

Homework 1

Due: Jan 20, 2012

CS103, Winter 2011-2012

(9:47 AM 1/14/2012) Several clarifications and corrections have been made since this was first released on Friday.

- Guidance about the level of detail in equational proofs has been added.
- Problem 1b was mangled, and has been changed significantly.
- The third column of the truth table in problem 4 had \leftrightarrow but now has \oplus .
- Problem 4 has been clarified to say you can use either truth tables or equational proof to prove it.

Points will be allocated to the problems below in approximate proportion to the amount of work required, which varies.

In all equational proofs, you can use the associative and commutative laws without justification or comment, and you can merge several applications of the same law into one step when what you did is obvious.

Problem 1:

Write equational proofs of the following in the style shown in lecture. In addition to the Boolean identities in the lecture, you may use the property $P \rightarrow Q \equiv \neg P \vee Q$ and call it “implies-or.”

These are all very important identities for \rightarrow and \leftrightarrow .

- $P \rightarrow (Q \rightarrow R) \equiv (P \wedge Q) \rightarrow R$
- $(P \wedge Q) \rightarrow R \equiv (P \rightarrow R) \vee (Q \rightarrow R)$
- $P \leftrightarrow Q \equiv (\neg P \vee Q) \wedge (P \vee \neg Q)$
- $P \leftrightarrow Q \equiv (P \wedge Q) \vee (\neg P \wedge \neg Q)$
- $P \leftrightarrow Q \equiv \neg P \leftrightarrow \neg Q$

Problem 2:

The second lecture asserted that

$$[P_1 \rightarrow (\neg P_2 \wedge \neg P_3)] \wedge [P_2 \rightarrow (\neg P_1 \wedge \neg P_3)] \wedge [P_3 \rightarrow (\neg P_1 \wedge \neg P_2)]$$

is logically equivalent to $(\neg P_1 \wedge \neg P_2) \vee (\neg P_1 \wedge \neg P_3) \vee (\neg P_2 \wedge \neg P_3)$.

Prove that the two formulas are logically equivalent by constructing a truth table. To reduce tedium, just write the truth values for the three major subformulas in each formula.

Problem 3:

\oplus is the “XOR” (exclusive-or) operation, which has the following truth table:

P	Q	$P \oplus Q$
T	T	F
T	F	T
F	T	T
F	F	F

Prove that

- a. $P \oplus Q \equiv \neg(P \leftrightarrow Q)$
- b. $P \oplus Q \equiv (\neg P \leftrightarrow Q)$

You may use truth tables or an equational proof using Boolean identities.

Problem 4:

Section 1.2 of the Rosen book discusses logic puzzles, such as those by Raymond Smullyan about islands with knights, who always tell the truth, and knaves who always lie. Unfortunately, the example solution in the book is a bit wordy, so we would like a more crisp, mathematical approach.

Here is an example from the book:

Person A says “B is a knight”

Person B says “The two of us are of opposite types”?

To translate this into propositional logic, let KA represent “A is a knight”, KB represent “B is a knight”. Since everyone is either a knight or knave, “A is a knave” is simply represented by $\neg KA$. Now, let’s think about how to translate the following assertion:

“Person A says that ‘B is a knight’”

Note that person A is either knight or knave, and these two possibilities are exhaustive. Let’s consider the implications of each possibility:

If person A is a knight, then they always tell the truth, which means that their assertion, “B is a knight”, is true. In other words, we know that “if A is a knight, then B is a knight”, or, in the language of propositional logic, $KA \rightarrow KB$.

If person A is a knave, then they always lie, which means that their assertion, ‘B is a knight,’ is false. Thus, we know that “if A is a knave, then B is a knave” ,or $\neg KA \rightarrow \neg KB$ (recall that $\neg KB$ is equivalent to “B is a knave”).

Thus $KA \rightarrow KB$ and $\neg KA \rightarrow \neg KB$ both hold, and together they contain all of the “content” contained in our original assertion. This means we encode our assertion as the conjunction of these two predicate sentences. In other words, “Person A says that ‘B is a knight’” is encoded as

$$(KA \rightarrow KB) \wedge (\neg KA \rightarrow \neg KB)$$

Note that $\neg KA \rightarrow \neg KB$ is equivalent to $KB \rightarrow KA$, so we can instead encode our assertion as $(KA \rightarrow KB) \wedge (KB \rightarrow KA)$. But then this is simply equivalent to $(KA \leftrightarrow KB)$, so we obtain the following equivalent (but more compact) result):

“Person A says that ‘B is a knight’” is encoded as $(KA \leftrightarrow KB)$.

Note that in general ,as long as we’re on knight/knave island , the assertion that “Person X says Y” is encoded as $(X \leftrightarrow Y)$

- a. Encode person B’s assertion from the problem discussed above. In other words, encode the following fact:

Person B says “The two of us are of opposite types”

Hint: your answer should be in the form $__ \leftrightarrow (__ \leftrightarrow __)$

- b. Consider the encoding everything we know from example, ie

Person A says “B is a knight”

Person B says “The two of us are of opposite types”

Using the work done in the problem prompt and your solution to part **a**, we know that we can encode all of this information as the following formula:

$$(KA \leftrightarrow KB) \wedge (__ \leftrightarrow (__ \leftrightarrow __))$$

If this formula is not satisfiable, the situation described in the problem cannot exist (this would mean that the information we've been given is logically inconsistent). Otherwise, there is at least one solution. However, these problems are often written so that there's a single, clean solution. We will examine whether or not that's the case here.

Write a truth table for this formula. To save tedium, you don't need to show the truth values for all of the subformulas - just show three columns, two for KA, KB and one for the whole formula. There should be one **T** entry.

Finally, give the answer to the puzzle in English.

- c. Now, suppose we have the following two facts

A says "Both of us are knights"

B says "A is a knave"

Encode this information as a formula in predicate logic, and use this formula to solve the riddle. Using a truth table is fine, although manipulating the formula to arrive at an answer is OK as well.

Hint: consider converting the formula into disjunctive normal form, and reasoning from there. Note: ignore the two assertions we considered in parts **a** and **b**.

Problem 5:

This problem is about converting arbitrary logical formulas to CNF in polynomial time (CNF) using the method discussed at the end of lecture 2 that introduces additional propositional variables.

- a. The method produces a conjunction of formulas like $X_i \leftrightarrow (P \vee Q)$, which then need to be converted to CNF, and the lecture said this could be done with only a constant factor expansion.

Using Boolean identities from the lecture and this homework, show how $X_i \leftrightarrow (P \wedge Q)$ can be rewritten to a logically equivalent CNF formula.

- b. Convert the formula $(P_1 \wedge Q_1) \vee (P_2 \wedge Q_2)$ to CNF using the *first* method discussed in lecture 2, which uses the distributive law of \vee over \wedge and *does not* introduce new variables. Show the major steps in your translation.

- c. Based on the previous problem, consider a formula of the form $(P_1 \wedge Q_1) \vee (P_2 \wedge Q_2) \vee \dots \vee (P_n \wedge Q_n)$. If you used the first method to convert to CNF that was discussed in the lecture, which uses the distributive law of \vee over \wedge and *does not* introduce new variables, how many clauses would the resulting CNF formula have (as a function of n)?
- d. Convert the formula $(P_1 \wedge Q_1) \vee (P_2 \wedge Q_2)$ to CNF using the second method discussed in lecture 2, which introduced new variables. Show the major steps in your translation, and show the resulting formula.
- e. Suppose you did the same translation to a longer formula, like $(P_1 \wedge Q_1) \vee (P_2 \wedge Q_2) \vee \dots \vee (P_n \wedge Q_n)$. How large would the translated formula be, as a function of n (based on your answer to the previous formula)?

Problem 6:

The Sudoku game rules of lecture 2 are actually written in a form of predicate logic. Here, we will write it using standard logic notation instead of lisp.

This problem is to prove that the way it was written is equivalent to a more obvious translation.

Although it is a first-order formula, the reasoning is propositional. You can replace one side of a Boolean identity by the other side anywhere in a first-order formula, and the transformed formula will be equivalent to the original formula.

In English, the rule is: Two different squares in the same row must be filled in with different values.

Square(s) is a predicate saying that s is a square somewhere on the board.

board is a function whose value is the digit written in that square.

A natural translation of the English might be:

$$\forall s_1, s_2 (\text{Square}(s_1) \wedge \text{Square}(s_2) \rightarrow [\text{SameRow}(s_1, s_2) \wedge s_1 \neq s_2 \rightarrow \text{board}(s_1) \neq \text{board}(s_2)])$$

The actual formula was closer to this:

$$\forall s_1, s_2 (\text{Square}(s_1) \wedge \text{Square}(s_2) \rightarrow [\text{SameRow}(s_1, s_2) \wedge \text{board}(s_1) = \text{board}(s_2) \rightarrow s_1 = s_2])$$

In English: “If two values are squares, then if they are in the same row and have the same value filled in, they must be the same square.”

*Write an equational proof showing that the first formula is equivalent to the second.
The contrapositive rule is important, and so is the first property in problem 1.*