Imperial College London

Department of Computing

Third Year Individual Project

LOST: The Logic Semantics Tutor

Author:
Alina Boghiu

Supervisor:
Ian Hodkinson

Second marker:
Fariba Sadri

May 31, 2013

Abstract

The aim of this project was to develop a software tool that can teach the semantics of first order predicate logic to students by helping them visualise the process of sentence evaluation. Thus, the focus was on developing an intuitive and engaging user interface to show and allow modification of structures, signatures and sentences, as well as provide relevant exercises for the student to practice with. The latter is arguably the most important feature of this tool and an addition to the functionality of the previous LOST. The user can now ask to solve 3 automatically generated types of puzzles, within a tutorial environment that measures their progress. Completing each lesson is an actual achievement and provides real confirmation of understanding the semantics of first order logic.

For robustness, the tool is linked to a lesson database which can be accessed by students and lecturers alike. The former also have permissions to add, edit or remove lessons and see their student's progress.

I believe these are firm grounds for many possible extensions (such as a Hintikka game) and can be of real use, standalone or alongside the first year predicate logic course. This report will provide further detail of its implementation and purpose.

Acknowledgements

I would like to express gratitude and appreciation to my supervisor, *Ian Hodkinson*, for his continuous support and motivation that have guided me towards completing this project, as well as to my second marker, *Fariba Sadri*, for her useful remarks and suggestions.

Altogether, the resources, staff and students of the Department of Computing have made working on this project enjoyable every day.

Contents

Chapter 1

Introduction

Chapter 2

Background

2.1 First order predicate logic semantics

In order to understand the product I am aiming for we should first take a look at what first order predicate logic is and why its semantics can be tricky. As an extension of propositional logic, it expresses statements such as *Socrates is a man* in much more detail. While propositional logic would regard this sentence as atomic and simply assign it a truth value, predicate logic provides a way of describing its internal structure and of evaluating it inside a relevant context. It does this by introducing:

- *Constants* - which name the objects (or terms) inside a context (e.g. Socrates)
- *Relations* - which describe properties of the objects they take as arguments or, in the case of nullary relations (which take no arguments), general properties of the structure (e.g. *man* is a unary relation that describes Socrates)

These elements form the signature of a structure (i.e. the technical term for the context we need in order to evaluate a logic formula and assign it a truth value). For a computer scientist it may be easier to understand that the structure contains instances of the signature's elements, as well as other objects, which one can iterate over with variables. Using these concepts, we can now rewrite the sentence as *man(Socrates)*. Two questions arise: First, which is the object that Socrates describes. Second, what does it mean for something to be a man. If we take our structure to be an imaginary world of dwarfs and name one of them Socrates, our sentence would be false. However in the context of the real world where everyone is human and Socrates names the famous philosopher, the sentence is true.

Next, if we want to express *All men are mortal* we must introduce the two quantifiers. This was not possible in first order propositional logic and understanding quantifiers and how the iterate over the objects in a structure is the main source of distress amongst students. The quantifiers are:

- $\exists$ *(Exists)* - which checks that there is at least one object in the structure that satisfies the sentence it refers to and makes it true
- $\forall$ *(For all)* - which checks that all of the objects in the structure satisfy the sentence it refers to and each make it true.

Now the sentence can now be written as $\forall$*(men(x)* $\rightarrow$ *mortal(x))* and we refer to x as a variable which in this case is bound by the for all quantifier. Another aspect that the user must understand is that sentences containing unbound variables are valid but cannot be evaluated to a truth value.

When considering my own aproach, I decided a good product would offer a way of visualising all of this and allow the user to play around with structures and sentences, as experimenting would make it easier to understand the inner works.

2.2 Existing solutions

Chapter 3

Approach and Implementation details

3.1 The Evaluator

The structure of my project's back end naturally lays upon the 3 main parts of logic semantics: structures, signatures and sentences. Together these form the Evaluator, a package which makes them interact correctly. With no front end, it can still take a sentence input through the entire journey to a boolean outcome. In fact this is the journey we will make to better understand its components.

3.2 The User Interface

3.3 The Lesson Generator

3.4 The Lesson Database

Chapter 4

Evaluation

4.1 Quantitative evaluation

4.2 Qualitative evaluation

Chapter 5

Conclusions and Future Work

5.1 Learning outcomes

5.2 Potential improvements

5.3 Potential extensions

Bibliography

Appendix A

Short demo

Appendix B

Program Listings