CS 2742 (Logic in Computer Science) – Winter 2014
Lecture 2

Antonina Kolokolova

January 13, 2014

2.1 Logical connectives (continuing)

We will also use logical connectives $\leftarrow$ ($p \leftarrow q$ is "p only if q", where if q is true then p must also be true) and $\leftrightarrow$ meaning that p and q are either both true or both false, p if and only if q (sometimes pronounced as "p iff q"). Note that the last one has the meaning opposite to the "exclusive-OR" $\oplus$; we would not be using $\oplus$ much.

Instead of p and q we could have whole propositional formulas which are themselves made out of propositions, logical connectives and parentheses (to specify what logical connectives apply to). For example, if we denote "I am a dolphin" by a propositional variable r, then the following is a propositional formula: $(\neg p \vee q) \wedge r$, which reads as "It is not raining or it is cloudy and, besides, I am a dolphin". Here, the precedence order is first $\neg$, then $\wedge$, then $\vee$, and then $\rightarrow$: that is, $\neg q \rightarrow \neg p \vee q \wedge r$ is parenthesized as $(\neg q) \rightarrow ((\neg p) \vee (q \wedge r))$.

It may help to think of $\neg, \wedge$ and $\vee$ as unary $-$, $*$ and $+$ in arithmetic formulas: we will show in this course that there is a deep connection between them. So parenthesizing $\neg p \vee q \wedge r$ is just like parenthesizing $-5 + 3 * 8$.

Note that "or" in mathematical logic has a bit different meaning from English "either/or": here, both propositions can be true, whereas in English we often mean "or" as an exclusive: either the first one is true, or the second, but not both. For example, "it is raining or I am a dolphin" is a perfectly valid propositional formula which is true if either it is raining, or I am a dolphin, or both it is raining and I am a dolphin.

Similarly, the implication is only false if its left hand side (i.e., p) is true while the right hand side (q) is false. That is, "if it is raining then it is cloudy" is false only when it is raining out of blue sky. If it is not raining this propositional formula is true no matter whether it is cloudy or not.

One way to think of the implication $p \to q$ is to consider all possible scenarios for the values of p and q. If both p and q are true, then $p \to q$ is true. If p is true and q is false, then $p \to q$ is false. Finally, if p is false then $p \to q$ is true both when q is true and when q is false.

We will talk later that some of the logical connectives here are "redundant": we can express the same functionality as $p \to q$, for example, using just $\wedge$ and $\neg$. So $p \to q$ is false only when p is true and q is false, that is, when $p \wedge \neg q$ is true. Then saying $\neg(p \wedge \neg q)$ will be false when p is true and q is false, and true everywhere else, just like $p \to q$. As an example, think of saying "if it rains it must be cloudy" as having the same meaning as "it can't happen that both it's not cloudy and raining".

2.2 Truth tables

Recall the puzzle about twin brothers from last class. It turns out that the question that allows us to find out the name of the twin in front of us is "Is John lying?". Let us consider all possible situations:

1) Suppose that we met John and John tells the truth. Then he will truthfully reply "No".

2) Suppose that we met John and John lies. Then he will also say "No": "No, I am not a liar, it's the other guy!"

3) Now suppose that we met Jim and Jim tells the truth. Then he will truthfully reply "Yes", since the other brother is the liar.

4) Finally, suppose that we met Jim, and Jim is the liar. Then he will also say "Yes": "Yes, it is John who is the liar!"

So as you see if the answer was "No" then it must have been John and if the answer was "Yes" then it must have been Jim. Note that here we only learn what is the name of the brother, not whether he lies or not. Also, you can check that "Are you John?" does not give you an answer to this question.

We can write it as a table as follows. Let the propositional variable p denote "this is John", and the propositional variable q denote "John tells the truth".

p: This is John	**q: John tells the truth**	**Answer to "Is John lying?"**
True	True	False
True	False	False
False	True	True
False	False	True

This kind of a table is called a *truth table*: it contains all possible truth values for the pair of propositional variables p and q. This will be our tool for calculating truth values of propositional formulas. For example, we can define the logical connectives from the previous lecture using a truth table. We can also define formulas with values "always true" (a tautology) and "always false" (a contradiction); for example

p	q	$p \to q$ (p implies q)	$p \wedge q$ (p and q)	$p \vee q$ (p or q)	$\neg p$ (not p)	$p \wedge \neg p$ (contradiction)	$p \vee \neg p$ (tautology)
T	T	T	T	T	F	F	T
T	F	F	F	T	F	F	T
F	T	T	F	T	T	F	T
F	F	T	F	F	T	F	T

So you can see that the answer to "Is John lying?" is the negation of "This is John".

Definition 1 *A* truth assignment *(to variables in a formulas) is an assignment of values true/false to every variable.*

When values of all variables are known, it is possible to calculate the truth value *of the whole formula, according to rules for logical connectives from the table above.*

A truth assignment satisfies *a formula if the formula evaluates to true under this assignment. A formula is* satisfiable *if such a truth assignment exists. If a formula is not satisfiable by any truth assignment, we call it a* contradiction*. If a formula is true under all truth assignments, we call it a* tautology*.*

Sometimes, a truth assignment satisfying a formula is referred to as a model*. Usually, though, the word* model *is used in predicate logic setting.*

To be more formal, we would define a propositional formula by structural induction, that is, by saying that a proposition is a propositional formula, negation of a formula is a formula, and two formulas connected by $\vee, \wedge, \to$ or other logical connective is also a formula. That immediately gives us a way to evaluate propositional formulas starting from the propositions and evaluating the truth values of the connectives according to the rules in the table above.

Example 1 (It is either not raining, or it is cloudy), and today is Monday.
Let's set p: "it is raining", q: "it is cloudy" and r: "today is Monday". Then (setting the parentheses as above) the formula can be written as $(\neg p \vee q) \wedge r$. Today is Monday, it is cloudy but not raining. So p is false, q is true and r is true. Then $\neg p$ is true, making the $\vee$ true. And r is also true, making the $\wedge$ true. The whole formula is, thus, true. Now suppose it is raining, not cloudy and today is Monday ($p = T, q = F, r = T$). In this case, both $\neg p$ and q are false, so $\neg p \vee q$ is false, and therefore the whole formula is false. Thus, this formula is neither a tautology nor a contradiction.

How to evaluate logic formulas? Similar to arithmetic formulas, with $\neg$ being a $-$ as in -5, $\wedge$ a $\times$ and $\vee$ a $+$. Precedence rules: $\neg$ over $\wedge$, $\wedge$ over $\vee$, $\vee$ over $\rightarrow$. Otherwise, use parentheses. For example, in above formula $(\neg p \vee q) \wedge r$ it is OK not to put parentheses around $\neg p$ since it has higher precedence than $\vee$,

2.3 Showing Satisfiability vs. proving tautologies and contradictions.

Truth tables allow us to check if a given formula is a tautology, contradiction or is satisfiable, by trying all possible truth assignments (how many truth assignments? $2^{\text{number of variables}}$). However, it is not a very efficient method, since checking a formula with even a thousand variables (in software verification this is a reasonable number) would require a truth table larger than the size of the universe!

Although there are some methods for checking if a formula is satisfiable, or is not a tautology, none that we know can give us an answer significantly faster then the truth tables. However, we don't know if this is an inherent limitation or whether we haven't come up with a smart enough algorithm yet. This is essentially the statement of a major open problem in mathematics, usually stated as "P vs. NP" problem (one of 7 mathematical problems for the new millennium defined by Clay institute – which offers a million of dollars for solving each one). More precisely, checking if a given formula is satisfiable (that is, not a contradiction) is a classical – moreover, the very first – example of an "NP complete" problem; the main part of the "P vs. NP" question, as it is usually stated, is finding out whether NP-complete problems are inherently hard or if there is a polynomial-time algorithm for them (such algorithm for one of them, for example satisfiability, would efficiently solve all NP-complete problems). You will see much more about this in COMP 3719.

Example 2 Let's look again at $(\neg p \vee q) \wedge r$ and compute its truth table. We already calculated two lines of the table: the $p = F, q = T, r = T$ line and $p = T, q = F, r = T$ line.

p	q	r	$\neg p$	$(\neg p \vee q)$	$(\neg p \vee q) \wedge r$
T	T	T	F	T	T
T	T	F	F	T	F
T	F	T	F	F	F
T	F	F	F	F	F
F	T	T	T	T	T
F	T	F	T	T	F
F	F	T	T	T	T
F	F	F	T	T	F

Puzzle 2 (Knights and knaves 1) Some remote island is populated by two kinds of people: knights, who always tell the truth, and knaves, who always lie. Suppose you met two

islanders, call them A and B, and you hear A saying "at least one of us is a knave". Can you tell which of A and B is a knight and which is a knave?