CS 2742 (Logic in Computer Science)
Lecture 7

Antonina Kolokolova

January 31, 2014

3 Normal forms of propositional formulas.

Any formula has an equivalent one in a normal form. In computer science, the two normal forms we are interested in are *conjunctive normal form (CNF)* and *disjunctive normal form (DNF)*. The first one is a $\wedge$ of $\vee$s, where $\vee$ is over variables or their negations (*literals*), the second is a $\vee$ of $\wedge$s of literals. A $\vee$ of literals is also called a *clause*, and a $\wedge$ of literals a *term*.

Example 1.

- $\neg p$, x, s are examples of literals, whereas $\neg\neg p$ or $(x \vee y)$ is not a literal
- $(x \vee \neg y \vee z)$, $(\neg p)$ are clauses,
- $(x \wedge \neg y \wedge z)$, $(\neg p)$ are terms.
- $(x \vee \neg z \vee y) \wedge (\neg x \vee \neg y) \wedge (\neg y)$ is a CNF.
- $(x \wedge z) \vee (\neg y \wedge z \wedge x) \vee (\neg x \wedge z)$ is a DNF.
- $(x \wedge \neg(y \vee z) \vee u)$ is neither a CNF nor a DNF, but it is equivalent to the following DNF: $(x \wedge \neg y \wedge \neg z) \vee u$.

Why do we need CNFs and DNFs? Other than the fact that they are convenient normal forms, they are very useful for constricting formulas given a truth table. In particular, DNFs can be constructed from truth table by taking a disjunction (that is, $\vee$) of all satisfying truth assignments and CNFs by taking a conjunction ($\wedge$) of negations of falsifying truth assignments

Example 2. Recall the truth table for $(p \to q) \wedge q \to p$. Suppose that you are given just the first two and the last column. How do you construct a formula that would have such a truth table?

Let us start by constructing a DNF formula. It would say, basically, "either the first line satisfies me, or the second, or the last". We write it as $(p \wedge q) \vee (p \wedge \neg q) \vee (\neg p \wedge \neg q)$. Each term here fully describes one satisfying assignment, and the whole formula is true iff $(p \to q) \wedge q \to p$ is. Now, to construct a formula in CNF we make it say "I am satisfied by neither of the following assignments". One way to write it is as a negation of a DNF formula listing all falsifying assignments. In this example there is just one falsifying assignment, so the formula becomes $\neg(\neg p \wedge q)$. By DeMorgan's law, this is equivalent to $(\neg\neg p \vee \neg q)$, and finally $(p \vee \neg q)$. If there were more than one falsifying assignment, here we would have several of clauses with $\wedge$ between them.

This construction also shows us that for any formula there is an equivalent CNF and an equivalent DNF formula (these are the formulas constructed from the truth table of the original formula). In particular, if the formula is a tautology, its DNF and CNF consist of just $(p \vee \neg p)$ for an arbitrary p, and if it is a contradiction, of $(p \wedge \neg p)$.

Let us do a larger example of constructing a CNF and DNF from a truth table.

A crucial point is that every truth assignment can be encoded as a propositional formula which is true just on that assignment and false everywhere else. In order to make such a formula, take $\wedge$ of all variables that are true in the truth assignment and negations of all variables that are false in that truth assignment. Since every variable is either true or false, every variable will occur exactly once in this conjunction, either positively or negatively. For example, a truth assignment $p = T, q = F, r = T$ can be represented by a formula $p \wedge \neg q \wedge r$. This formula will be true when $p = T, q = F$ and $r = T$ and false everywhere else.

This representation of truth assignments by formulas is our building block for constructing DNFs and CNFs from a truth table. A DNF says that one of the truth assignments listed is true, where the assignments listed are all the satisfying assignments of a formula. A CNF is just a negation of a DNF formula listing falsifying truth assignments, simplified using DeMorgan's law to place negations on variables.

Let us look at a larger example for constructing CNFs and DNFs, equivalent to formulas (given by truth tables). That is, the CNF,the DNF and original formula, although they look quite different, share the same set of variables and are satisfied by exactly the same truth assignments. Their truth tables are identical.

Example 3. Last time we used a truth table of a propositional formula $(p \to q) \wedge q \to p$ for constructing DNFs and CNFs. Now let us add one more variable: consider a formula $((p \to q) \wedge q \to p) \wedge (\neg q \vee \neg r)$.

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

There are three F in the last column and five T, so our CNF will have 3 clauses (that is, it will be the $\wedge$ of three $\vee$), and our DNF will have 5 terms.

So, to construct a CNF for this formula, take

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

After simplification (applying DeMorgan's law and double negation law), this becomes

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

You can check yourself that this formula is CNF and is indeed equivalent to $((p \to q) \wedge q \to p) \wedge (\neg q \vee \neg r)$; that is, it has the same truth table. A DNF for this formula is:

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

Note that between the CNF and DNF all 8 truth assignments are mentioned: the falsifying ones in the CNF and satisfying in DNF.

How large would such CNF or DNF be? Well, potentially nearly as large as the truth table itself (say half of the entries are false, and half are true). But does it have to be so, or are there always smaller formulas? If a column is filled with T and F pretty much randomly, it is likely not to have a smaller description than the column itself, and so any formula encoding it would have to be nearly as large as a truth table itself. In a few lectures we will start talking about (classes of) Boolean functions, and describing them by sequences (one for each number of variables) of truth tables. In that case, the question would be if such a Boolean function can, as $n \to \infty$, be described by formulas asymtotically smaller than their truth table. In that case, although we know that a random such Boolean function would not have small descriptions, and we also know functions for which there are no small CNF/DNF, the general question of finding a natural example of a "hard" (that is, with no small formula) Boolean function is an open problem.

Recall that a formula is in the CNF (conjunctive normal form) if it is a $\wedge$ of $\vee$s of literals (variables or their negation.)

In this lecture we will talk about proving (or, actually, finding contradictions) statements that are in this special form.

Definition 1 (Resolution rule)**.** *: Given two clauses of the form $C \vee x$) and $(D \vee \neg x)$, where C and D are (possibly empty) disjunction of variables, can derive a (possibly empty) clause $(C \vee D)$.*

That is,

$$(C \vee x) \wedge (D \vee \neg x) \rightarrow (C \vee D)$$

where $C = (l_1 \vee \cdots \vee l_k)$ and $D = (l'_1 \ldots l'_{k'})$ for some literals. If there is a repeated literal, only write it once.

Note that you may end up deriving an "empty" clause, that is, a $\vee$ of zero literals. But the only way this could happen is when two clauses being resolved are (x) and $(\neg x)$. But $x \wedge \neg x$ is always false, a contradiction. So deriving an empty clause proves that the original formula was a contradiction (which is what we usually want to show).

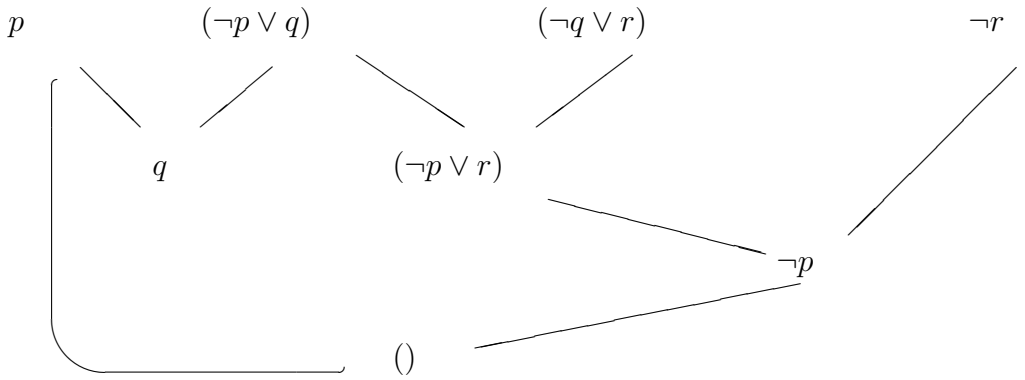
Example 4. Consider the following statements:
p: it is sunny
q: the weather is good
r: I spend the day outside

Now consider the following argument:

p
$p \rightarrow q$
$q \rightarrow r$
$\therefore r$

We want to prove that this is a valid argument, that is, p, $p \rightarrow q$ and $q \rightarrow r$ together imply r (this in general is called a *transitivity* law). Resolution proof method allows us to find contradictions: so we represent this problem as a contradiction $p \wedge (p \rightarrow q) \wedge (q \rightarrow r) \wedge \neg r$. Note that here again we are using the fact that the negation of an implication (in this case, premises imply the conclusion) is a conjunction of premises and negation of the conclusion.

Resolution works with CNF formula, so the first step is to convert the formula above into CNF. In this case, it is very easy: just apply the definition of implication to the second and third clause to obtain $p \wedge (\neg p \vee q) \wedge (\neg q \vee r) \wedge \neg r$.



Puzzle 6. Prove that in a big city like Toronto with more than a million people there would be at least two people with the same number of hairs on their heads. Assume that the number of hairs on somebody's head cannot be more than $300,000$.