CPSC 121 Midterm 1
Monday February 6th, 2017

[8] 1. **Logical Equivalence Proof**

Using a sequence of logical equivalence rules from "Dave's Awesome Handout", simplify:

$((q \vee (p \ \wedge p)) \wedge (q \rightarrow p)) \vee \sim(p \rightarrow (r \vee \sim s))$

Please write the name of the law(s) you applied at each step. *Hint:* your final simplified form should be **very** simple.

Solution :

LHS	$\equiv ((q \vee p) \wedge (q \rightarrow p)) \vee \sim(p \rightarrow (r \vee \sim s))$	ID
	$\equiv ((p \vee q) \wedge (q \rightarrow p)) \vee \sim(\sim p \vee r \vee \sim s)$	IMP
	$\equiv ((p \vee q) \wedge (q \rightarrow p)) \vee (p \wedge \sim r \wedge s)$	DM
	$\equiv ((p \vee q) \wedge (p \vee \sim q)) \vee (p \wedge \sim r \wedge s)$	IMP
	$\equiv p \vee (q \wedge \sim q) \vee (p \wedge \sim r \wedge s)$	DIST
	$\equiv p \vee (q \wedge \sim q)$	ABS
	$\equiv p \vee F$	NEG
	$\equiv p$	I

[8] 2. **Tautologies, Contingencies, and Contradictions:**

Determine whether the following statements are **tautologies** (true for all assignments of truth values to their variables, i.e., logically equivalent to T), **contradictions** (false for all assignments of truth values to their variables, i.e., logically equivalent to F), or contingencies (true or false depending on the values of their variables). Indicate your answer by circling *one* of TAUTOLOGY, CONTRADICTION, and CONTINGENCY for each. You do **not** need to justify your answer.

[2] a. d

Solution : CONTINGENCY: true when $d = T$ and false when $d = F$.

[2] b. $(r \wedge s) \rightarrow r$

Solution : TAUTOLOGY: When $r = T$, then the right-hand-side of implication is true; so, the implication itself is true. When $r = F$, $r \wedge s$ is also false, and so the left-hand-side of the implication is false; so, the implication itself is again true.

[2] c. $(p \oplus q) \leftrightarrow (q \vee p)$

Solution: CONTINGENCY: It's tempting to say "these two don't mean the same thing so the biconditional is false". However, go back to truth values. When $p = T$ and $q = F$, then both sides of the biconditional are true and the biconditional itself is true. When p and q are both true, the left side is false but the right is true, and so the biconditional is false.

[2] d. $(p \vee q) \wedge (p \vee \sim q) \wedge (\sim p \vee q) \wedge (\sim p \vee \sim q)$

Solution: CONTRADICTION: Let's simplify. The left half of the expression simplifies as follows: $(p \vee q) \wedge (p \vee \sim q) \equiv p \vee (q \wedge \sim q) \equiv p \vee F \equiv p$. But, the right half of the expression simplifies somewhat similarly: $(\sim p \vee q) \wedge (\sim p \vee \sim q) \equiv \sim p \vee (q \wedge \sim q) \equiv \sim p \vee F \equiv \sim p$. We're left with $p \wedge \sim p \equiv F$.

[23] 3. **Number Representation**

THIS TEXT EXACTLY REPEATS THE PRE-READING YOU WERE GIVEN, UP TO "END OF PRE-READING" BELOW.

In our discussion of number representation, we usually discuss **fixed-width** representations. That is, a number might be represented by 8 bits or 32 bits, but not by "however many bits it takes".

However, when people write out numbers, we usually use a variable-width representation. For instance, the grade you get for this course might be a 3-digit number (100), a 2-digit number (10–99), or (hopefully not) a 1-digit number (0–9). If grades were "fixed-width", then, we'd have to write a grade like 83 as 083 instead. An apparent advantage of our variable-width representation is that we can write numbers as large as we like just by taking up more space.

For this problem, we'll consider an approach that uses bits to achieve this sort of variable-width representation of unsigned binary values. We call our proposed representation "flag bit" numbers: a number is composed of one or more 2-bit "blocks". The first bit in each block is a normal base-2 digit. The second bit is the "flag bit". It is 0 if this is the last block in the number and 1 if there is another block. To determine the value of the variable-width number, we collect up all the normal digits, **reverse them**, and interpret the result as an unsigned binary number.

For clarity, we'll write "flag bit" numbers with commas separating the blocks, although those wouldn't be included in the stored representation of the numbers.

For example, the numbers 0 and 1 would be 00 and 10 (respectively) as "flag bit" numbers.

The numbers 2 and 3 would be 01,10 and 11,10 (respectively).

The number 01,11,01,11,10 is the decimal number 26 because: the first four blocks' flag bits indicate that the number continues up to the fifth block's flag bit, which indicates that the number ends there; the binary digits in those five blocks, read from left to right, are 01011; we reverse these bits to get 11010; and, that unsigned binary number is the decimal number 26.

END OF PRE-READING

[2] a. Translate the "flag bit" number 01,11,10 to a decimal number.

Solution : The flag bits indicate that this is a single 3-block number. The binary digits are 011. Reversing them we get 110. That's the decimal number 6.

[2] b. Express the decimal number 13 as a "flag bit" number. (Write commas between your blocks in your answer so we can read it easily!)

Solution : The decimal number $13 = 8+4+1$ is the binary number 1101. Reversing that we get 1011. Adding flag bits to make it four blocks long, we get 11,01,11,10.

[2] c. An 8 bit unsigned binary number can represent $2^8 = 256$ distinct values. How many distinct values can we represent with a single "flag bit" number of 8 bits (or fewer)?

Solution : 8 bits is 4 blocks, which means a maximum of 4 binary digits. That's only $2^4 = 16$ distinct values.

[2] d. "Flag bit" numbers have the interesting property that we can "smash" them together and still tell where one ends and the next begins. For example, 01,10,10 must be two numbers (the decimal numbers 2 and then 1) rather than just one because the second block's 0 flag bit ends the first number and the third block's 0 flag bit ends the second number.

Circle each separate "flag bit" number in this sequence of blocks: 10,01,10,11,11,10

Solution : Here are the separate numbers: 10; 01,10; 11,11,10

[15] e. In this part, you will design a circuit that takes 4 bits of input f_1, f_2, f_3, f_4 that constitute two blocks like $f_1 f_2, f_3 f_4$. You'll first design a circuit to test whether the input contains at least one whole flag bit number. Then, you'll design a circuit that assumes the input has at least one whole flag bit number and produces its value as a 2-bit unsigned binary number.

You **must** show clear design steps, **including truth table(s), logical expression(s), and a correct circuit**. Some credit is reserved for the elegance of your solution, but most can be earned for any clear, correct answer.

i. **Using only f_2 and f_4, produce an output v that is 1 (true) exactly when the input contains at least one whole flag bit number**. For example, all of these should produce $v = 1$: 01,10 is a single whole flag bit number, 00,10 is two whole flag bit numbers, 10,11 contains a whole flag bit number (10) even though it ends with an incomplete flag bit number. **However**, 11,11 should produce $v = 0$ because it is not a whole flag bit number.

ii. **Using only f_1, f_2, and f_3 (and assuming that $f_4 = 0$), convert the *first* whole flag bit number in the input into the corresponding 2-bit unsigned binary number** $o = o_1 o_2$. For example, 01,10 should produce $o = 10$ (i.e., $o_1 = 1$ and $o_2 = 0$, representing the decimal number 2) because the whole input is the single flag bit number representing the decimal number 2. 00,10 should produce

$o = 00$ because the first whole flag bit number in 00,10 is 00, which represents the decimal value 0. Similarly, 10,11 should produce $o = 01$. Finally, even though 11,11 does not represent a whole flag bit number, we assume for this part that f_4 is actually 0 instead and produce $o = 11$ because the decimal value of the flag bit number 11,10 is 3.

Solution : v **output:** Remember that we only need f_2 and f_4. Let's make a truth table:

f_2	f_4	v
0	0	1
0	1	1
1	0	1
1	1	0

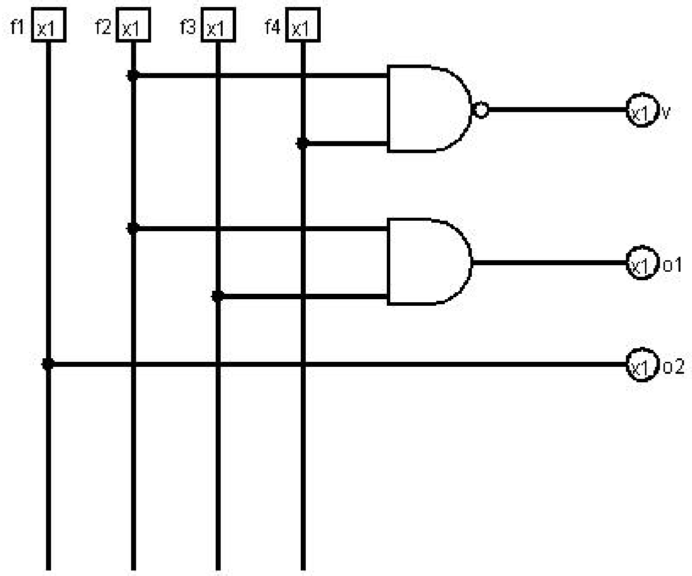
We could build an expression for each true row: $(\sim f_2 \wedge \sim f_4) \vee (\sim f_2 \wedge f_4) \vee (f_2 \wedge \sim f_4)$. However, we could also note that this is true when it's not the case that f_2 and f_4 are both true: $\sim(f_2 \wedge f_4)$.

o **output:** We only need f_1, f_2, and f_3. There are two bits of output:

f_1	f_2	f_3	o_1	o_2
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Again, a solution that builds expressions for each true row for o_1 and for each true row for o_2 would be fine, but we'll jump straight to simpler solutions. We note that o_2 simply equals f_1. o_1 is true when f_2 and f_3 are both true and false otherwise: $f_2 \wedge f_3$.

We made a single overall circuit for these two problems:



[17] 4. **Propositional Logic Proof**

Prove/disprove the argument in the following two questions. The premises in both parts a and b are the same.

[9] a. Prove that the following argument is **valid**. Justify each step using logical equivalences or rules of inference on Dave's Awesome Handout. Do not re-write the premises.

1. $\sim(r \vee \sim s)$
2. $(\sim u \wedge t) \rightarrow r$
3. $\sim x \oplus \sim q$
4. $\sim s \vee t$
5. $t \rightarrow \sim q$

$\therefore\ x$

Solution : Proof:

6.	$\sim r \wedge s$	De Morgan's on (1).
7.	s	Specialization on (6).
8.	t	Elimination on (4) and (7).
9.	$\sim q$	Modus ponens on (5) and (8).
10.	$(\sim x \vee \sim q) \wedge \sim(\sim x \wedge \sim q)$	Definition of XOR on (3).
11.	$\sim(\sim x \wedge \sim q)$	Specialization on (10).
12.	$x \vee q$	De Morgan's on (11).
13.	x	Elimination on (9) and (12).

QED

[8] b. Prove that the following argument is **invalid**.

1. $\sim(r \vee \sim s)$
2. $(\sim u \wedge t) \to r$
3. $\sim x \oplus \sim q$
4. $\sim s \vee t$
5. $t \to \sim q$

$\therefore \sim u \wedge t$

Solution: Let $r = F, s = T, t = T, q = F, x = T$, and $u = T$, then all the premises are true and the conclusion is false. So the argument is invalid.

[7] 5. **Translating From English to Predicate Logic**

THIS TEXT EXACTLY REPEATS THE PRE-READING YOU WERE GIVEN, UP TO "END OF PRE-READING" BELOW.

Consider the following definitions:

- A: all (non-human) animals near the barn.
- P: all people near the barn.
- L: all locations near the barn.
- $\mathit{Cow}(x)$: animal x is a cow.
- $\mathit{Pig}(x)$: animal x is a pig.
- $\mathit{Friends}(x, y)$: x and y are friends.
- $\mathit{Noise}(x, y)$: x has made an animal noise at y.
- $\mathit{Enjoy}(x, y)$: x enjoys spending time at location y.

For example, we can translate the English statement "Alice enjoys spending time anywhere near the barn" into predicate logic as $\forall x \in L, \mathit{Enjoy}(\text{Alice}, x)$.

END OF PRE-READING

Translate each of the following English statements into predicate logic.

[1] a. There is a cow who is friends with Ryan.

Solution : $\exists x \in A, \mathit{Cow}(x) \wedge \mathit{Friends}(x, \text{Ryan})$

[3] b. There is at least one cow and at least one pig who have made animal noises at Steve.

Solution : $\big(\exists x \in A, \mathit{Cow}(x) \wedge \mathit{Noise}(x, \text{Steve})\big) \wedge \big(\exists x \in A, \mathit{Pig}(x) \wedge \mathit{Noise}(x, \text{Steve})\big)$

[3] c. Every cow enjoys spending time in the pasture (a particular location in L), and is friends with at least one pig.

Solution : $\forall x \in A, \mathit{Cow}(x) \to \big(\mathit{Enjoy}(x, \text{pasture}) \wedge \exists y \in A, \mathit{Pig}(y) \wedge \mathit{Friends}(x, y)\big)$

[4] 6. **Translating From Predicate Logic to English** Using the same definitions as in the previous question, translate the following predicate logic statement into English:

[2] a. Translate:

$$\exists x \in A, \mathit{Cow}(x) \wedge \big(\forall y \in A, \mathit{Pig}(y) \to \mathit{Friends}(x, y)\big)$$

Solution : There is a cow who is friends with every pig.

[2] b. Translate:

$$\exists x \in L, \big(\exists y \in A, \mathit{Cow}(y) \land \mathit{Enjoy}(y, x)\big) \land \big(\exists y \in A, \mathit{Pig}(y) \land \mathit{Enjoy}(y, x)\big)$$

Solution : There is a location near the barn at which both a cow and a pig enjoy spending time.

[3] 7. **A Little Set and Function Theory**

[2] a. Which of the following sets is equal to $S = \{2, 3, 5, 7\}$? *Clearly* indicate **all** that apply.

i. $\{2, 3, 5, 7\}$
ii. $\{3, 2, 5, 7\}$
iii. $\{\{2, 3\}, \{5\}, \{7\}\}$
iv. $\{2, 3, 3, 7, 7, 5\}$
v. $\{\{2, 2, 3\}, \{5, 5\}, \{\}, \{7\}\}$
vi. $\{2, 3, 5, 7, \{\}, \{\}\}$

Solution : 7(a)i, 7(a)ii, 7(a)iv

[1] b. Let P be a predicate on the domain D. We can conceive of P as a function mapping elements from D to true or false: $\{T, F\}$. Write a predicate logic expression that expresses the following fact about P: P never maps any element of D to F.

Solution : $\forall x \in D, P(x)$
Alternately: $\sim\exists x \in D, \sim P(x)$