CISC 204   Class 10

Semantic Equivalence, Satisfiability, and de Morgan's Laws

Text Correspondence: pp. 45–57

Main Concepts:

- *Syntactically equivalent: two formulas deducable from each other*
- *Syntactically equivalent: two formulas that semantically entail each other*
- *Satisfiable formula: has at least one model that assigns it to "true"*
- *A satisfiable formula is the negation of a valid formula*
- *de Morgan's Laws are semaantic equivalences*

When we were proving sequents we sometimes had two formulas, ϕ and ψ, where $\phi \vdash \psi$ and $\psi \vdash \phi$. We described these formulas as *syntactically equivalent*, because from one formula we could deduce the other.

10.1 Semantic Equivalence in Propositional Logic

We can now write a semantic version of this important previous syntactic result, where we describe the *semantic equivalence* of two formulas.

Definition: Semantically equivalent

Let ϕ and ψ be formulas of propositional logic. We say that ϕ and ψ are *semantically equivalent*, written $\phi \equiv \psi$, if and only if $\phi \models \psi$ and $\psi \models \phi$.

This extends our previous concept of provable equivalence, which was written $\phi \dashv\vdash \psi$. We can now confidently assert that two formulas that are syntactically equivalent, which means that there are valid proofs from one to the other, are also semantically equivalent, which means that their truth tables are the same.

Which of the following equivalences hold?

$$\begin{aligned} p \to q &\overset{?}{\equiv} \neg q \to \neg p \\ p \to q &\overset{?}{\equiv} \neg p \lor q \\ p \to (q \to r) &\overset{?}{\equiv} (p \to q) \to r \end{aligned}$$

This leads us to an important related term, the *validity* of a formula.

Definition: Valid

We say that, for a formula ϕ, if the semantic entailment $\models \phi$ holds then ϕ is *valid*.

We introduced this concept earlier, as a tautology. A tautological formula and a valid formula are different terms for the same thing: a formula that evaluates as true for all models of the propositions within it.

10.2 Satisfiability in Propositional Logic

Previously, we introduced the idea of a valid formula. Being computer scientists, we seek a way to automatically check for validity of a formula. This would be made easier if we had a minimal notation for propositional logic. One path to this goal is to try to avoid the use of implication "$\to$". Because we have a semantic equivalence

$$\phi \to \psi \equiv \neg\,\psi \lor \psi$$

we can replace every occurrence of implication with negation and disjunction. These, plus conjunction, will suffice.

Thinking now about efficiency of computation as well as about automation, we can see that if a formula contain k atomic propositions then its truth table will have 2^k rows.

Can we do better? That is, can we "gather" terms together more uniformly that will give us a more parsimonious representation? The answer is that we can, and there are two useful ways of doing this. These are called *normal forms*, derived from the meaning of "norm" as a standard that must be met.

We already know that disjunction and conjunction are associative, so we can omit the internal parentheses without changing the semantics of a formula. For disjunction, using formulas ϕ and ψ and χ, the semantic equivalence is

$$(\phi \vee (\psi \vee \chi)) \equiv ((\phi \vee \psi) \vee \chi) \quad \text{so we abbreviate} \quad (\phi \vee \psi \vee \chi)$$

For conjunction, the semantic equivalence is

$$(\phi \wedge (\psi \wedge \chi)) \equiv ((\phi \wedge \psi) \wedge \chi) \quad \text{so we abbreviate} \quad (\phi \wedge \psi \wedge \chi)$$

The normal form we will consider is *Conjunctive Normal Form*, or CNF. There are many other forms but we will not need them in this course.

Definition: Satisfiable

Given a formula ϕ in propositional logic, if there is some valuation of ϕ that evaluates to $\mathbf{T}$ then we say that ϕ is *satisfiable*.

The two key uses of normal forms are determine whether a formula is *valid* and whether it is *satisfiable*. The one we will look at, CNF, is composed of clauses such that *every* clause must be valid for the formula to be valid.

We now understand that a formula is satisfiable if and only if the formula is true for *some* valuation. A formula is valid if and only if the formula is true for *every* valuation. If we were using a truth table, we would say that a formula is satisfiable if and only if the formula was true for *some* row, and a formula is valid if and only if the formula was true for *every* row.

This leads us to a powerful theorem in propositional semantics.

Theorem: Satisfiability and Validity

Let ϕ be a formula of propositional logic. The formula ϕ is satisfiable if and only if $\neg\,\phi$ is not valid.

Proof Outline: We can prove this by assuming ϕ is satisfiable and showing that $\neg\,\phi$ is not valid, then by assuming $\neg\,\phi$ is not valid and showing that ϕ is satisfiable. (Proof details are in the text.)

One way to use this theorem is to "read out" a truth table of a formula into CNF. To better understand the concept, let us use one form of de Morgan's Laws as an example.

10.3 de Morgan's Laws

We can understand satisfiability more deeply if we look carefully at de Morgan's Laws. Recall that one variant of the rule is

$$\neg(\psi \land \chi) \equiv \neg\,\psi \lor \neg\,\chi$$

Consider the truth table

p	q	ϕ
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

We seek a concise formula for ϕ in terms of p and q. One way to get such a formula is to examine the row in which ϕ evaluates to **F**. For this line,

$$\neg\,\phi \equiv \neg\,p \land \neg\,q$$

Using one of de Morgan's Laws, we can write $\phi \equiv \neg\neg\,\phi$ as

$$\begin{aligned} \phi &\equiv \neg(\neg\,p \land \neg\,q) \\ &\equiv (\neg\neg\,p \lor \neg\neg\,q) \\ &\equiv (p \lor q) \end{aligned}$$

Next, consider the truth table

p	q	ϕ
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

The lesson from the first truth table is: wherever $\neg\,\phi$ is not valid, ϕ is satisfiable. Here, $\neg\,\phi$ is not valid in the first and third rows; we can say that it is not valid if the first row is satisfiable *or* if the third row is satisfiable. We could write

$$\neg\,\phi \;\equiv\; (p \land q) \lor (\neg\,p \land \neg\,q)$$

A first application of one of de Morgan's Laws produces

$$\phi \equiv \neg(p \wedge q) \wedge \neg(\neg p \wedge \neg q)$$

A second application of one of de Morgan's Laws, to each of the disjuncts, produces

$$\begin{aligned} \phi &\equiv (\neg p \vee \neg q) \wedge (p \vee q) \\ &\equiv D_1 \wedge D_2 \end{aligned}$$

The second application of one of de Morgan's Laws has expressed ϕ as the conjunction of simpler clauses, ones with only disjunction or negation. We can use these simple clauses to quickly and automatically verify the expression using the truth table. The process is:

```
    For each row $i$,
    Is $D_1$ true *and* is $D_2$ true
```

Going through the truth table, we see that the results are:

Row 1:	D_1 is **F** and D_2 is **T** so ϕ is **F**
Row 2:	D_1 is **T** and D_2 is **T** so ϕ is **T**
Row 3:	D_1 is **T** and D_2 is **T** so ϕ is **T**
Row 4:	D_1 is **T** and D_2 is **F** so ϕ is **F**

We have shown by example that when a formula is equivalent to a conjunction, then the formula is valid provided that *every* conjunct is satisfiable. In the next class we will formalize this idea so that we can convert any formula into a conjunction.