Problem Set 4

This fourth problem set explores logic in many forms. Some of the problems explore how logic is useful as a proof aid, while others explore computational aspects of logic.

Start this problem set early. It contains seven problems (plus one survey question and one extra-credit problem), several of which require a fair amount of thought. I would suggest reading through this problem set at least once as soon as you get it to get a sense of what it covers.

As much as you possibly can, please try to work on this problem set individually. That said, if you do work with others, please be sure to cite who you are working with and on what problems. For more details, see the section on the honor code in the course information handout.

In any question that asks for a proof, you **must** provide a rigorous mathematical proof. You cannot draw a picture or argue by intuition. You should, at the very least, state what type of proof you are using, and (if proceeding by contradiction, contrapositive, or induction) state exactly what it is that you are trying to show. If we specify that a proof must be done a certain way, you must use that particular proof technique; otherwise you may prove the result however you wish.

If you are asked to prove something by induction, you may use either weak induction or strong induction. You should state your base case before you prove it, and should state what the inductive hypothesis is before you prove the inductive step.

As always, please feel free to drop by office hours or send us emails if you have any questions. We'd be happy to help out.

This problem set has 125 possible points. It is weighted at 7% of your total grade. **There is no checkpoint problem on this problem set**, and future problem sets will not have checkpoints.

Good luck, and have fun!

Due Friday, May 4 at 2:15PM

Problem One: Two Flavors of Antisymmetry (4 Points)

In lecture, we claimed that the following definitions of antisymmetry are equivalent:

$$\forall x.\ \forall y.\ (xRy \land yRx \to x = y)$$

and

$$\forall x.\ \forall y.\ (xRy \land x \neq y \to \neg yRx)$$

The reason for this is that the propositional formulas

$$(p \land q) \to r$$

and

$$(p \land \neg r) \to \neg q$$

are equivalent. Prove this by writing a truth table for each and showing that they are equivalent.

Problem Two: Translating into Logic (24 points)

In each of the following, you will be given a list of first-order predicates and functions along with an English sentence. In each case, write a statement in first-order logic that expresses the indicated sentence. The statement you write should can use any first-order construct (equality, connectives, quantifiers, etc.), but you must only use the predicates and functions provided.

As an example, if you were given just the predicates *Integer(x)*, which returns if x is an integer, and the function *Plus(x, y)*, which returns $x + y$, you could write the statement "there is some even integer" as

$$\exists n.\ \exists k.\ (Integer(n) \land Integer(k) \land Plus(k, k) = n)$$

since this asserts that some integer n is equal to $2k$ for integer k. However, you could not write

$$\exists n.\ (Integer(n) \land Even(n))$$

because there is no *Even* predicate. The point of this question is to get you to think how to express certain concepts in first-order logic given a limited set of predicates, so feel free to write any formula you'd like as long as you don't invent your own predicates or functions.

i. Given the predicate

   Natural(x), which states that x is an natural number,

   the function

   Product(x, y), which yields the product of x and y,

   and the constant symbols *1* and *137*, write a statement in first-order logic that says "137 is a prime number."

ii. Given the predicates

   Word(x), which states that x is a word,

   Definition(x), which states that x is a definition, and

   Means(x, y), which states that x means y,

   write a statement in first-order logic that says "some words have exactly two meanings."

iii. Given the predicates

$x \in y$, which states that x is an element of y, and

Set(S), which states that S is a set,

write a statement in first-order logic that says "every set has a power set." As an aside, the formula that you will be writing is sometimes called the **axiom of power set**.

iv. Given the predicates

Lady(x), which states that x is a lady,

Glitters(x), which states that x glitters,

IsSureIsGold(x, y), which states that x is sure that y is gold,

Buying(x, y), which states that x buys y,

StairwayToHeaven(x), which states that x is a Stairway to Heaven

write a statement in first-order logic that says "There's a lady who's sure all that glitters is gold, and she's buying a Stairway to Heaven."*

Problem Three: Propositional Negations (16 points)

For each of the following propositional logic statements, find another statement that is the negation of the given statement. The statement you choose should not contain any instances of ¬ before a parenthesis. For example, to get the contradiction of the statement

$p \to q \to r$

you might use the following line of reasoning:

$\neg(p \to q \to r)$

$p \land \neg(q \to r)$

$p \land q \land \neg r$

Once you have found your negation, prove that is is correct by constructing a truth table for the negation of the original statement and showing it is equal to the truth table for your resulting statement. For the above case, we would construct truth tables for $\neg(p \to q \to r)$ and $p \land q \land \neg r$ as follows:

p	***q***	***r***	***¬(p → q → r)***
F	F	F	**F** F T F T F
F	F	T	**F** F T F T T
F	T	F	**F** F T T F F
F	T	T	**F** F T T T T
T	F	F	**F** T T F T F
T	F	T	**F** T T F T T
T	T	F	**T** T F T F F
T	T	T	**F** T T T T T

p	***q***	***r***	***p ∧ q ∧ ¬r***
F	F	F	F **F** F F T F
F	F	T	F **F** F F F T
F	T	F	F **F** T T T F
F	T	T	F **F** T F F T
T	F	F	T **F** F F T F
T	F	T	T **F** F F F T
T	T	F	T **T** T T T F
T	T	T	T **F** T F F T

* Let's face it – the lyrics to Led Zeppelin's "Stairway to Heaven" are impossible to decipher. Hopefully we can gain some insight by translating them into first-order logic!

Since these truth tables have the same truth values, the formulas are equivalent. The truth tables above are very detailed and you don't need to provide this level of detail in yours. However, you should at least specify the truth value of each connective.

i. $p \leftrightarrow q$

ii. $r \lor (\neg p \land q)$

iii. $\neg(p \to q) \to (p \to \neg q)$

Problem Four: First-Order Negations (16 points)

Proof by contradiction can be difficult because it is often tricky to determine what the negation of the theorem is. In this problem, you'll use first-order logic to explicitly determine the negation of statements in first-order logic.

For each of the first-order logic formulas below, find a first-order logic formula that is the negation of the original statement. Your final formula must not have any negations in it, except for direct negations of predicates. For example, the negation of

$$\forall x.\ (p(x) \to \exists y.\ (q(x) \land r(y)))$$

would be found by pushing the negation in from the outside as follows:

$$\neg(\forall x.\ (p(x) \to \exists y.\ (q(x) \land r(y))))$$

$$\exists x.\ \neg(p(x) \to \exists y.\ (q(x) \land r(y)))$$

$$\exists x.\ (p(x) \land \neg\exists y.\ (q(x) \land r(y))$$

$$\exists x.\ (p(x) \land \forall y.\ \neg(q(x) \land r(y)))$$

$$\exists x.\ (p(x) \land \forall y.\ (q(x) \to \neg r(y)))$$

You must show every step of the process of pushing the negation into the formula (along the lines of what is done above), but you do not need to formally prove that your result is correct.

i. $\forall x.\ (p(x) \to \exists y.\ q(x, y))$

ii. $(\forall x.\ \forall y.\ \forall z.\ (R(x, y) \land R(y, z) \to R(x, z))) \to (\forall x.\ \forall y.\ \forall z.\ (R(y, x) \land R(z, y) \to R(z, x)))$

iii. $\forall n \in \mathbb{N}.\ (n \geq 6 \to \exists x \in \mathbb{N}.\ \exists y \in \mathbb{N}.\ 3x + 4y = n)$

Problem Five: DPLL (8 Points)

The DPLL algorithm runs unit propagation before it runs pure literal elimination. Explain, but do not prove, why DPLL would be less efficient if it ran pure literal elimination before unit propagation. Give at least one example to support your reasoning.

Problem Six: SAT Solving Algorithms (24 Points)

Suppose that you are handed a "black-box" SAT-solving algorithm – that is, a device that takes in a propositional logic formula φ and returns whether or not φ is satisfiable. You don't know anything about the workings of this algorithm – perhaps it uses DPLL, or perhaps it constructs a truth table – but you are assured that given a formula φ it returns whether or not φ is satisfiable. Let's denote this algorithm *A*, so *A*(φ) is true iff φ is satisfiable.

i. Describe an algorithm that uses *A* as a subroutine to determine whether φ is a tautology. Prove that your algorithm is correct. Do **not** just list all possible assignments and check each individually; leverage off of algorithm *A* to get an answer directly.

ii. Suppose that you have two propositional formulas φ and ψ. You are interested in determining whether φ ≡ ψ; that is, whether φ and ψ always have the same truth values. Create an algorithm that uses *A* as a subroutine to answer this question, and prove that your algorithm is correct. Do **not** just list all possible assignments and check each individually; leverage off of algorithm *A* to get an answer directly.

Problem Seven: 3CNF (28 Points)

A boolean formula φ is in *3CNF* if it is in CNF, and each clause contains exactly three literals. For example, the following formula is in 3CNF:

$$(x_1 \lor \neg x_2 \lor \neg x_3) \land (x_1 \lor x_2 \lor \neg x_3) \land (x_4 \lor x_5 \lor x_6) \land (x_3 \lor x_4 \lor \neg x_5)$$

This formula is not in 3CNF, because it contains a clause containing four literals:

$$(x_1 \lor x_2 \lor x_3 \lor x_4) \land (x_5 \lor x_6 \lor x_7)$$

This formula is not in 3CNF, because it contains a clause containing just one literal:

$$x_1 \land (x_2 \lor x_3 \lor x_4)$$

Suppose that you are interested in converting a formula φ in CNF to an equisatisfiable formula φ' in 3CNF. To do so, you would need to come up with a way of converting clauses of any number of literals into equisatisfiable clauses of exactly three variables. In this problem, you will determine how to do exactly that.

In what follows, remember that to prove that φ ≅ ψ, you need to show that φ is satisfiable **if and only if** ψ is satisfiable. This means that your proofs will probably contain two parts, one for each direction of implication.

i. Prove that for any clause *c* containing exactly one literal, there is an 3CNF formula φ such that *c* ≅ φ. Some mathematicians insist that all of the literals in a clause be distinct, so you should not duplicate any literals in a clause.

ii. Prove that for any clause *c* containing exactly two literals, there is a 3CNF formula φ such that *c* ≅ φ. As with above, you should not duplicate any literals in a clause.

iii. Using (i) and (ii) as a starting point, prove by induction on the number of literals that for any clause *c* there is an equisatisfiable 3CNF formula ψ.

Your proof from (iii) can easily be extended to show that for any CNF formula φ, there is a 3CNF formula ψ such that φ ≅ ψ. Simply apply your result to each clause independently. The fact that any CNF formula can be converted to an equisatisfiable 3CNF formula is of great theoretical and practical importance. The problem of determining whether a 3CNF formula is satisfiable is called *3SAT*, which we'll visit later this quarter.

Problem Eight: Course Feedback (5 Points)

We want this course to be as good as it can be, and we'd really appreciate your feedback on how we're doing. For a free five points, please answer the following questions. We'll give you full credit no matter what you write (as long as you write something!), but we'd appreciate it if you're honest about how we're doing.

i. How hard did you find this problem set? How long did it take you to finish?

ii. Does that seem unreasonably difficult or time-consuming for a five-unit class?

iii. Were the checkpoints on the previous problem sets useful? Should we continue to use them in future quarters? Should we continue to have them on later problem sets?

iv. Did you attend Monday's problem session? If so, did you find it useful?

v. How is the pace of this course so far? Too slow? Too fast? Just right?

vi. Is there anything in particular we could do better? Is there anything in particular that you think we're doing well?

Submission instructions

There are three ways to submit this assignment:

1. Hand in a physical copy of your answers at the start of class. This is probably the easiest way to submit if you are on campus.
2. Submit a physical copy of your answers in the filing cabinet in the open space near the handout hangout in the Gates building. If you haven't been there before, it's right inside the entrance labeled “Stanford Engineering Venture Fund Laboratories.” There will be a clearly-labeled filing cabinet where you can submit your solutions.
3. Send an email with an electronic copy of your answers to the submission mailing list (cs103-spr1112-submissions@lists.stanford.edu) with the string “[PS4]” somewhere in the subject line.

If you are an SCPD student, we would strongly prefer that you submit solutions via email, especially for the checkpoint problems, so that we can get your solution graded and returned as quickly as possible. Please contact us if this will be a problem.

Extra Credit Problem: Oversimplification (5 Points Extra Credit)

In lecture, we saw that some propositional connectives can be rewritten in terms of other connectives. For example, $p \to q \equiv \neg p \lor q$, so we can rewrite any propositional logic formula so that it contains no instances of $\to$.

Prove that there exists a propositional logic formula φ containing at least one variable that cannot be rewritten as a formula using just the connectives $\neg$ and $\leftrightarrow$.