École Polytechnique
Master 1 - Institut Polytechnique de Paris

December 18, 2020

INF551 - Computational Logic: from Artificial Intelligence to Zero Bugs

Implementing a propositional prover

Salwa Tabet Gonzalez

Introduction

The goal of this project was to implement an interactive propositional prover: the user inputs deduction rules as commands and the program checks the validity of the proof by constructing the corresponding λ-term and typechecking it. We first created a prover for simply typed λ-calculus, then a second one implementing dependent types.

1 Simply typed λ-calculus

For the case of simply typed λ-calculus (parts 1,2,3), I made a fully functional prover, though I did not answer the fourth part of the subject. I managed to write all the proofs that the subject asked with my prover.

I improved the `string_of_ty` and `string_of_expr` functions in order to minimize the amount of parentheses printed. To do so, I created several subfunctions corresponding to the different levels of priority of the operators, similarly to how the parsing code works.

I also improved the first prover in order to let the user save proofs in files and continue them later. When typing `./prover file.proof`, commands in the file are given as arguments to the prover, and the user can type new commands which will be stored at the end of the file. I also automatically added the prefix '`Proofs/`' to the name of the file, so that all files would be stored in the same folder named 'Proofs'.

2 Dependent types

When it comes to dependent types, I had trouble using the second interactive prover, since the interface was less practical. I found it hard to directly type λ-terms without using elimination rules. Due to that, I was not able to prove the associativity and commutativity of multiplication in question 5.14.

If I had more time to work on this project, I would have improved the second prover to make it work like the first one, showing the context and the goal type at each stage, and letting the user enter commands for introduction and elimination rules. I would have especially added a command for the elimination of the transitivity of equality.