

CSE 140 Homework Two

January 29, 2013

*Only **Problem Set Part B** will be graded. Turn in only **Problem Set Part B** which will be due on February 13, 2013 (Wednesday) at 3:00pm.*

1 Problem Set Part A

- textbook 2.28
- textbook 2.29
- textbook 3.1(a)(b)
- textbook 3.6(b)(d)
- textbook 3.7
- textbook 3.10(b)(c)
- textbook 3.12(a)(d)
- textbook 3.14
- textbook 3.15(a)(b)
- textbook 3.16(a)
- textbook 3.19(b)
- textbook 4.3
- textbook 4.5
- textbook 4.6
- textbook 4.8

2 Problem Set Part B

1 (Error Correction Code)

As a distance-3 code, the **Hamming Code** can correct single-bit errors or detect up to two-bit errors. Because of these good features, a communication company has been trying to improve the reliability of their communication system by utilizing the 7-bit Hamming Code. However, because of some complicated interference effect between the signals, a single error, called the **aggressor** error, in this communication system seems to always effect another bit, called the **victim** bit, which flips slightly to surpass the voltage threshold and be toggled to an incorrect value, resulting in eventually a two-bit error. (The good news is that because the victim bit has been pushed only enough to surpass the voltage threshold, it does not seem to have sufficient strength to in turn toggle another bit, thus, ensuring that all errors are exactly double-bit errors.) After extensive experiments, the engineers in this company have noticed that the possibility of having multiple *aggressor errors* at the same time in this system is so insignificant that we do not have to worry about them.

Hamming code format

Position	7	6	5	4	3	2	1
Bit	m_3	m_2	m_1	p_2	m_0	p_1	p_0

(Part A) As you might have already realized, the error correlation mechanism imposes challenge in error correction as the single-bit error correction algorithm of the Hamming code will pinpoint a false error bit (the **syndrome**) that differs from the actual error bits (the aggressor and the victim bits). In the following, we are providing you two possible correlation mechanisms in two tables. For each table, specify the location of the false error bit (the syndrome) that will be mistakenly identified as the source of the error by the single-bit error correction algorithm.

Error correlation set #1

Aggressor bit	Victim bit	Syndrome
p_0	m_2	
p_1	m_0	
m_0	p_0	
p_2	m_3	
m_1	p_0	
m_2	m_0	
m_3	p_0	

Error correlation set #2

Aggressor bit	Victim bit	Syndrome
p_0	m_0	
p_1	p_0	
m_0	p_2	
p_2	m_2	
m_1	p_1	
m_2	m_1	
m_3	p_2	

(Part B) For each set of double-bit errors specified by the two correlation tables above, analyze whether the set of errors is fully correctable or not. If you think they are, please tell us how you would go about correcting them. If you think they are not fully correctable, please tell us whether you think at least one of the two errors can be corrected or whether neither of the errors can be corrected in your estimation.

2 (Boolean Table Algebra)

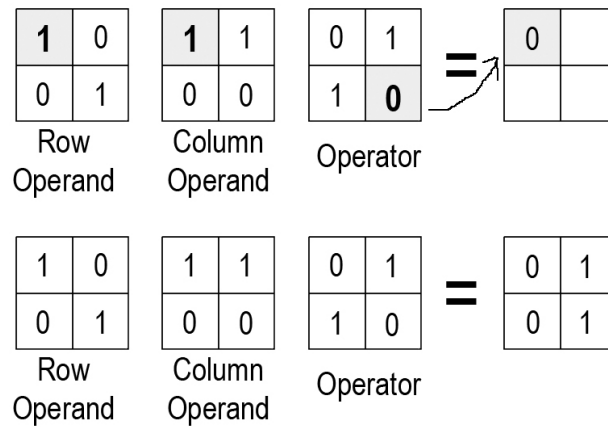
Whilst hacking into top-secret government servers one day, you discover a set of files which contain a curious sequence of characters. These characters appear to be squares, each of which has four quadrants, and in each of these four quadrants is either the number 0 or 1. For example:

		0	1	
A =	0	1	0	
	1	0	1	$A(0,0) = 1$ $A(0,1) = 0$ $A(1,0) = 0$ $A(1,1) = 1$

You can see that each entry is specified by a row and a column, each of which can be either 0 or 1, and each entry is referenced symbolically by “A(row, column)”.

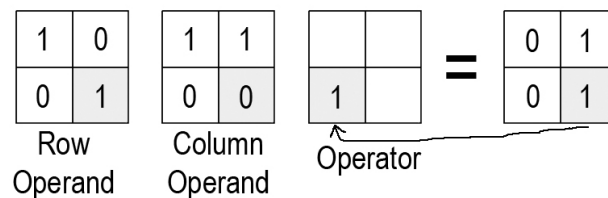
Curious, you send a quick email to one of your friends who happens to be a member of a clandestine group of hackers, and ask whether they know anything about this. They reply by saying that their group uses this system of characters to encode messages to one another, and that you have likely found one of their intercepted messages. You are told not to worry, the code is so clever that the CIA will require numerous Mayan Calendar iterations to decode it, which of course does not put you at ease since you are fully aware that the CIA is populated with CSE 140 dropouts. Your friend, who trusts you completely, explains that these symbols are actually mathematical symbols, and that they double as operators on each other. They inform you that an operation is computed as follows:

1. Each operation is specified by three symbols in a row. The first two symbols are the **operands**, and the third symbol is the **operator**.
2. Look at the entries i,j of the **operands**. The entry for the first operand specifies a row R , and the entry for the second operand specifies a column C .
3. In the **operator**, find the entry R,C . Place this value in the entry i,j in the **result** symbol.



Your friend also goes on to say that in some cases, if the two operand symbols are known and the result is known, but the operator is unknown, we can compute the entries of the operator with the following steps:

1. For entry i,j in the **operands**, find the row and column by looking at the operands, as in step 2 previously.
2. Look at the corresponding entry i,j in the **result** symbol. Place that value (1 or 0) at the appropriate row and column of the **operator** symbol.



You note that not all operand-result combinations allow you to infer all entries of the operator. This is because in order to infer all of the entries, the operands need to exhibit all four (row, column) combinations, 00, 01, 10, and 11.

At the end of their email, your friend, who knows that you are a brilliant computer scientist, invites you to solve a sequence of puzzles involving these symbols. Having nothing better to do, and being unable to resist a challenge, you reply enthusiastically that you will accept his invitation.

(Part A) You are given that the following symbols are the “1” and “0” (ie, the multiplicative and additive identities) in a boolean algebra:

$$1 \equiv \begin{array}{|c|c|} \hline 0 & 0 \\ \hline 0 & 1 \\ \hline \end{array} \quad 0 \equiv \begin{array}{|c|c|} \hline 0 & 1 \\ \hline 1 & 1 \\ \hline \end{array}$$

You are given that $1 = 0'$, and vice versa. Use the **Complement axiom** of boolean algebras to derive the symbols which serve as the operators “+” and “•”. (Hint: If you get stuck, the **Commutativity** of the operators “+” and “•” may prove useful in deriving these operators.)

(Part B) In less than a minute, you derive the operators and send a reply with your response. Pretending to be unphased by the speed of your delivery of the solution, your friend replies with another puzzle. He asks you to use the **Identity-Absorption theorem** (Theorem 2 in your book) of boolean algebras to discover if there are other symbols that are compatible with this boolean algebra along with the 1 and 0 already given, and the “+” and “•” operators you derived in (Part A). In other words, are there other symbols that satisfy the **Identity-Absorption** equations? If so, list them, and specify the complement of each. (Hint: Start out with a completely generic symbol with entries A, B, C and D and then try to resolve these entries.)

(Part C) As you are emailing to your friend the solutions to the previous problem, a sudden thought disturbs you. The original 2-valued algebra that you derived in Part A obviously satisfies the Cardinality, Identity, Complement, and Commutativity axioms, yet you are suspicious that it might not be **Closed** under “+” or “•”. Please put your mind at ease by showing that indeed, the 2-valued algebra is closed under both operators.

(Extra Credit) (Part D) At the end of your email to your friend containing the solutions to the last problem, you add that you are suspicious that the original 2-valued algebra from the first puzzle is really a boolean algebra. Though this system obviously satisfies the Cardinality, Closedness, Identity, Complement, and Commutativity axioms of a boolean algebra, you cannot quite see that it satisfies the Distributivity axiom. (You are of course lying, as you have verified this fact in your head, but it is apparent that your friend is becoming offended by your magnanimous brilliance.) Excited by his chance to show you that he too is brilliant, your friend replies that indeed the algebra is distributive, and agrees to complete half of the proof for you. He computes the generic equation $A \bullet (B+C)$:

$$\begin{aligned}
 (B+C) &= \begin{array}{|c|c|} \hline 0 & b \\ \hline b & 1 \\ \hline \end{array} \begin{array}{|c|c|} \hline 0 & c \\ \hline c & 1 \\ \hline \end{array} \begin{array}{|c|c|} \hline +(0,0) & +(0,1) \\ \hline +(1,0) & +(1,1) \\ \hline \end{array} = \begin{array}{|c|c|} \hline +(0,0) & +(b,c) \\ \hline +(b,c) & +(1,1) \\ \hline \end{array} \\
 A(B+C) &= \begin{array}{|c|c|} \hline 0 & a \\ \hline a & 1 \\ \hline \end{array} \begin{array}{|c|c|} \hline +(0,0) & +(b,c) \\ \hline +(b,c) & +(1,1) \\ \hline \end{array} \begin{array}{|c|c|} \hline \bullet(0,0) & \bullet(0,1) \\ \hline \bullet(1,0) & \bullet(1,1) \\ \hline \end{array} = \begin{array}{|c|c|} \hline \bullet(0,+(0,0)) & \bullet(a,+(b,c)) \\ \hline \bullet(a,+(b,c)) & \bullet(1,+(1,1)) \\ \hline \end{array}
 \end{aligned}$$

He then asks you to compute the result of the distributed expression, $A \bullet B + A \bullet C$, and then to complete the proof by explaining why the result is equivalent to the result computed above, and thus why the operator “ $\bullet$ ” is distributive over “ $+$ ”.

(Part E) After you have solved the last puzzle, you see that it is trivial to explain that the operator “ $+$ ” is also distributive over “ $\bullet$ ”, and you include your explanation of your moment of insight along with your final email to your friend, thanking him for such an interesting puzzle (but remember, you’re emailing your friend, not the CSE 140 teaching staff).