CS113 Lab:1

Saw Thinkar Nay Htoo

February 12, 2016

Purpose

To explore the tools to create documents with math symbols combined with programming and to learn about logic connectives used in propositional calculus. [1]

- The truth table value of $p \wedge q$.(Conjunction)

Table 1: Truth Table

p	q	$(p \wedge q)$
F	F	F
F	T	F
T	F	F
T	T	T

[1] lab1 second draft

- The truth table value of $p \vee q$.(Disjunction)

Table 2: Truth Table

p	q	$(p \vee q)$
F	F	F
F	T	T
T	F	T
T	T	T

- The truth table value of $p \oplus q$.(Exclusive or)[2]

Table 3: Truth Table

p	q	$(p \oplus q)$
F	F	F
F	T	T
T	F	T
T	T	F

[2]Unnecessary ”r” is removed from the truth table.

- The truth table value of $\sim p$ and $\sim q$.(Negation)[3]

Table 4: Truth Table

p	$\sim p$
F	T
T	F

Table 5: Truth Table

q	$\sim q$
F	T
T	F

[3]Negation signs are changed from $\neg$ to $\sim$.

SML codes for conjuntion, disjunction, exclusive or and negation

SML code for conjunction. When the two values are True, the result is True. Otherwise the rest of the results are False.[4]

SML

```
datatype Bool = F | T;
fun conjunction(F,x) = F
 | conjunction(T,x) = x;
conjunction(F,F);
conjunction(T,F);
conjunction(F,T);
conjunction(T,T);
val truth_values = [(F,F),(F,T),(T,F),(T,T)];
map conjunction truth_values;
```

```
> val truth_values = [(F,F),(F,T),(T,F),(T,T)];
val truth_values = [(F, F), (F, T), (T, F), (T, T)]: (Bool * Bool) list
> map conjunction truth_values;
val it = [F, F, F, T]: Bool list
```

[4]Output results for SML codes are added and function names are changed to long form to reduce confusion.

SML code for disjunction. When the two values are False, the output is False. The rest of the results are True.

SML

```
datatype Bool = F | T;
fun disjunction(F,F) = F
| disjunction(T,x) = T
| disjunction(x,T) = T;
disjunction(F,F);
disjunction(T,F);
disjunction(F,T);
disjunction(T,T);
val truth_values = [(F,F),(F,T),(T,F),(T,T)];
map disjunction truth_values;
```

```
> val truth_values = [(F,F),(F,T),(T,F),(T,T)];
val truth_values = [(F, F), (F, T), (T, F), (T, T)]: (Bool * Bool) list
> map disjunction truth_values;
val it = [F, T, T, T]: Bool list
```

SML code for exclusive or. When the values are the same, the output is False. When the values are different, the output is True.

SML

```
datatype Bool = F | T;
fun exclusive_or (T,T) = F
 | exclusive_or (F,F) = F
 | exclusive_or (x,y) = T;
exclusive_or(F,F);
exclusive_or(T,F);
exclusive_or(F,T);
exclusive_or(T,T);
val truth_values = [(F,F),(F,T),(T,F),(T,T)];
map exclusive_or truth_values;
```

```
> val truth_values = [(F,F),(F,T),(T,F),(T,T)];
val truth_values = [(F, F), (F, T), (T, F), (T, T)]: (Bool * Bool) list
> map exclusive_or truth_values;
val it = [F, T, T, F]: Bool list
```

SML code for negation. The values are reversed.

SML

```
datatype Bool = F | T;
fun negation T = F
  | negation F = T;
negation (T);
negation (F);
val truth_values = [F,T];
map negation truth_values;
```

```
> val truth_values = [F,T];
val truth_values = [F, T]: Bool list
> map negation truth_values;
val it = [T, F]: Bool list
```

Problem 1.18

Show that $p \oplus q \equiv (p \vee q) \wedge \sim (p \wedge q)$. [5]

1. Left hand side

Table 6: Truth Table

p	q	$(p \oplus q)$
F	F	F
F	T	T
T	F	T
T	T	F

2. Right hand side

Table 7: Truth Table

p	q	$(p \vee q)$	$(p \wedge q)$	$\sim (p \wedge q)$	$((p \vee q) \wedge \sim (p \wedge q))$
F	F	F	F	T	F
F	T	T	F	T	T
T	F	T	F	T	T
T	T	T	T	F	F

3. LHS = RHS (Hence shown)

[5] Negation signs are changed from $\neg$ to $\sim$.

This is the result created by inputting the entire equation into the truth table generator.[6]

Table 8: Truth Table

p	q	$(p \oplus q)$	$(p \vee q)$	$(p \wedge q)$	$\sim (p \wedge q)$	$((p \vee q) \wedge \sim (p \wedge q))$	$((p \oplus q) \equiv ((p \vee q) \wedge \sim (p \wedge q)))$
F	F	F	F	F	T	F	T
F	T	T	T	F	T	T	T
T	F	T	T	F	T	T	T
T	T	F	T	T	F	F	T

section*SML code for 1.18[7]

SML

```
datatype Bool = F | T;
fun negation T = F
 | negation F = T;
fun exclusive_or (T,T) = F
 | exclusive_or (F,F) = F
 | exclusive_or (x,y) = T;
fun disjunction(F,F) = F
 | disjunction(T,x) = T
 | disjunction(x,T) = T;
fun conjunction(F,x) = F
 | conjunction(T,x) = x;
```

[6]Negation signs are changed from ¬ to ∼. Table is created again with details.

[7]Function names are changed to long form to help readers not to be confused.

conjunction(p,q) is created first since it is most inner function. Then its negation and disjunction(p,q) are built simultaneously. Then both of their results are used for outer most function(conjunction) of the left hand side.[8]
For left hand side, "exclusive or" function is operated.

SML

```
conjunction(disjunction(T,F),negation(conjunction(T,F)));
exclusive_or(T,F);
```

Since the result of the both functions are the same as True's, it is concluded that left hand side is equal to right hand side. (Below figure)

```
> conjunction(disjunction(T,F),negation(conjunction(T,F)));
val it = T: Bool
> exclusive_or(T,F);
val it = T: Bool
```

[8]Explanation added.

Sample Code and its result by the professor.

SML

```
datatype Bool = F | T;
fun conjunction (F,F) = F
  | conjunction (F,T) = F
  | conjunction (T,F) = F
  | conjunction (T,T) = T;
val truth_values = [(F,F),(F,T),(T,F),(T,T)];
map conjunction truth_values;

```

Running above SML code produces this truth table:[9]

```
Poly/ML 5.5.2 Release
datatype Bool = F | T
val conjunction = fn: Bool * Bool -> Bool
val it = F: Bool
val it = F: Bool
val it = F: Bool
val it = T: Bool
val conjunction = fn: Bool * Bool -> Bool
val truth_values = [(F, F), (F, T), (T, F), (T, T)]: (Bool * Bool) list
val it = [F, F, F, T]: Bool list
infix 0 AND
val AND = fn: Bool * Bool -> Bool
val it = F: Bool
debian@debian:~/labs/lab2$ import out1.png
```

[9]Explanation added.

SML

```
infix AND;
fun p AND q = conjunction (p,q);
T AND F;
```

```
> infix AND;
infix 0 AND
> fun p AND q = conjunction (p,q);
val AND = fn: Bool * Bool -> Bool
> T AND F;
val it = F: Bool
```

A new function is created using infix AND, thus making the code less confusing and easier to understand. This method is supposed to be used in problem 1.18

Review Questions

- Problem 1.9
  Question: Use De Morgan's laws to write the negation for the proposition: "The dollar is at an all-time high and the stock market is at a record low."
  Answer: The dollar is **not** at an all-time high **or** the stock market is **not** at a record low.
  Note: This is not the result of the negation: "The dollar is at an all-time **low or** the stock is at a record **high**."

- Problem 1.10
  Question: Assume $x \in \mathbb{R}$. Use De Morgan's laws to write the negation for the proposition: $-5 < x <= 0.$[10]
  Answer: $x \leq -5$ or $x > 0$
  Note: Less than or equal sign is changed to the other side of the equation. Definition of DeMorgan's Law: "The negation of a conjunction is the disjunction of the negations. The negation of a disjunction is the conjunction of the negations." Ref: wikipedia. In this case, as we can see less-than-or-equal signs are changed on both side and "or" is added in between two ranges of values of possible value of x.

---

[10] Changed to correct symbols by adding delimiters. Explanation for DeMorgan's law added.

- Problem 1.12 Question: Show that the proposition $s = (p \wedge \sim q) \wedge (\sim p \vee q)$ is a contradiction. [11]
  Answer: **Contradiction**: A compound proposition that has the value F for all possible values of the propositions in it.

Table 9: Truth Table

p	q	$((p \wedge \sim q) \wedge (\sim p \vee q))$
F	F	F
F	T	F
T	F	F
T	T	F

[11] Explanation added.

- Problem 1.13 (c) Question: Is $(p \oplus q) \wedge r \equiv (p \wedge r) \oplus (q \wedge r)$? Justify your answer.
  Answer: As we can see the results of the fifth and eighth columns of the table are the same, it is concluded that left hand side is equal to right hand side.

Table 10: Truth Table

p	q	r	$(p \oplus q)$	$((p \oplus q) \wedge r)$	$(p \wedge r)$	$(q \wedge r)$	$((p \wedge r) \oplus (q \wedge r))$	$(((p \oplus q) \wedge r) \equiv ((p \wedge r) \oplus (q \wedge r)))$
F	F	F	F	F	F	F	F	T
F	F	T	F	F	F	F	F	T
F	T	F	T	F	F	F	F	T
F	T	T	T	T	F	T	T	T
T	F	F	T	F	F	F	F	T
T	F	T	T	T	T	F	T	T
T	T	F	F	F	F	F	F	T
T	T	T	F	F	T	T	F	T

- Problem 1.14 (c) Question: $\sim t \equiv$ c and $\sim c \equiv t$, where t is a tautology and c is a contradiction.
  Answer: [12]
  Tautology: A compound prosition which is always true, regardless of the truth values of the propositional variables which comprise it.
  Contradiction: A compound proposition that has the value F for all possible values of the propositions in it.

Table 11: Truth Table

t	$\sim t$
T	F
T	F

Table 12: Truth Table

c	$\sim c$
F	T
F	T

---

[12] Definitions added. Tables edited.

- Problem 1.14 (d) Question: $p \vee p \equiv p$ and $p \wedge p \equiv p$
  Answer:

Table 13: Truth Table

p	$(p \vee p)$	$((p \vee p) \equiv p)$
F	F	T
T	T	T

Table 14: Truth Table

p	$(p \wedge p)$	$((p \wedge p) \equiv p)$
F	F	T
T	T	T

- Problem 1.20 Question: Show that $\sim p \wedge (p \wedge q)$ is a contradiction.
  Answer:

Table 15: Truth Table

p	q	$\sim p$	$(p \wedge q)$	$(\sim p \wedge (p \wedge q))$
F	F	T	F	F
F	T	T	F	F
T	F	F	F	F
T	T	F	T	F