Project 3

Michael Hrishenko, LaTeX WIZARD

31 January 2020

Abstract

This report begins actual proofs and theorems in HOL and ML, using logic structures and principles to achieve desired ends. Several new tools including Holmake and EmiTex are introduced. Our goal is to show how ML code and HOL theorems are reversible and reproducible in order to create assured code by design.

Acknowledgements: Professors Marvine Hamner, Shiu-Kai Chin, & Susan Older

Contents

Chapter 1

Executive Summary

Unable to complete problems 8.4.2 & 8.4.3 due to time constraints. Problem 8.4.1 completed but unable to compile properly given Holmake errors:

```
\% Holmake                                                          4:22:11
chapter8Theory                                real:    0s  user:    0sFAIL<1>
 Couldn't find required output file: /home/troy/dev/ms500/CIS-634_AsrFound/03_Project/chapter8Theory.sml
```

The file "chapter8Script.sml" does contain a valid proof for 8.4.1, it was verified in the HOL REPL just not compiled due to the error above. Continuing to troubleshoot Holmake, however already wasted enough time and focusing now on completing problems. All problems from Chapter 7 completed successfully, no errors to report. Many thanks to Professor Hamner for helping me work around my ever-changing schedule.

Chapter 2

Exercise 7.3.1

2.1 Problem Statement

1. Create a function andImp2Imp term that operates on terms of the form $p \wedge q \Rightarrow r$ and returns $p \Rightarrow q \Rightarrow r$.

2.2 Relevant Code

```
fun andImp2Imp exp1 =
let
        val (im1,im2) = dest_imp exp1
        val (im3,im4) = dest_conj im1
        val im5 = mk_imp (im4,im2)
in
        mk_imp (im3,im5)
end;
```

2.3 Test Cases

```
val test1 = ``(p /\ q) ==> r``;
andImp2Imp test1;
```

2.4 Execution Transcripts

```
> andImp2Imp test1;
val it = ``(p :bool) ==> (q :bool) ==> (r :bool)``: term
```

Chapter 3

Exercise 7.3.2

3.1 Problem Statement

1. Create a function impImpAnd term that operates on terms of the form $p \Rightarrow q \Rightarrow r$ and returns $p \wedge q \Rightarrow r$. Show that impImpAnd reverses the effects of andImp2Imp, and vice verse.

3.2 Relevant Code

```
fun impImpAnd exp1 =
let
        val (im1,im2) = dest_imp exp1
        val (im3,im4) = dest_imp im2
        val im5 = mk_conj (im1,im3)
in
        mk_imp (im5,im4)
end;
```

3.3 Test Cases

```
val test2 = ``p ==> q ==> r``;
andImp2Imp(impImpAnd test2);
```

3.4 Execution Transcripts

```
> andImp2Imp(impImpAnd test2);
val it = ``(p :bool) ==> (q :bool) ==> (r :bool)``: term
```

Chapter 4

Exercise 7.3.3

4.1 Problem Statement

1. Create a function notExists term that operates on terms of the form $\neg\exists x.P(x)$ and returns $\forall x.\neg P(x)$.

4.2 Relevant Code

```
fun notExists exp1 =
let
        val (im1,im2) = dest_comb exp1
        val (im3,im4) = dest_exists im2
        val im5 = mk_neg im4
in
        mk_forall (im3,im5)
end;
```

4.3 Test Cases

```
val test3 = ``~?z.Q z``;
notExists test3;
```

4.4 Execution Transcripts

```
> notExists test3;
val it = ``!(z :'a). ~(Q :'a -> bool) z``: term
```

Appendix A

Source Code for All Answers

```
(*******************************************************)
(* Chapter 7 Answers *)
(* Author: Michael Hrishenko *)
(* Date: 31JAN2020 *)
(*======================= Ex 7.3.1 ========================*)
fun andImp2Imp exp1 =
let
        val (im1,im2) = dest_imp exp1
        val (im3,im4) = dest_conj im1
        val im5 = mk_imp (im4,im2)
in
        mk_imp (im3,im5)
end;

val test1 = ``(p /\ q) ==> r``;
andImp2Imp test1;
(*======================= Ex 7.3.2 ========================*)
fun impImpAnd exp1 =
let
        val (im1,im2) = dest_imp exp1
        val (im3,im4) = dest_imp im2
        val im5 = mk_conj (im1,im3)
in
        mk_imp (im5,im4)
end;

val test2 = ``p ==> q ==> r``;
andImp2Imp(impImpAnd test2);
(*======================= Ex 7.3.3 ========================*)
fun notExists exp1 =
let
        val (im1,im2) = dest_comb exp1
        val (im3,im4) = dest_exists im2
        val im5 = mk_neg im4
in
        mk_forall (im3,im5)
end;

val test3 = ``~?z.Q z``;
notExists test3;
```