Recitation 2

Logic and Lean

SRC

This recitation will start off with an SRC activity led by the TAs! Here are some important links as you follow along:

- Slides [http://tinyurl.com/7tna54nb]
- Twitter Github (full) [http://tinyurl.com/3ry7neum]
- Twitter Github (current, censored) [http://tinyurl.com/49v2n5t4]

Review

In Recitation 1, we started discussing logical equivalences and propositions. Recall some of the terminology for logic, most of which we talked about last week:

1. A **proposition** is a statement that evaluates to true or false. For example, "grass is green" is a proposition.
2. A **propositional variable** is a symbol that represents a proposition. Propositional variables are assigned truth values ('true' or 'false'). For example, if we let p represent the proposition "grass is green," then p is a propositional variable.
3. A **propositional formula** can be constructed from atomic propositions via logical connectives. The truth value of a propositional formula can be calculated from the truth values of the atomic propositions it contains.
4. The term **logical expression** is often used synonymously with the word proposition.
5. Two propositions are **logically equivalent** when they have the same truth tables.
6. A proposition is **valid** if it evaluates to true on any choice of inputs; it is true no matter what. That is, a valid proposition is logically equivalent to the expression $(p \vee \neg p)$. This is also called a *tautology*.

7. A proposition is **satisfiable** if it evaluates to true on some choice of inputs. A valid proposition is satisfiable, but so are many propositions which sometimes evaluate to false.

8. If a proposition is **not satisfiable**, it evaluates to false on any choice of inputs; it is false no matter what. That is, it is logically equivalent to the expression $(p \wedge \neg p)$. This is called a *contradiction*.

Normal Forms

We say a proposition is in **DNF (disjunctive normal form)** when it is the disjuction (clauses ORed together) of conjuctions (terms ANDed together).

We say a proposition is in **CNF (conjunctive normal form)** when it is the conjuction (clauses ANDed together) of disjuctions (terms ORed together).

Here's a truth table, and propositions in DNF and CNF which represent it:

p	q	r	?
T	T	T	F
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	T

DNF: $(p \wedge q \wedge \neg r) \vee (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge \neg q \wedge r) \vee (\neg p \wedge \neg q \wedge \neg r)$

CNF: $(\neg p \vee \neg q \vee \neg r) \wedge (\neg p \vee q \vee \neg r) \wedge (p \vee \neg q \vee \neg r) \wedge (p \vee \neg q \vee r)$

If we have an arbitrary truth table, here are two ways we can think about describing it:

- Listing the true rows
- Listing the false rows

Since every row must be either true or false, both of these ways will uniquely describe our truth table.

These two ways correspond to DNF and CNF, respectively. To write a proposition in DNF, we can think about it like this: we find all rows where our proposition should evaluate to true, and we say that we must be in one of those rows. On the other hand, to write a proposition in CNF, we find all rows where our proposition should evaluate to false, and say we are not in any of those rows.

Task: How do we specify that we are in one of the true rows (DNF)? How do we specify that we are not in any of the false rows (CNF)?

Hint : Look at the DNF and CNF representations of the truth tables above. How do they relate to this idea?

Solution:

For DNF, we AND the true variables and negations of the false variables (to be in the row, the inputs must exactly correspond to the row). For CNF, we OR the false variables and the negations of the true variables (to not be in the row, we just need at least one variable to be different)

Task: Write two propositions corresponding to the following truth table: one in DNF and one in CNF.

p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

Solution:

DNF: $(p \wedge q \wedge r) \vee (p \wedge q \wedge \neg r) \vee (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge \neg r)$

CNF: $(\neg p \vee q \vee \neg r) \wedge (p \vee \neg q \vee r) \wedge (p \vee q \vee \neg r)$

Checkpoint 1 — call over a TA!

First-Order Logic

The language of propositional logic that we've used so far has been limited to atomic statements that are either true or false. But now let's consider, for instance, the following statement:

$$x \text{ is divisible by } 22$$

It makes sense if x is an integer, such as 44 or 50, but it doesn't make sense at all if x is a dinosaur named Fred. In any case, even when it does make sense, its truth value depends on x; indeed, '44 is divisible by 22' is a true proposition, but '44 is divisible by 50' is a false proposition.

To represent statements with variables like x, we introduce the notion of *predicates*. A **predicate** is a statement whose truth depends on the value of one or more variables. The **domain** of a predicate variable is the collection of all possible values that the variable may take.

Going back to the example above, 'x is divisible by 22' is not a proposition by itself, but it becomes a proposition when specific integers (such as 44 or 50) are substituted for x. In other words, we can represent the statement 'x is divisible by 22' as a predicate $p(x)$ such that x has a domain of $\mathbb{Z}$; then, $p(44)$ is the true proposition '44 is divisible by 22' and $p(50)$ is the false proposition '50 is divisible by 22'.

A **quantifier** is a symbol that denotes how many elements of a set make a statement true. There are two main quantifiers that we will focus on: universal quantifier $\forall$ and the existential quantifier $\exists$. Universal quantification asserts that a predicate is always true. Existential quantification asserts that a predicate is sometimes true. Formally, the expression '$\forall x : X$' denotes 'for all x in X' and '$\exists x : X$' denotes 'there exists x in X'.

Task

Consider the following sentence: "If a CS22 student can solve any problem from class, then they will get an A." Notice how this could have a double meaning? The phrase "a CS22 student can solve any problem from class" can be reasonably interpreted as either (1) the student can solve **all** problems from class or (2) the student can solve **at least one** problem from class.

Define variables and represent each of these interpretations in mathematical language. (Hint: One will start with the universal quantifier, the other will start with the existential quantifier.)

Solution:

S: the set of all CS22 students
P: the set of all CS22 problems

$F(x, y)$: "student x can solve problem y"
$A(x)$: "student x gets an A in the class"

1. $\forall x : S, (\exists y : P, F(x, y)) \rightarrow A(x)$
2. $\forall x : S, (\forall y : P, F(x, y)) \rightarrow A(x)$

Task

Using the variables you defined above, convert the following English sentences into formulas:

a. There exists a CS22 problem that is unsolvable.

b. There exists a CS22 student who cannot solve at least one CS22 problem.

c. If all CS22 students get an A, then there is some CS22 problem that can be solved by all CS22 students.

d. If all CS22 problems can be solved by at least one student, then at least one student will get an A.

Solution:

a. $\exists y : P, \forall x : S, \neg F(x, y)$

b. $\exists x : S, \exists y : P, \neg F(x, y)$

c. $(\forall x : S, A(x)) \rightarrow (\exists y : P, \forall x : S, F(x, y))$

d. $(\exists x : S, \forall y : P, F(x, y)) \rightarrow \exists x : S, A(x)$

Note: there can be more than one right answer.

Why First-Order Logic?

The language of propositional logic that we learned last week is not expressive enough to talk about real concepts in math or computer science. Consider the proposition, "every Python program runs without crashing." Not very plausible!

But, it *implies* the proposition "the Python program `print "Hello world!"` runs without crashing," for instance. With quantifiers and variables, we can now dive deeper into real concepts in math or computer science. For example, let P denote the set of Python programs, and let $R(x)$ mean "program x runs without crashing." Our sentences can then be respectively translated to $\forall x : P, R(x)$ and $R($`print "Hello world!"`$)$.

We could even *prove* the implication $\big(\forall x : P, R(x)\big) \to R($`print "Hello world!"`$)$.

Task

Let the following symbols be defined as:

- P: the set of Python programs. (You can use this as the domain for quantifiers.)
- *Predicate symbols* represent predicates with variable placeholders that can be filled in with any term (Recall that once a variable is assigned, it represents a proposition):
  - $R(x)$: "program x runs without crashing"
  - $T(x)$: "program x terminates" (i.e. doesn't run forever)
  - $C(x, y)$: "program x calls program y"
  - Familiar mathematical relations, like $<$, $\leq$, $=$
- *Function symbols* take in terms as arguments and output new terms:
  - $l(x)$: the number of lines of code in program x
- *Constant symbols* are terms:
  - 0, 1, 2, ... are constant symbols
  - *hw* is a constant symbol representing the "Hello world" program `print "Hello world!"`.
  - *mp* is a constant symbol representing the Python program I'm writing right now.

For a first example: we could translate the sentence "There is a program with fewer than 10 lines of code that does not terminate" to $\exists y : P, (l(y) < 10) \wedge \neg T(y)$.

Now, translate the sentences below using the symbols above.

a. Every program with fewer than 10 lines of code terminates.

Solution:

$\forall y : P, l(y) < 10 \rightarrow T(y)$

b. There is a program that doesn't crash but never terminates.

Solution:

$\exists y : P, R(y) \wedge \neg T(y)$

c. The "Hello world" program is the shortest program.

Solution:

$\forall y : P, l(y) \geq l(hw)$ or $\neg \exists y : P, l(y) < l(hw)$

d. The Python program I'm writing right now does not crash, and only calls programs that terminate.

Solution:

$R(mp) \wedge \forall y : P, C(mp, y) \rightarrow T(y)$

e. Every program calls another program.

Solution:

$\forall x : P, \exists y : P, C(x, y)$

f. Some program is called by every program. (What's the difference between this and the previous one?)

Solution:

$\exists y : P, \forall x : P, C(x, y)$

Task

Thought you were done? Try again! It's very common with these types of translations that there are multiple ways to translate any English sentence into logic. Go back

and see if you can find an alternate "phrasing" for two of your above answers that seems to capture the same meaning.

Can you see a general pattern in the way you rephrased your answer? Talk it over for a few. (If not, that's okay. We'll come back to this in class soon enough.)

🏴 Checkpoint 2 — call over a TA!

Lean

A real number q is defined to be *rational* if it is equal to the quotient of two integers that have no factors in common.

An integer k is divisible by d if there exists an integer m such that $k = m \cdot d$.

Theorem. $\sqrt{3}$ is not rational.

Proof.

1. Suppose, for the sake of contradiction, that $\sqrt{3}$ is rational. We wish to find a contradiction.
2. This means that $\sqrt{3}$ is equal to the quotient of two integers that have no factors in common.
3. Call these integers n and d; then $\sqrt{3} = \frac{n}{d}$, and n and d have no factors in common.
4. Squaring both sides, we know that $3 = \frac{n^2}{d^2}$.
5. Multiplying across, we know that $3d^2 = n^2$.
6. Since n^2 is divisible by 3, n must be divisible by 3.
7. Let n_1 be the integer such that $n = 3 \cdot n_1$.
8. By substitution, we have that $3d^2 = 9n_1^2$.
9. Cancelling 3, we have $d^2 = 3n_1^2$.
10. Thus d must be divisible by 3.
11. From steps 6 and 10, n and d have a factor of 3 in common.
12. This contradicts our assumption in step 2, completing our proof.

Done!

Task

a. What logic proof rules can you identify in this informal proof? (And intro, negation intro, ...?)

b. What Lean tactics do these correspond to? What steps in this proof do *not* have corresponding Lean tactics, from what we've seen so far in class?

c. After proof step 1, what does the context of our proof look like? What is the goal? You can write it in "fake Lean" notation: don't worry about the syntax, but show us an "infoview" ;)

d. After proof step 7, what does the context of our proof look like? What is the goal?

e. What about after step 11?

Solution:

a. This is a bit of a discussion question, without one specific right answer. Some students might see proof rules in other steps, which is fine – get them to try to justify what they see.

- Step 1 is a negation introduction. We're proving a "not," so assume it and show false.
- Step 3 is existential elimination! We're naming two variables bound by an existential quantifier.
- Step 6 sounds like a modus ponens (implication elimination). But it's invoking an "external" implication, one that we didn't actually state in this proof! The same modus ponens happens in step 10.
- Step 7 is also existential elimination. Divisibility is an "exists" statement in disguise.
- Step 12 is negation elimination. We have contradictory facts in our context, so we can prove anything.

b. The items above are `assume`, `eliminate`, `have`, `contradiction`. But there are a lot of `have`s scattered through this proof, as we add new facts to our context.

The substitution and cancelling steps certainly don't seem to be connected to anything we've seen so far.

c. $h : \sqrt{3}$ is rational

$\vdash$ False

d. Again, there's a lot of flexibility here, and students should discuss what they think "makes sense." Something like this is the target:

$n : \mathbb{Z}$

$d : \mathbb{Z}$

$h_1 : \sqrt{3} = \frac{n}{d}$

$h_2 : \neg$ (n and d have a factor in common)

$h_3 : 3d^2 = n^2$

$n_1 : \mathbb{Z}$

$h_4 : n = 3n_1$

$\vdash$ False

e. Something like the above, with the addition of

h_5 : n and d have a factor in common

🏁 Final checkoff — call over a TA!