects and predicate symbols to relations, we can now put them together to form **atomic sentences**.An atomic sentence is formed from a predicate symbol followed by(optionally) parenthesized list to terms.For example:*Mother(Alice,Bob)* states that Alice is the mother of Bob[1].

Complex Sentences

Complex sentences are formed by using logical connectives on atomic sentences(with the same syntax as we did in propositional logic). For example: *Mother(Alice,Bob)* $\wedge$ *Father(Charlie,Alice)* states that Alice is the mother of Bob **and** Charlie is the father of Alice.

[1]It is just a convention that P(x,y) is read as "x is P of y"

Quantifiers

Since enumerating through all the objects names (which grows very fast) is computationally expensive, we want notations to express properties of entire collection of objects.First order logic contains two standard quantifiers that lets us do that,called *universal* and *existential*.

- **Universal Quantifier(∀)**,as its name suggests provides notation for representing sentences that are satisfied in **all** possible models,i.e. the assertion is true in all of the models.∀ symbol is used for expressing general rules that applies to all objects in first order logic.For example:

  $$\forall x \;\; Dogs(x) \Rightarrow Cute(x)$$

  implies that all dogs are cute[2].

- **Existential Quantifier(∃)**,provides notation for representing sentences that are satisfied in **atleast one** of the possible models. ∃ symbol is used for expressing general rules that applies to atleast on object in first order logic.Other symbol such as ∃! are used to express that there exist only one object.For example:

  $$\exists x \; Person(x) \Rightarrow Tall(x)$$

  implies that there exist a person who is tall.

The two quantifiers Universal() and Existential() are intimately connected through negation.For example: asserting that all animals are cute is same as saying there exist no animal that isn't cute,and vice-versa,therefore the two statements are equivalent :

$$\forall x \;\; Animal(x) \Rightarrow Cute(x)$$

$$\neg\exists x \;\; Animal(x) \Rightarrow \neg Cute(x)$$

As the two quantifiers are negation of each other,they obey De-Morgan's rules

Equality

Other than using predicates and terms as described above,we can also use the **equality symbol** to signify that two terms refer to same object.For example:

$$\exists x, y \;\; Brother(x, Alice) \wedge Brother(y, Alice) \wedge \neg(x = y)$$

implies that Alice has atleast two brothers.

Knowledge Engineering in First Order Logic

Knowledge Engineering is a general process of knowledge-base construction.It is an iterative development process which involves identifying tasks,assembling relevant knowledge,forming atomic sentences,debugging the knowledge base among others.The following steps are involved in the Knowledge-Engineering Process:

1. *Identifying the Task:* This step involves specifying the range of questions that the knowledge-base will support and the kind of facts that will be available fore each specific problem instance.

2. *Assemble the relevant knowledge:* This step involves **knowledge acquisition**.*i.e* acquiring knowledge about the knowledge-base to understand it's scope and to understand how the domain actually works.

3. *Deciding on vocabulary of predicates,functions and constants:*This step involves translating domain-level concepts into logic-level names and deciding the *style* of knowledge-engineering.Once the choices have been made,the result is a vocabulary dictionary,also known as the **ontology** of the domain.Ontology determines what kinds of things exist.

[2]A common mistake in these kind of examples is to use conjunction instead of implication

4. *Encode general knowledge about the domain:* This involves writing down axioms for all the vocabulary terms.This solidifies the meaning of the term in vocabulary and also acts as a way for knowledge-engineer to cross-check the hypothesis that were made in step 3 and make adjustments as necessary.

5. *Encode a description of the specific problem instance:* This step involves writing simple atomic sentences about instances of concepts that are already part of the ontology.The instances could be supplied by sensors and knowledge base is supplied with additional sentences in the same way inputs are provided to traditional programs.

6. *Debug the knowledge base:* This step involves careful testing of all sentences,axioms to make sure that our expectation and output match.This might involve moving through the steps again to ensure that all assumptions and their representations are correct.Incorrect axioms give false information about the world and they should be corrected.

References

[1] Russel,Stuart and Norvig,Peter.*Artificial Intelligence-A Modern Approach*(3rd ed.).Pearson Education,2015