Project 4

Bharath Karumudi

August 5, 2019

Abstract

This project is to demonstrate the capabilities of implementing constructing and deconstructing HOL Terms using the tools and techniques - LaTeX, AcuTeX, emacs and ML.

Each chapter documents the given problems with a structure of:

1. Problem Statement

2. Relevant Code

3. Execution Transcripts

4. Explanation of results

Acknowledgments: Professor Marvine Hamner and Professor Shiu-Kai Chin who taught the Certified Security By Design.

Contents

Chapter 1

Executive Summary

Some requirements for this project are statisfied specifically, and by using HOL proved the below theorems:

[`absorptionRule`]

$\vdash \forall p\ q.\ (p \Rightarrow q) \Rightarrow p \Rightarrow p \wedge q$

[`absorptionRule2`]

$\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

[`constructiveDilemmaRule`]

$\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

[`constructiveDilemmaRule2`]

$\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

[`problemonethm`]

$\vdash M\ s$

[`problemtwothm`]

$\vdash p \Rightarrow \neg q$

10.4.3 is not included.

Chapter 2

Exercise 9.5.1

2.1 Problem Statement

In this exercise we need to prove the theorem: $\vdash \forall p\ q.\ (p \Rightarrow q) \Rightarrow p \Rightarrow p \wedge q$

2.2 Relevant Code

```
val absorptionRule =
TAC_PROOF (
([],``!p q. (p ==> q) ==> p ==> p/\q``),
(REPEAT STRIP_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC) );

val _ = save_thm("absorptionRule",absorptionRule);
```

2.3 Execution Transcripts

1

```
---------------------------------------------------------------------
       HOL-4 [Kananaskis 11 (stdknl, built Sat Aug 19 09:30:06 2017)]

       For introductory HOL help, type: help "hol";
       To exit type <Control>-D
---------------------------------------------------------------------
> > > > # # # # # # # # # ** types trace now on
> *** Globals.show_assums now true ***
> # # # # # # # # # ** Unicode trace now off
>
> # # # # # val absorptionRule =
    [] |- !(p :bool) (q :bool). (p ==> q) ==> p ==> p /\ q:
   thm
>
```

2.3.1 Explanation of Results

The above results shows that the requirements are satisfied.

Chapter 3

Exercise 9.5.2

3.1 Problem Statement

In this exercise we need to prove the theorem: $\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

3.2 Relevant Code

```
val constructiveDilemmaRule =
 TAC_PROOF (
([], ``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
REPEAT STRIP_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC THEN
ASM_REWRITE_TAC []
);

val _ = save_thm("constructiveDilemmaRule",constructiveDilemmaRule);
```

3.3 Execution Transcripts

1

```
---------------------------------------------------------------------
       HOL-4 [Kananaskis 11 (stdknl, built Sat Aug 19 09:30:06 2017)]

       For introductory HOL help, type: help "hol";
       To exit type <Control>-D
---------------------------------------------------------------------
> > > > # # # # # # # # # ** types trace now on
> *** Globals.show_assums now true ***
> # # # # # # # # # ** Unicode trace now off
>
> # # # # # # # # # val constructiveDilemmaRule =
    []
|- !(p :bool) (q :bool) (r :bool) (s :bool).
     (p ==> q) /\ (r ==> s) ==> p \/ r ==> q \/ s:
   thm
>
>
```

3.3.1 Explanation of Results

The above results shows that the requirements are satisfied.

Chapter 4

Exercise 9.5.3

4.1 Problem Statement

In this exercise we need to prove the theorem:

$\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

$\vdash \forall p\ q\ r\ s.\ (p \Rightarrow q) \wedge (r \Rightarrow s) \Rightarrow p \vee r \Rightarrow q \vee s$

4.2 Relevant Code

```
val absorptionRule2 =
 TAC_PROOF (
([], ``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
PROVE_TAC []
);

val _= save_thm("absorptionRule2",absorptionRule2);

val constructiveDilemmaRule2=
 TAC_PROOF (
([],``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
PROVE_TAC []
);

val _ = save_thm("constructiveDilemmaRule2",constructiveDilemmaRule2);
```

4.3 Execution Transcripts

```
---------------------------------------------------------------------
       HOL-4 [Kananaskis 11 (stdknl, built Sat Aug 19 09:30:06 2017)]

       For introductory HOL help, type: help "hol";
       To exit type <Control>-D
---------------------------------------------------------------------
> > > > # # # # # # # # # # ** types trace now on
> *** Globals.show_assums now true ***
> # # # # # # # # # # ** Unicode trace now off
>
> # # # # Meson search level: ................
val absorptionRule2 =
    []
|- !(p :bool) (q :bool) (r :bool) (s :bool).
     (p ==> q) /\ (r ==> s) ==> p \/ r ==> q \/ s:
   thm
>
> # # # # Meson search level: ................
val constructiveDilemmaRule2 =
    []
|- !(p :bool) (q :bool) (r :bool) (s :bool).
     (p ==> q) /\ (r ==> s) ==> p \/ r ==> q \/ s:
   thm
>
>
```
1

4.3.1 Explanation of Results

The above results shows that the requirements are satisfied.

Chapter 5

Exercise 10.4.1

5.1 Problem Statement

In this exercise we need to prove the theorem: $\vdash M\ s$

5.2 Relevant Code

```
val problemonethm=
TAC_PROOF(
([ `` !x: 'a.P(x) ==> M(x) ``, ``(P: 'a->bool) (s: 'a)``],
``(M:'a->bool) (s: 'a)``),
RES_TAC
);

val _=save_thm("problemonethm",problemonethm);
```

5.3 Execution Transcripts

```
---------------------------------------------------------------------
       HOL-4 [Kananaskis 11 (stdknl, built Sat Aug 19 09:30:06 2017)]

       For introductory HOL help, type: help "hol";
       To exit type <Control>-D
---------------------------------------------------------------------
> > > > # # # # # # # # # ** types trace now on
> *** Globals.show_assums now true ***
> # # # # # # # # # ** Unicode trace now off
>
> # # # # # val problemonethm =

[(P :'a -> bool) (s :'a),
 !(x :'a). (P :'a -> bool) x ==> (M :'a -> bool) x]
|- (M :'a -> bool) (s :'a):
   thm
> >
>
```
1

5.3.1 Explanation of Results

The above results shows that the requirements are satisfied.

Chapter 6

Exercise 10.4.2

6.1 Problem Statement

In this exercise we need to prove the theorem: $\vdash p \Rightarrow \neg q$

6.2 Relevant Code

```
val  problemtwothm=
TAC_PROOF(
([``p /\ q ==> r``, ``r ==> s``, ``~s``], ``p ==> ~q``),
(PAT_ASSUM ``r  ==>s``
           (fn th =>
           ASSUME_TAC
             (DISJ_IMP (ONCE_REWRITE_RULE [DISJ_SYM] (IMP_ELIM th) )
             )
           )
) THEN
(PAT_ASSUM ``p /\ q ==> r``
           (fn th2 =>
           ASSUME_TAC
           (DISJ_IMP (ONCE_REWRITE_RULE [DISJ_SYM] (IMP_ELIM th2))))) THEN
REPEAT STRIP_TAC THEN
RES_TAC
)
val _=save_thm("problemtwothm",problemtwothm);
```

6.3 Execution Transcripts

1

```
---------------------------------------------------------------------
       HOL-4 [Kananaskis 11 (stdknl, built Sat Aug 19 09:30:06 2017)]

       For introductory HOL help, type: help "hol";
       To exit type <Control>-D
---------------------------------------------------------------------
> > > > # # # # # # # # # ** types trace now on
> *** Globals.show_assums now true ***
> # # # # # # # # # ** Unicode trace now off
>
> # # # # # # # # # # # # # # # # val problemtwothm =

[~(s :bool), (r :bool) ==> (s :bool),
 (p :bool) /\ (q :bool) ==> (r :bool)] |- (p :bool) ==> ~(q :bool):
   thm
>
*** Emacs/HOL command completed ***

>
>
```

6.3.1 Explanation of Results

The above results shows that the requirements are satisfied.

Chapter 7

Appendix A: Chapter 9

The following code is from the file project3bScript.sml

```
(*************************************************************************)
(*   Exercise: Chapter 9                                                 *)
(*   Author: Bharath Karumudi                                            *)
(*   Date: Aug 2, 2019                                                   *)
(*************************************************************************)

structure exercise9Script = struct
open HolKernel Parse boolLib bossLib;

val _ = new_theory "exercise9";

(*************************************************************************)
(*   Exercise: 9.5.1                                                     *)
(*   `       p q. (p       q)      p      p      q
*)
(*************************************************************************)

val absorptionRule =
TAC_PROOF (
([],``!p q. (p ==> q) ==> p ==> p/\q``),
(REPEAT STRIP_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC) );

val _ = save_thm("absorptionRule",absorptionRule);


(*************************************************************************)
(*  Exercise: 9.5.2                                                      *)
(*  `       p q r s. (p       q)       (r       s)      p      r      q      s
*)
(*************************************************************************)

val constructiveDilemmaRule =
 TAC_PROOF (
([],  ``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
REPEAT STRIP_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC THEN
ASM_REWRITE_TAC [] THEN
RES_TAC THEN
ASM_REWRITE_TAC []
```

```
);

val _ = save_thm("constructiveDilemmaRule",constructiveDilemmaRule);


(***********************************************************************)
(* Exercise: 9.5.3                                                     *)
(* Repeat 9.5.1, 9.5.2 using PROVE_TAC                                 *)
(***********************************************************************)

val absorptionRule2 =
 TAC_PROOF (
([], ``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
PROVE_TAC []
);

val _= save_thm("absorptionRule2",absorptionRule2);

val constructiveDilemmaRule2=
 TAC_PROOF (
([],``!p q r s.(p ==> q) /\ (r ==> s) ==> (p\/r) ==> (q\/s)``),
PROVE_TAC []
);

val _ = save_thm("constructiveDilemmaRule2",constructiveDilemmaRule2);


(***********************************************************************)
(* Exporting Theory                                                    *)
(***********************************************************************)

val _ = export_theory();

end (* Structure *)
```

Chapter 8

Appendix B: Chapter 10

The following code is from the file project3bScript.sml

```
(***********************************************************************)
(*   Exercise: Chapter 10                                              *)
(*   Author: Bharath Karumudi                                          *)
(*   Date: Aug 2, 2019                                                 *)
(***********************************************************************)

structure exercise10Script = struct
open HolKernel Parse boolLib bossLib;

val _ = new_theory "exercise10";


(***********************************************************************)
(*   Exercise: 10.4.1                                                  *)
(*   set_goal                                                          *)
(*       ([``!x:`a.P(x) ==> M(x)``,``(P:`a->bool)(s:`a)``],            *)
(*   ``(M:   a ->bool)(s:   a )        );
*)
(***********************************************************************)

val problemonethm=
TAC_PROOF(
([  `` !x: 'a.P(x) ==> M(x)  ``,  ``(P: 'a->bool) (s: 'a)``],
``(M:'a->bool) (s: 'a)``),
RES_TAC
);

val _=save_thm("problemonethm",problemonethm);


(***********************************************************************)
(* Exercise: 10.4.2                                                    *)
(*     set_goal([``p /\  q ==> r``,``r ==> s``,``   s ``],``p ==>     q ``)
*)
(***********************************************************************)

val  problemtwothm=
TAC_PROOF(
([``p /\ q ==> r``,  ``r ==> s``,  ``~s``],  ``p ==> ~q``),
(PAT_ASSUM  ``r   ==>s``
                (fn  th =>
                ASSUME_TAC
```

```
                    (DISJ_IMP (ONCE_REWRITE_RULE [DISJ_SYM] (IMP_ELIM th) )
                    )
                 )
) THEN
(PAT_ASSUM ``p /\ q ==> r``
               (fn th2 =>
               ASSUME_TAC
               (DISJ_IMP (ONCE_REWRITE_RULE [DISJ_SYM] (IMP_ELIM th2))))) THEN
REPEAT STRIP_TAC THEN
RES_TAC
)
val _=save_thm("problemtwothm",problemtwothm);



(******************************************************************************)
(* Exercise: 10.4.3                                                           *)
(*    set_goal([``   (p /\ q)``,  ``   p ==> r``,``    q ==> s``],``r \/ s  ``);
*)
(******************************************************************************)




(******************************************************************************)
(* Exporting Theory                                                           *)
(******************************************************************************)

val _ = export_theory();

end (* Structure *)
```