Logic in Computer Science
Chapter 1

Antonina Kolokolova*

Jan 10, 2014

1.1 What is logic in Computer Science?

Why do we study mathematical logic? Natural languages (such as English) are too ambiguous:

> "Every student knows this" and "Any student knows this" have the same meaning.
>
> "She'd be happy if she passes every course" and "She'd be happy if she passes any course" mean very different things.

Mathematical logic is a kind of language, where everything is designed to have a precise meaning. This is the language of unambiguous reasoning. So this course is somewhat like a foreign language course: there will be a lot of vocabulary to memorize. Also, like in a language course, you need to practice "speaking" logic language for it to become natural.

Logic comes into computer science in many different ways:

- Digital circuit logic
- Proof of correctness of programs, verification
- Automated reasoning in artificial intelligence

*The material in this set of notes came from many sources, in particular "Discrete mathematics with applications" by Susanne Epp, several books by Raymond Smallian, course notes of U. of Toronto CS 238 by Vassos Hadzilacos.

- Database query languages
- ...

We will try to cover, briefly, at least some of these applications.

1.2 Propositional logic

The basic unit of our reasoning is a sentence that can have a truth value, that is it can be either true or false. We call such a sentence a *proposition*. For example, "it is raining" is a proposition. It is true or false at any particular point in time and space. So is "I am a dolphin" (which happens to be false, as far as I know), or "2 is a prime number". When referring to a proposition, we often will give it a short name (a variable); for example, we can use a variable p to denote the sentence "it is raining". Now, if we say "p is true" we will mean that it is, indeed, raining. Such variables denoting propositions are called *propositional variables*. A propositional variable can have a *truth value* of exactly one of "true" or "false".

Simple propositions can be combined. For example, we can say "if it is raining, then it must be cloudy". Here, we mean that "raining" (that is, our proposition p) implies that it is also "cloudy". If we use a propositional variable q to mean "it is cloudy" then in the language of propositional logic we say that "p implies q". Implication is one of *logical connectives* that allow us to make more complicated statements, *propositional formulas*, out of propositions. The following table lists several common logical connectives.

Meaning	**Name**	**Notation**	**Pronunciation**
both p and q are true	*conjunction*	$p \wedge q$	p and q
at least one of p, q is true	*disjunction*	$p \vee q$	p or q
opposite of p is true	*negation*	$\neg p$	not p
if p is true then q is true	*implication*	$p \rightarrow q$	p implies q

Puzzle 1 (Twins puzzle) There are two identical twins, John and Jim. One of them always lies, another always says the truth (that is, every sentence the truth-teller says is true; every sentence the liar utters is false). Suppose you run into one of them and want to find out whether you met John or Jim. Which 3-word question with a yes/no answer could you ask to learn the name of the twin in front of you? You don't know which one of them is the liar.

Chapter 2

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 $p \rightarrow q$ is to consider all possible scenarios for the values of p and q. If both p and q are true, then $p \rightarrow q$ is true. If p is true and q is false, then $p \rightarrow q$ is false. Finally, if p is false then $p \rightarrow q$ 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 $\to$. 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?

Chapter 3

3.1 Logical equivalences

Recall the puzzle from the previous class: on some island, there are knights (who always tell the truth) and knaves (who always lie). You meet two islanders (call them Paul and Steve) and hear Paul say "at least one of us is a knave". Can you tell whether the islanders are knights or knaves and which islander is which?

We solve this puzzle using a truth table. Take p :"Paul is a knight" and q : "Steve is a knight.". Then the sentence "At least one of us is a knave" is translated as $(\neg p \vee \neg q)$, since being a knave is the negation of being a knight. Now, we want to know when the truth value of this sentence $(\neg p \vee \neg q)$ is the same of the truth value of p: that is, if Paul is a knight, then what he said must be true, and if Paul is a knave, then what he said must be false. This we can state as Paul is a knight if and only if "at least one of us is a knave" is true. We represent this as a truth table as follows:

p	q	$(\neg p \vee \neg q)$	$p \leftrightarrow (\neg p \vee \neg q)$
T	T	F	F
T	F	T	T
F	T	T	F
F	F	T	F

As you can see, the only scenario when what Paul says corresponds correctly with Paul's being a knight/knave is the second line: that is, when Paul is a knight and Steve is a knave. Let us introduce the notation $\iff$ to mean *logical equivalence* (that is, two formulas having the same truth values for any truth assignment to their variables). A better way way of stating the last condition is as a logical equivalence of p and $f(\neg p \vee \neg q)$, that is, $p \iff (\neg p \vee \neg q)$.

Definition 1 *We say that two formulas are* logically equivalent *($A \iff B$) if they have the same truth value under any truth assignment.*

That is, in a truth table the columns of logically equivalent formulas are identical.

Theorem 1 *Two formulas A and B are logically equivalent if and only if a formula $A \leftrightarrow B$ is a tautology.*

For the proof, Check that the columns for A and B are the same if $A \leftrightarrow B$ is a tautology, and different if it is not. Although A and B are formulas here, this still can be checked with just a 4-line truth table.

A useful property of logically equivalent formulas, called *substitution*, is that if in any formula you replace a subformula by another that is logically equivalent to it, then the value of the whole formula would not change. For example, $p \wedge (q \vee \neg q)$ is logically equivalent to $(p \wedge T)$, which is in turn equivalent to p (can check this using a truth table). So if in a formula there is an occurrence of $p \wedge (q \vee \neg q)$ it can be safely replaced with just p, thus simplifying the formula. Another example that you see quite often is substituting $(p \to q)$ with $(\neg p \vee q)$, so, for example, $r \wedge (p \to q) \vee \neg p$ is logically equivalent to $r \wedge (\neg p \vee q) \vee \neg p$.

3.2 Logical identities

Now that we have a notion of logical equivalences we can talk about a few identities in propositional logic. We will list them as pairs of equivalent formulas.

Name	$\wedge$-version	$\vee$-version
Double negation	$\neg\neg p \iff p$	
DeMorgan's laws	$\neg(p \wedge q) \iff (\neg p \vee \neg q)$	$\neg(p \vee q) \iff (\neg p \wedge \neg q)$
Commutativity	$(p \wedge q) \iff (q \wedge p)$	$(p \vee q) \iff (q \vee p)$
Associativity	$(p \wedge (q \wedge r)) \iff ((p \wedge q) \wedge r)$	$(p \vee (q \vee r)) \iff ((p \vee q) \vee r)$
Distributivity	$p \wedge (q \vee r) \iff (p \wedge q) \vee (p \wedge r)$	$p \vee (q \wedge r) \iff (p \vee q) \wedge (p \vee r)$
Identity	$p \wedge T \iff p$	$p \vee F \iff p$
	$p \wedge F \iff F$	$p \vee T \iff T$
Idempotence	$p \wedge p \iff p$	$p \vee p \iff p$
Absorption	$p \wedge (p \vee q) \iff p$	$p \vee (p \wedge q) \iff p$

Notice again (as working our way to boolean algebras) that many of these identities behave just like algebraic and arithmetic identities, with $\wedge$ behaving like $\times$, $\vee$ like $+$, T like 1 and F like 0. For example, commutativity and associativity laws are the same as for numbers: $(3+2)+5 = 3+(2+5)$. One notable exception is that with numbers there is just one form of distributed law, namely the $\wedge$ form $(a \times (b+c) = (a \times b) + (a \times c)$, and the $\vee$ form does not hold $(a + b \times c) \neq (a \times b) + (a \times c))$, whereas in logic both forms are true.

It is convenient to treat $\to$ and $\leftrightarrow$ as "syntactic sugar", and define them to be their equivalent formulas with just $\vee, \wedge, \neg$. That is, we will say that $(p \to q) \iff (\neg p \vee q)$ by definition, and so is $(p \leftrightarrow q) \iff (\neg p \vee q) \wedge (\neg q \vee p)$.

Puzzle 3 (Wason Selection Task) You see 4 cards on the table; all cards have a letter on one side and a number on the other. On those four cards you see written "A", "D", "4" and "7". Which cards do you need to turn over to verify the statement "If a card has a vowel on one side, then it has an even number on the other"?