

CSE 140 Homework Two

October 9, 2016

*Only **Problem Set Part B** will be graded. Turn in only **Problem Set Part B** which will be due on October 19, 2016 (Wednesday) at 3:00pm.*

1 Problem Set Part A

All questions in this part are from Roth&Kinney, 7th Edition.

- 2.1, 2.3, 2.5, 2.6, 2.7, 2.8, 2.11, 2.22, 2.30
- 3.6, 3.9, 3.10, 3.12, 3.15, 3.16, 3.18, 3.19, 3.20, 3.21, 3.29, 3.30, 3.33, 3.34, 3.36, 3.37, 3.38
- 4.5, 4.6, 4.8, 4.10, 4.13, 4.15, 4.17, 4.26, 4.29, 4.32, 4.40, 4.44

2 Problem Set Part B

1. (X(N)ORS in Boolean Algebra)

Ever since you saw XORs doing wonderful feats in Error Correction & Hamming codes, you have fallen in love with them. Your obsession with XORs and XNORs has been kindled further as you saw a number of theorems related to XORs and XNORs in Chapter 3 of your book. Equalities such as $X \oplus X' = 1$ & $X \odot X' = 0$ seemed so similar to the complement axioms defined over AND & OR that it prodded you to see if XOR and/or XNOR can take their rightful position in the pantheon of Boolean algebras by perhaps displacing one (or maybe both) of the (AND,OR) pair defining Boolean algebras¹.

To help you in your quest, you start off by brushing up on your knowledge of Boolean theorems involving XORs & XNORs. Please use *Boolean algebra based proof techniques* to prove any of the following equalities that hold; of course, if you think that a particular equality does not hold, a simple instance of Boolean values will suffice to disprove it.

(Part A) Does $X \oplus (YZ) = (X \oplus Y)(X \oplus Z)$ hold? Please prove or disprove.

(Part B) Does $X(Y \oplus Z) = XY \oplus XZ$ hold? Please prove or disprove.

(Part C) Does $X \odot X = 1$ hold? Please prove or disprove.

(Part D) Does $X + (Y \odot Z) = (X + Y) \odot (X + Z)$ hold? Please prove or disprove.

¹Note: We use $\oplus$ to denote XOR and $\odot$ to denote XNOR.

Now that you have refreshed some of your familiarity with manipulating Boolean expressions involving XORs and XNORs, it is time to move onto the Holy Grail and see if you can help these rather sympathetic operators take their rightful place in the Boolean pantheon.

(Part E) Can $\cdot$ and $\oplus$ be the two constituent operators (instead of $\cdot$ & $+$) of the Boolean algebra axiomatic definitions? Why or why not?

(Part F) Can $\odot$ and $+$ be the two constituent operators (instead of $\cdot$ & $+$) of the Boolean algebra axiomatic definitions? Why or why not?

(Part G) Can $\odot$ and $\oplus$ be the two constituent operators (instead of $\cdot$ & $+$) of the Boolean algebra axiomatic definitions? Why or why not?

2. When Consensus Attacks

This problem will test your ability to correctly apply boolean axioms and theorems to complete proofs, as well as to identify mistakes in the use of these axioms and theorems.

(Part A) In class, you were shown the consensus theorem, which is not obvious at first glance, but which can prove useful in simplifying boolean expressions.

A related and also useful extension of the consensus theorem allows one to convert a two-term sum-of-products expression into a two-term product-of-sums expression:

$$xy + x'z = (x + z)(x' + y)$$

We will call this equivalence the **Distributed Consensus Theorem**.

For this part of the problem, please complete the proof of the Distributed Consensus Theorem by filling in the expression which results from the application of each specified axiom or theorem.

$xy + x'z$	Given
	Distributivity (all terms)
	Complement
	Identity
$(x + z)(x' + y)$	Consensus

(Part B) Now, let us see how this theorem can be useful. Complete the proof below by filling in the expression produced by the specified axiom/theorem, or by filling in the axiom or theorem that results in the given expression. (Hint: try working forwards and backwards.)

$x'y(w + x) + (x + y')(w' + y)$	Given
$x'y(w + x) + (x'y)'(w' + y)$	
	Consensus
	DeMorgan's Laws
	Distributed Consensus (all but first term)
$x'y(w + x) + w'(y' + x) + y(w + x)$	Distributivity (all but first term)
$w'(y' + x) + y(x + w)$	

(Part C) A friend of yours is still struggling with boolean axioms. He is working on a proof, and he thinks he has completed it, but he decides to double-check his work with you. As you read his proof, you see that although the beginning and end expressions are indeed equivalent, he has made a few mistakes in his applications of boolean axioms (in addition to taking an extremely roundabout path to what is a straightforward application of distributivity). In the space below the given proof, please identify which line numbers have mistakes, and explain the error. (There are **three errors**, all of which are in the *application* of the specified axiom of theorem.)

Line	Expression	Axiom/Theorem
1	$y'z + wz$	Given
2	$wy' + y'z + wz$	Consensus
3	$((wy' + y'z + wz)')'$	Involution
4	$((w' + y)(y + z')(w' + z'))'$	DeMorgan's Laws
5	$(w'yw' + w'yz' + w'z'w' + w'z'z' + yyw' + yyz' + yz'w' + yz'z')'$	Distributivity
6	$(w'y + w'z' + yz' + w'yz')'$	Idempotency
7	$(w'y + w'z' + yz')'$	Absorption
8	$(w + y')(w + z)(y' + z)$	DeMorgan's Laws
9	$(w + z)(y' + z)$	Consensus
10	$y'z + wz$	Distributed Consensus
11	$z(y' + w)$	Distributivity

3. (Many-Bit Error Detection)

This question expands your understanding of 7-bit Hamming Code codewords to larger numbers of bit errors.

For the parts that ask for a numeric value, please show your intermediate work to arrive at the numeric value as well.

(Part A) Consider triple-bit errors in a 7-bit (4 data bit, 3 parity bit) Hamming Code codeword. A triple-bit error is the occurrence of three single-bit errors at three distinct locations. Since the Hamming Codes are a distance-3 code, one might think that a 3-bit error would transform one valid codeword into another, thus rendering 3-bit errors undetectable. However, it turns out that some 3-bit errors can in fact be detected. (Remember that detecting an error is not the same as being able to identify or correct it.)

- (i) Suppose that a triple-bit error occurs, and suppose you knew that m_1 and m_2 are two of the three bits that flipped (i.e., these bits are in error). **If this triple-bit error was not detected by the correction equations, what are the possible positions of the third bit that was flipped?**

- (ii) Now, suppose that you only knew that m_1 flipped, but not the other two. **If the error was not detected, what are the possible pairs of bits that could have flipped along with m_1 ?**

- (iii) Expanding on this thought process, **how many triples of bit positions constitute an undetectable triple-bit error?** (Hint: How many choices did you get in parts i) and ii)? Also, remember that two triples with the same elements in different order are equivalent, so you may have to account for counting each triple multiple times. Keep in mind that in a 7-bit code word, there are $\binom{7}{x}$ possible x -bit errors, so if your answer exceeds this value, you should re-check your work.)

(Part B) Now let's analyze N -bit errors, wherein an error occurs at N distinct bit positions. This is actually fairly easy given the results of (Part A), the results from the homework, and a bit of cleverness.

- (i) Let's start with an extreme but easier case. Suppose during a transmission, a 7-bit error occurs. That is, ALL the bits in the codeword got flipped. **How many 7-bit errors are not detectable?**

- (ii) With the results of (Part B.i) in mind, one can think of a 4 bit error as a 7-bit error followed by a 3-bit error. The 3-bit error cancels three of the flips that occurred during the 7-bit error, leaving only 4 bits erroneous.

Given this, **how many 4-bit errors are undetectable?**

- (iii) In the same way, we can analyze 5- and 6-bit errors. **How many 5-bit errors are undetectable? What about 6-bit errors?**

(Part C) Now that we have completely analyzed the 7-bit Hamming Code, let's turn our attention to a related, but larger, 15-bit Hamming Code (with 11 data bits and 4 parity bits). Below, we give a table that shows a possible mapping between single-bit error locations and correction equation values, so that single-bit error correction is possible. **Tell us what the associated correction equations and parity equations should be so that this mapping is realized.** (Hint: Start with the correction equations. What does the value in a particular row and column indicate about the corresponding bit position and correction equation?)

Error Location	Syndrome ²				p_0	=
	C_3	C_2	C_1	C_0	p_1	=
m_{10}	1	0	0	1	p_2	=
m_9	1	1	0	1	p_3	=
m_8	1	0	1	1	C_0	=
m_7	1	1	1	1	C_1	=
m_6	1	0	1	0	C_2	=
m_5	1	1	1	0	C_3	=
m_4	0	0	1	1		
p_3	0	0	1	0		
m_3	1	1	0	0		
m_2	0	1	1	1		
m_1	0	1	0	1		
p_2	0	1	0	0		
m_0	0	1	1	0		
p_1	0	0	0	1		
p_0	1	0	0	0		

2. The word "Syndrome" means the apparent result of an error – in this case, the values of the correction equations.

(Part D) Now, let us turn to analyzing N -bit errors in this 15-bit Code.

(i) How many 2-bit errors are undetectable?

(ii) How many 3-bit errors are undetectable?

(iii) How many 12-bit errors are undetectable?

(iv) How many 11-bit errors are undetectable?