Topic 1: Logic

02-680: Essentials of Mathematics and Statistics

August 27, 2024

Propositional logic usually contains unbound variables so the totality of the truth statement is unknown without a specific value.

Predicate logic qualifies those variables so the statement (sentence) can be fully evaluated; typically contains within it a propositional logic statement.

1 Propositional Logic

In logic a ***proposition*** is simply a statement that can be evaluated for truth. Something like "$2 + 2 = 4$" or "DJ is the CMU President". We usually use lower-case letters to represent ***atomic propositions***, sort of like variables. Something like

$$p \leftarrow \text{“}2 + 2 = 4\text{”}$$

$$q \leftarrow \text{“DJ is the CMU President”.}$$

We know that p is true, and q is false.

A ***compound proposition*** can take into account multiple atopic propositions to create a single statement:

$$p \wedge q$$

(which doesn't need to be true). We read the previous as "p *and* q", so for the whole statement to be true both atomic elements need to be true.

But we can also *negate* a proposition:

$$p \wedge \neg q$$

now the statement is true! We read this as "p and not q", or "$2 + 2 = 4$" and "DJ is ***not*** the CMU President".

All of the connectives and operations are listed below:

name	symbol/use	description
negation	$\neg p$	"not p"
conjunction	$p \wedge q$	"p and q"
disjunction	$p \vee q$	"p or q"
exclusive or	$p \oplus q$	"p or q but not both"
implication	$p \implies q$	"if p, then q"
mututal implication	$p \iff q$	"p if and only if q" (sometimes shortened to "iff")

While the order of operations on propositions is top to bottom in the table above, its best to use parenthesis to make sure your statements are clear. Technically the two statements below are the same, one is much more clear:

$$p \vee q \implies \neg r \wedge \ell \quad \text{and} \quad (p \vee q) \implies (\neg r \wedge \ell)$$

Negating Statements. Sometimes called ***De Morgan's Laws***, negating a compound statement requires a little thought, and we have some examples later in the topic that may help reinforce the ideas. The basic rules are as follows:

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

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

(here $\equiv$ is logical equivalence). As text were saying "if not both p and q" is equivalent to saying "either not p or not q".

Associativity and Commutativity. Just like in addition, most logical operators (specifically conjunction[$\wedge$], disjunction[$\vee$], exclusive or [$\oplus$], and mutual implication [$\iff$]) are associative and commutative. As a reminder some examples:

$$(p \vee q) \vee r \equiv v \vee (q \vee r) \qquad \text{(associativity)}$$

$$p \wedge q \equiv q \wedge p \qquad \text{(commutativity)}$$

Its important to note that negation and implication are not associative or commutative (I'll leave that as a thought experiment).

We may discuss other logical properties as we move through the semester.

Bound vs Free Variables. While all the stuff above we know if p and q are *always* true or false (we call these tautologies), most of the time a proposition looks more like this:

$$r \leftarrow \text{"}2 + x = 4\text{"}$$

where x is not know (or in other words it's ***unbound*** or free, thus without a definition of x we can't say if the statement is true or not.

2 Predicate Logic

We often need to talk about groups of items.

For example, if we want to say something like: "for any integer x, the value of $x * 0$ is 0." We would write that using the ***universal qualifier***:

$$\forall x \in \mathbb{Z} : x * 0 = 0$$

(we will talk about it in a coming discussion, but $\mathbb{Z}$ is the symbol for the set of all integers and $\in$ is read as "element of" or simply "in"). We say $\forall$ as "for all", in fact the latex command for the symbol is `\forall`. Noice that "$x * 0 = 0$" is a proposition!

But sometimes we want to not talk about all items, but one (or more) in particular, in which case we can use the ***existential qualifier***:

$$\exists y \in \mathbb{Z} : y^2 = |y + y|.$$

We say $\exists$ as "there exists", and the latex is `\exists`. In this case there exists *more than one* y that satisfies the proposition, so the statement is true.

It may help to think of qualifiers as loops: if you looped through all of the items (in this case integers) you need to find one/or determine that all are true.

Negation (De Morgan again). We may also need to negate expressions with qualifiers. Just like with propositional logic its not trivial. Once again De Morgan comes to the rescue though:

$$\neg\left[\forall x : P(x)\right] \iff \left[\exists x : \neg P(x)\right]$$
$$\neg\left[\exists x : P(x)\right] \iff \left[\forall x : \neg P(x)\right].$$

Notice that whats left to do ($\neg P(x)$) is just the negation of a proposition, which we learned how to do earlier.

2.1 Nested Qualifiers

It is possible to combine multiple qualifiers together, but we need to be careful about the ordering. For instance we can say something like

$$\forall x \in \mathbb{Z} : \left[\exists y \in \mathbb{Z} : x = -1y\right]$$

but if we write it incorrectly we find a statement that is ***unsatisfiable***

$$\exists x \in \mathbb{Z} : \left[\forall y \in \mathbb{Z} : x = -1y\right].$$

When negating nested qualifiers, its basically a recursive procedure:

$$\neg\left[\exists x : \left[\forall x : P(x)\right]\right] \iff \left[\forall x : \neg\left[\forall x : P(x)\right]\right] \iff \left[\forall x : \left[\exists x : \neg P(x)\right]\right].$$

Useful References

Liben-Nowell, "Connecting Discrete Mathematics and Computer Science, 2e". Ch. 3