Report for ML Exercises

Wei-Ling Wang

January 28, 2020

Abstract

This article is mainly about the HOL excercise. By learning the video given by the teacher, I have mastered the basic use of functions of ML and the expression of variables, including int, list and string, and learned Logic Semantics. At the same time, I used emacs to compile the report and write the source code, which made me more familiar with the syntax of LaTaX.

- Problem statement
- Relevant code
- Results from running ML code or HOL proofs

For each problem or exercise-oriented chapter in the main body of the report there is a corresponding chapter in the Appendix containing the source code in ML. This source code is not pasted into the Appendix, rather it is input directly from the source code file itself. This means changes in source code are easily captured in the report by recompiling the report in LaTeX.

We introduce the use of style files and packages. Specifically, we use:

- A style file for the course, *634format.sty*
- The *listings* package for displaying and inputting ML source code
- HOL style files and commands to display interactive ML/HOL sessions

Finally, we show how to:

- Easily generate a table of contents for the report
- Refer to chapter and section labels in our report

Acknowledgments: I did this project by myself. I tried to write the HOL program for the first time by studying the video given by the teacher.

Contents

Chapter 1

Executive Summary

All requirements for this project are satisfied. Specifically,

Report Contents
Our report has the following content:

- Chapter 1: Executive Summary
- Chapter 2: Exercise 2.5.1
  - Section 2.1: Problem Statement
  - Section 2.1.2: Test Cases
  - Section 2.2: Relevant Code
  - Section 2.3: Execution Transcripts
- Chapter 3: Source Code for Sample Exercise
  - Section 3.1: Problem Statement
  - Section 3.2: Relevant Code
  - Section 3.3: Execution Transcripts
- Chapter 4: Exercise 3.4.2
  - Section 4.1: Problem Statement
  - Section 4.2: Relevant Code
  - Section 4.3: Execution Transcripts
- Appendix A: Source Code for Exercise 2.5.1
- Appendix B: Source Code for Exercise 3.4.1
- Appendix C: Source Code for Exercise 3.4.2

Reproducibility in ML and LaTeX
The ML and LaTeX source files compile with no errors.

Chapter 2

Exercise 2.5.1

2.1 Problem Statement

2.1.1 Function to implement

We are required to execute the following function in ML:

$$timesPlus\ x\ y\ =\ (\ x \times y\ ,\ x + y\ )$$

2.1.2 Test cases

The required tests for *timesPlus* are as follows:

```
(* Test Cases specified in the requirements *)

timesPlus 100   27;
timesPlus  10   26;
timesPlus   1   25;
timesPlus   2   24;
timesPlus  30   23;
timesPlus  50  200;
```

2.2 Relevant Code

The following code runs the function definition using *fun* in ML, and *currying*, i.e., defining functions with multiple arguments as a sequence of functions. This supports partial evaluation.

```
fun timesPlus x y = ( x * y , x + y )
```

2.3 Execution Transcripts

```
> timesPlus 100 27;
val it = (2700, 127): int * int
```
1

```
> timesPlus 10 26;
val it = (260, 36): int * int
```
2

```
> timesPlus 1 25;
val it = (25, 26): int * int
```
3

```
> timesPlus 2 24;
val it = (48, 26): int * int
```
4

```
> timesPlus 30 23;
val it = (690, 53): int * int
```
5

```
> timesPlus 50 200;
val it = (10000, 250): int * int
```
6

Chapter 3

Exercise 3.4.1

3.1 Problem Statement

We are to devise ML expressions for the following tasks where we have values and require to assign them to variables as specified:

1. Devise a list of pairs [(0,"Alice"), (1,"Bob"), (3,"Carol"), (4,"Dan")] and assign it the name listA
2. Using listA and pattern matching, create the following value assignments: elB has the value (0,"Alice") and listB has the value [(1,"Bob"), (3,"Carol"), (4,"Dan")]
3. Using elB, listB and pattern matching, create the following value assignments: elC1 has the value 0, elC2 has the value "Alice", elC3 has the value (1,"Bob"), elC4 has the value (3,"Carol"), and elC5 has the value (4,"Dan").

3.2 Relevant Code

The following code uses value declarations and pattern matching on tuples and lists.

```
val listA = [(0,"Alice"),(1,"Bob"),(3,"Carol"),(4,"Dan")];
val (elB :: listB) = listA;
val (elC1, elC2) = elB;
val (elC3 :: (elC4 :: (elC5 :: []))) = listB;
```

3.3 Execution Transcripts

1

```
> val listA = [(0,"Alice"), (1,"Bob"), (3,"Carol"), (4,"Dan")];
val listA = [(0, "Alice"), (1, "Bob"), (3, "Carol"), (4, "Dan")] : (int * string) list
```

2

```
> val (elB :: listB) = listA;
val elB = (0, "Alice"): int * string
val listB = [(1, "Bob"), (3, "Carol"), (4, "Dan")] : (int * string) list
```

3

```
> val (elC1, elC2) = elB;
val elC1 = 0: int
val elC2 = "Alice": string
> val (elC3 :: (elC4 :: (elC5 :: []))) = listB;
val elC3 = (1, "Bob"): int * string
val elC4 = (3, "Carol"): int * string
val elC5 = (4, "Dan"): int * string
```

Chapter 4

Exercise 3.4.2

4.1 Problem Statement

We are to evaluate the following assignments in the order in which they appear in HOL, explain the errors that HOL detects using comments and store the results in ex-3-4-2.trans file:

1. val (x1, x2, x3) = (1, true, "Alice");
2. val pair1 = (x1, x3);
3. val list1 = [0, x1, 2];
4. val list2 = [x2, x1];
5. val list3 = (1 :: [x3]);

4.2 Relevant Code

The following code uses value declarations and pattern matching on tuples. Lists are given as part of the problem statement

```
val (x1, x2, x3) = (1, true, "Alice");
val pair1 = (x1, x3);
val list1 = [0, x1, 2];
val list2 = [x2, x1];
val list3 = (1 :: [x3]);
```

4.3 Execution Transcripts

1
```
> val (x1, x2, x3) = (1, true, "Alice");
val x1 = 1: int
val x2 = true: bool
val x3 = "Alice": string
```

2
```
> val pair1 = (x1, x3);
val pair1 = (1, "Alice"): int * string
```

3
```
> val list1 = [0, x1, 2];
val list1 = [0, 1, 2]: int list
```

4
```
> val list2 = [x2, x1];
poly: : error: Elements in a list have different types.
   Item 1: x2 : bool
   Item 2: x1 : int
   Reason:
      Can't unify bool (*In Basis*) with int (*In Basis*)
         (Different type constructors)
Found near [x2, x1]
Static Errors
```

```
> val list3 = (1 :: [x3]);                                                    5
poly: : error: Type error in function application.
   Function: :: : int * int list -> int list
   Argument: (1, [x3]) : int * string list
   Reason:
      Can't unify int (*In Basis*) with string (*In Basis*)
         (Different type constructors)
Found near (1 :: [x3])
Static Errors
```

Appendix A

Source Code for Exercise 2.5.1

The following code is from *ex-2-5-1.sml*

```
(*****************************************************************************)
(* Exercise : 2.5.1*)
(* Name : Wei-ling Wang *)
(* Email: wwang118@syr.edu *)
(*****************************************************************************)

fun timesPlus x y = (x*y, x+y);

(* Test Cases: *)

timesPlus 100  27;
timesPlus  10  26;
timesPlus   1  25;
timesPlus   2  24;
timesPlus  30  23;
timesPlus  50 200;
```

Appendix B

Source Code for Exercise 3.4.1

The following code is from *ex-3-4-1.sml*

```
(*****************************************************************************)
(*Exercise : 3.4.1*)
(* Name : Wei-ling Wang *)
(* Email: wwang118@syr.edu *)
(*****************************************************************************)

val listA = [(0,"Alice"), (1,"Bob"), (3,"Carol"), (4,"Dan")];
val (elB :: listB) = listA;
val (elC1, elC2) = elB;
val (elC3 :: (elC4 :: (elC5 :: []))) = listB;
```

Appendix C

Source Code for Exercise 3.4.2

The following code is from *ex-3-4-2.sml*

```
(*************************************************************************)
(*Exercise : 3.4.2                                                       *)
(* Name : Wei-ling Wang *)
(* Email: wwang118@syr.edu *)
(*************************************************************************)

val (x1, x2, x3) = (1, true, "Alice");
(*No error. x1=int (1), x2=bool (true) and x3=string ("Alice")          *)

val pair1 = (x1, x3);
(*No error. pair1 is assigned int*string (1, "Alice")                   *)

val list1 = [0, x1, 2];
(*No error. list1 = integer list consisting  of [0, 1, 2]              *)

val list2 = [x2, x1];
(*Static Error. Error arises due to assigning elements of different data types*)
(*bool and int in the same list. List have elements only of the same data type*)

val list3 = (1 :: [x3]);
(*Static Error. Error arises due to assigning elements of different data types*)
(*int and string list in the same list. List have elements of same data type  *)
```