Maggie Johnson
CS109

Propositional Logic

Key Topics

* Quick Review
* Conditional and Biconditional
* Logical Equivalence
* Brief Diversion: Satisfiability
* Rules of Inference
* Common Fallacies

Quick Review

- A **proposition** is a statement that is either true or false.
- The goal of propositional logic is to study ways of combining propositions to form new ones, and then to determine under what circumstances these "compound" propositions are true.
- A **compound proposition** consists of 2 or more propositions connected by the **logical connectives** AND, or OR. We also include in the definition of compound propositions a single proposition preceded by the logical modifier NOT.

Truth table for AND (^):

p	q	p^q	
T	T	T	
T	F	F	(conjunction)
F	T	F	
F	F	F	

Truth table for OR (v):

p	q	pvq	
T	T	T	
T	F	T	
F	T	T	(disjunction)
F	F	F	

Truth table for NOT (~)

p	~p	
T	F	
F	T	(negation)

- One way of analyzing the truth or falsity of a complex (more than 1 connective) proposition is through the use of **truth tables**.

- Precedence in compound propositions: negation is done first, then ^ and v from left to right
- Compound propositions that are always true are called **tautologies**. Compound propositions that are always false are called **contradictions**.

Conditional and Biconditional

The conditional connective is interpreted as "if p then q".

p	q	p->q
T	T	T
T	F	F
F	T	T
F	F	T

conditional truth table

Many theorems in math have a conditional form (e.g., if x = y and y = z, then x = z). The if part of a conditional is called the **hypothesis** and the then part is called the **conclusion**.

One thing to remember: In propositional logic, a hypothesis and a conclusion do not have to be related in subject matter. Thus, if 3+3 = 28, then Pongo is president of the United States is a true statement regardless of the fact that it makes absolutely no sense.

Some variations to be aware of:

1) ~q -> ~p is called the **contrapositive** of p -> q
2) q -> p is the **converse** of p -> q
3) ~p -> ~q is the **inverse** of p -> q

`Are any of these equivalent to p -> q?  Are any of these equivalent to each other?  How would you show this?`

The reason these other forms are important is they can be helpful when you are trying to solve certain problems. You may find the contrapositive form of a conditional statement is easier to work with than the original statement, or vice versa. Here is an example of a contrapositive:

If tomorrow is not Monday, then today is not Easter.

Not too tough to understand, but the corresponding conditional is even easier:

If today is Easter, then tomorrow is Monday.

One other important thing to remember: Many people think that if a conditional statement is true, then its converse and inverse are also true. The converse and inverse may or may not be true. For example, check out the converse and inverse of: If today is Easter, then tomorrow is Monday.

The **biconditional** is another connective denoted by the symbol <->. Note that the biconditional p <-> q is true when both p -> q is true and q -> p is true. That is why the terminology "if and only if" is used:

p	q	p<->q
T	T	T
T	F	F
F	T	F
F	F	T

biconditional truth table

What happens if we form the biconditional of a statement and its contrapositive?

Note that sometimes in logical arguments, we see the term “only if”. This is not the same as the biconditional. “p only if q” means p can take place only if q takes place also. That is, if q does not take place, then neither does p (if ~q, then ~p).

Logical Equivalence

Two propositions are **logically equivalent** if they have exactly the same truth values under all circumstances. For example, if we were to construct a truth table for ~ (p ^ r) v [(p ^ q) ^ ~ r], we would see that the last evaluated column is exactly the same as the last evaluated column for [(p ^ q) v r] ^ [~ (p ^ r)]. If two statements are logically equivalent, you can substitute one for the other.

· Some logical equivalencies which will be useful in proofs (note: <=> means "is equivalent to")

Identity Laws	p ^ T <=> p
	p v F <=> p
Domination Laws	p v T <=> T
	p ^ F <=> F
Idempotent Laws	p v p <=> p
	p ^ p <=> p
Double Negation	~(~p) <=> p
Commutative Laws	p v q <=> q v p
	p ^ q <=> q ^ p
Associative Laws	(p v q) v r <=> p v (q v r)
	(p ^ q) ^ r <=> p ^ (q ^ r)
Distributive Laws	p v (q ^ r) <=> (p v q) ^ (p v r)
	p ^ (q v r) <=> (p ^ q) v (p ^ r)
DeMorgan's Laws	~(p ^ q) <=> ~p v ~q
	~(p v q) <=> ~p ^ ~q
Other Useful Equivalences	p v ~p <=> T
	p ^ ~p <=> F
	(p -> q) <=> (~p v q)
	(p -> q) <=> (~q -> ~p)
	(p -> q) <=> ~(p ^ ~q)
	(p <-> q) <=> (p -> q) ^ (q -> p)
	p v q <=> ~p -> q
	~p -> F <=> p

Review Question:

“If Fred has a giant nose, then he also has a giant toe.” If this statement is true, which of the following are also true?

___ If Fred has a giant toe, then Fred has a giant nose.
___ If Fred does not have a giant toe, then Fred does not have a giant nose.
___ If Fred does not have a giant nose, then Fred does not have a giant toe.

Brief Diversion: Satisfiability

In studying algorithms in 106B or 106X, you saw that functions and big-O notation could be used to characterize the running time of algorithms. There is an important category of problems that we will study later in the quarter that have the following characteristic:

> The problem can be solved by an algorithm in exponential time, but there *may* be a solution that runs in polynomial time. The exponential solution is easy to define, but no one has been able to find the polynomial solution.

The reason we bring all this up now is one of the most important problems in this category is called the **Satisfiability problem**. The problem is to determine whether any given logical formula is satisfiable, i.e., whether there exists a way of assigning TRUE and FALSE to its variables so that the result is TRUE.

The problem is usually defined formally as follows. Given a formula

- composed of variables a, b, c, ... and their logical complements, ~a, ~b, ~c...
- represented as a series of clauses, in which each clause is the logical OR (v) of variables and their logical complements
- expressed as the logical AND (^) of the clauses

Is there a way to assign values to the variables so that the value of the formula is TRUE? If there exists such an assignment, the formula is said to be satisfiable.

```
Is: (a) ^ (b v c) ^ (~c v ~a) satisfiable? ________________________
Is: (a) ^ (b v c) ^ (~c v ~a) ^ (~b) satisfiable? __________________
```

```
How would you define the exponential solution to an instance of
this problem?
```

Rules of Inference

A **theorem** is a statement that can be shown to be true. We demonstrate that a theorem is true with a sequence of statements called a **proof**. The statements used in a proof include **axioms** or **postulates** (things we know or assume are true), statements implied by the statements that precede it, or previously proven theorems. The **rules of inference**, which are the means used to draw conclusions, tie together the steps of a proof.

The tautology (p ^ (p -> q)) -> q is the basis of a rule of inference called **modus ponens.** This rule says that whenever a statement of the form "if p, then q" is true and at the same time p is also true, then q is also true. (Recall the truth table for implication.) The notation used for rules of inference expresses modus ponens in this way: p, p -> q |- q

To understand the process of proof, we must understand the rules of inference which we are allowed to use. In propositional logic, it is easy to construct and test rules of inference. The basic rule is that p1, p2, ..., pn |- q is a rule of inference if and only if p1^p2^...^pn -> q is a tautology.

Commonly used rules of inference:

<table>
<tr><td>p, p->q |- q</td><td>modus ponens</td></tr>
<tr><td>p->q, q->r |- p->r</td><td>syllogism</td></tr>
<tr><td>p->q, ~q |- ~p</td><td>modus tollens</td></tr>
<tr><td>p |- p v q</td><td>addition</td></tr>
<tr><td>p ^ q |- p</td><td>specialization</td></tr>
<tr><td>p, q |- p ^ q</td><td>conjunction</td></tr>
<tr><td>p v q, ~p |- q</td><td>disjunctive syllogism</td></tr>
</table>

As an example, we will show that modus tollens is a valid rule of inference. To verify this, we need to show that [(p -> q) ^ ~q] -> ~p. The following truth table proves that this is a tautology:

								*		
p	q	p	->	q	^	~	q	->	~	p
T	T	T	T	T	F	F	T	T	F	T
T	F	T	F	F	F	T	F	T	F	T
F	T	F	T	T	F	F	T	T	T	F
F	T	F	T	F	T	T	F	T	T	F

Here is an example of a proof in propositional logic:

Prove: p->q, ~r->~q, ~r |- ~p

	Statements	Reasons
1.	p -> q	given
2.	~q -> ~p	contrapositive of 1
3.	~r -> ~q	given
4.	~r -> ~p	syllogism (2 & 3)
5.	~r	given
6.	~p	modus ponens (4 & 5)

How does one do a proof in propositional logic? Unfortunately, this is related to the satisfiability problem we saw earlier. Thus, we cannot expect to find proofs except by luck or by exhaustive search. Some skills in doing proofs can be acquired, however, through practice. Some helpful hints:

- Always keep in mind the conclusion, i.e., where you are going in the proof.
- Work backwards and forwards using logical equivalences and the rules of inference
- We assume the premises (the parts separated by commas in the statement to be proven) are true, so quite often you can gain insights right from the premises. For example, if "$p \wedge \sim q$" is a premise, what can we say about p and q?

The rules of valid inference are used constantly (both consciously and unconsciously) in all forms of problem-solving, even in everyday life:

```
Are the following arguments valid?  (i.e., Do they represent valid rules of
inference in drawing a conclusion?)

1) If I buy a new car, then I will not be able to go to Hawaii in December.
Since I am going to Hawaii in December, I will not buy a new car.

2) Either the butler or the maid murdered the Count.  If the butler did it, he
would not have been able to answer the phone at 11:00.  Since he did answer the
phone at 11:00, the maid must have done it.

3) If the football game runs late, then they will delay the start of 60
Minutes.  If 60 minutes runs late, the local news starts after 11:00.  The
local news started at 11:15 tonight, thus the football game ran late.

4) Either I will get an A in this course, or I will not graduate.  If I don't
graduate, I will go into the foreign legion.  I got an A in this course.
Therefore, I will not go into the foreign legion.
```

But remember in propositional logic, we really are not concerned with meaning as is implied by the exercises above. It is perfectly acceptable to prove the following in propositional logic:

```
If Fred does not live in Chicago, then he does not have a big nose.
Fred does not dance the hula.
If Fred lives in Chicago, he owns an elephant.
Either Fred has a big nose or he dances the hula.
Hence, Fred owns an elephant.
```

So who cares if we can do proofs and solve these little logic puzzles? Being able to do proofs is a key skill in computer science. One of our greatest heroes, Donald Knuth, once said:

> "Constructing a computer program from a set of specifications is equivalent in nature to writing a mathematical proof based on a set of axioms."

Another of our heroes, Jeffrey Ullman once said:

> "In addition to being the stuff of which all mathematics are ultimately made, formal proof has many applications in computer science. One application is automated theorem proving which is implemented in systems that find proofs of theorems by searching for sequences of steps that proceed from hypothesis to conclusion. Another application relates deduction (the process of finding a proof) to computation."

Some Common Fallacies

A **fallacy** is an error in reasoning that results in an invalid argument. Two of the most common forms of logical fallacies are called the **converse** error and the **inverse** error.

The converse error: If Fred is a cheater then Fred sits in the back row.
Fred sits in the back row.
Therefore, Fred is a cheater.

This comes down to the following argument that looks like modus ponens but it's not:

p -> q
q
Therefore, p

This is called the converse error because if we flipped the first premise to q -> p, the argument would be correct.

The inverse error: If Lassie is a human being, then Lassie is mortal.
Lassie is not a human being.
Therefore, Lassie is not mortal.

Here is the argument for this one:

p -> q
~p
Therefore, ~q

This is called an inverse error because if we replaced p -> q by its inverse, the argument would be correct.

Bibliography

G. Boole, *An Investigation of the Laws of Thought*, New York: Dover Press reprint, 1958 (originally, 1854)

L. Carroll, *Symbolic Logic*, New York: Crown, 1978.

H. B. Enderton, *A Mathematical Introduction to Logic*, New York: Academic Press, 1972.

E. Mendelson, *Introduction to Mathematical Logic*, New York: Van Nostrand Reinhold, 1964.

P. Suppes, *Introduction to Logic*, Princeton, NJ: D. Van Nostrand, 1987.

Historical Notes

Aristotle (384 BC-322 BC) wrote the first treatises on logic and deductive reasoning. His work is often called "traditional logic" because he described the properties of formal reasoning in terms of words and interpretation. Aristotle's example of modus ponens was expressed as

Socrates is a human being.
If Socrates is a human being, then Socrates is mortal.
Therefore, Socrates is mortal.

Several centuries later, logicians realized that language could get in the way of understanding the essence of formal reasoning. They espoused the use of symbols for the propositions, so language interpretation and ambiguity would not confuse matters. Gottfried Leibniz (1646-1716) came up with the idea of using symbols to mechanize the process of deductive logic. George Boole (1815-1864) and Augustus DeMorgan (1806-1871) are credited with the founding of the modern subject of symbolic logic.