Discrete Maths Notes

August 25, 2023

Contents

1 Logic

1.1 Propositional Logic

1.1.1 Basics

A **proposition** is a statement that is either true or false.
Prepositions will be represented mathematically with capital letters A, B, C...
These prepositions are then are connected into more complex compound prepositions using *connectives*. Connectives are statements like "and, implies, if-then" and are represented mathematically with the symbols below.

ℹ It's not always easy to determine if they're true/false.

Connectives			
Symbol	Name	English Term(s)	Reading
$\wedge$	AND	And, But, Also	A and B
$\vee$	OR	-	A or B
$\implies$	IMPLICATION	If A, Then B If A, then B A implies B A, therefore B A only if B B follows from A A is a sufficient condition for B B is a necessary condition for A	A implies B
$\iff$	BICONDITIONAL	If & only if A is necessary and sufficient for B	A if and only if B
$\neg$	NEGATION	Not...	Not A

ℹ *A Bicondtional can also be thought of* $(A \implies B) \wedge (B \implies A)$
Negation may sometimes be represented as $A\prime$ *or* $\overline{A}$

1.1.2 Terminology

$A \wedge B$ - conjuction of conjuncts A and B
$A \vee B$ - disjunction of disjuncts A and B
$A \implies B$ - A is the hypothesis$_{\text{/antecedant}}$ and B is the conclusion$_{\text{/consequence}}$

1.1.3 Examples

1 Compound Proposition

If all humans are mortal$_{\text{prp } A}$ and all Greeks are human$_{\text{prp } B}$
then all Greeks are moral$_{\text{prp } C}$ can be represented as $A \land B \implies C$

2 Negation

Chocolate is sweet $\rightarrow$ Chocolate is not sweet

Peter is tall and thin $\rightarrow$ Peter is short or fat

ℹ Short and fat would be incorrect!

The river is shallow or polluted $\rightarrow$ The river is deep and polluted.

ℹ Not shallow or not polluted would be incorrect!

3 Implication: hypothesis and conclusion

If the rain continues then the river will flood

A sufficient condition for a network failure is that the central switch goes down

The avocados are ripe only if they are dark and soft

A good diet is a necessary condition for a healthy cat

1.1.4 Satifiability, Tautology, Contradiction

A proposition is satisfiable if it is true for *at least one* combination of boolean values.

ℹ You don't need a whole truth table for this, just look for one!

A Boolean Satisfiablity Problem (SAT) is checking for satifiability in a propositional logic formula.

A Tautology is a proposition that is always true
$_{\text{ex}}$ $A \lor \neg A$

A Contradiction is a proposition that is always false.
$_{\text{ex}}$ $A \land \neg A$

1.2 Truth Tables

1.2.1 Basics

Truth Tables are used for determining all the possible outputs of a complex compound propostion.

The Columns Are for the prepositions, intermediate compound prepositions and the whole compound preposition.

The intrmt' prepositions are optional steps to make solving easier, use as needed.

The Rows Are to contian the different sets of possible truth values for each proposition. You will have 2^p rows where p is the number of propositions (then +1 for the header).

The connectives in a compound propositional logic problem follow an order of precedence (the PEMDAS of logic) in the following order;
$\neg$, $\wedge$, $\vee$, $\implies$, $\iff$

1.2.2 Connective Outputs

Negation

A	$\neg A$
T	F
F	T

And

A	B	$A \wedge B$
T	T	T
T	F	F
F	T	F
F	F	F

Or

A	B	$A \vee B$
T	T	T
T	F	T
F	T	T
F	F	F

Implication

A	B	$A \implies B$
T	T	T
T	F	F
F	T	T
F	F	T

An implication is true when the hypothesis is false or the conclusion is true.

Bicondtional

A	B	$A \iff B$
T	T	T
T	F	F
F	T	F
F	F	T

A Bicondtional is true when the two prepositions have the same value.

Out of all these outputs, the most unintuitive is the 3rd implication output ($F, T \implies T$). The easiest way to understand this output is with the proposition “If it is raining, then the ground is wet”; now say you step outside and it is not raining, but the ground is wet. The entire statement isn't false or incorrect, but the first part of it still has a false value.
The only way to make an implication false is when the hypothesis is true but the conclusion is false.

1.2.3 Examples

$A \implies B \iff B \implies A$

A	B	$A \implies B$	$B \implies A$	——
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

$A \wedge \neg B \implies \neg C$

A	B	C	$A \wedge \neg B$	——
T	T	T	F	T
T	T	F	F	T
T	F	T	T	F
T	F	F	T	T
F	T	T	F	T
F	T	F	F	T
F	F	T	F	T
F	F	F	F	T

ℹ *Remember, columns like $A \implies B$ are optional in-between steps to help solve each problem.*

1.2.4 Excercise: Finding Tautologies, Satisfiable & Contradicting Props'

Indicate whether each of the following is a tautology, satisfiable but not a tautology or a contradiction;

$A \implies B$

$A \implies A$

$A \implies \neg B \vee \neg C$

$A \vee B \implies B$

$(A \wedge B) \implies (A \vee B)$

$A \vee \neg A \implies B \wedge \neg B$

(Answers and explainations on the next page...)

ℹ *Notice how none of these rely on drawing out a whole truth table! Focus on trying to find a way to get each proposition to output true and a way to get it to output false!*

$A \implies B$
Satisfiable but not a tautology
Just knowing the properties of a implication you should know there's way to get true outputs and a false output.

$A \implies A$
Tautology
Only would be $T \implies T$ or $F \implies F$, both of which result in true.

$A \implies \neg B \vee \neg C$
Satisfiable but not a tautology
Instead of making a long unpleasant truth table, it's easiest to start by simply looking for one true and one false possible output.
We can make the left side true simply by making A false, since all that remains is an or statment we now have a true output.
We can just as easily find a false output for this proposition with $A = T$, $B = T(\neg B = F)$ to make the implication false, then we can just make $\neg C$ false to make the or output false.

$A \vee B \implies B$
Satisfiable but not a tautology
If we make B true then the biconditional will always be true regardless of A.
There is only one way to make an implication false, so if we can set up A and B to result in that false output, it won't be a tautology. If we make A true and B false it will make the implication false!

$(A \wedge B) \implies (A \vee B)$
Tautology
Remember the only way to make an implication false is if the hypothesis is true and the conclusion is false. There is absoluely no way to do this because of the and/or setup!

$A \vee \neg A \implies B \wedge \neg B$
Contradiction
The left side is always true and the right side is always false. So the result of the implication is always false!

1.3 Equivalence

1.3.1 Introduction to Equivalence

Two (compound) propositions P and Q are **logically equivalent** when their truth values always match (Meaning they'll have the same truth table!).
Equivalence is denoted by $P \equiv Q$.

Equivalence relates heavily to the concept of Tautologies;

P and Q are equivalent when $P \iff Q$ is a tautology.

A proposition P is a tautology iff (if and only if) it is equivalent to T (true), i.e $P \equiv T$

1.3.2 Examples

1

Given the implication $A \implies B$, are the following equivalent?

The contrapositive: $\neg B \implies \neg A$

The converse: $B \implies A$

A	B	$A \implies B$	$\neg B \implies \neg A$	$B \implies A$
T	T	T	T	T
T	F	F	F	T
F	T	T	T	F
F	F	T	T	T

Looking at the table we can see that $A \implies B$ and $\neg B \implies \neg A$ are equivalent.

Now, what about $\neg A \vee B$?

A	B	$A \implies B$	$\neg A \vee B$
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

Yep! $\neg A \vee B \equiv A \implies B$.

This is actually one of the equivalence laws you'll see in the next section!

Understanding equivalent boolean expressions is very important in computer science (for code) and chip design (for logic gates). Consider the code below;

```
if(x > 0 || (x <= 0 && y > 100))
```

Lets see if we can change this expression to something equivalent but simplified.
Let A be `x > 0` and let B be `y > 100`
Now we can compare the truth values of $A \vee (\neg A \wedge B)$ and $A \vee B$.

A	B	$A \vee (\neg A \wedge B)$	$A \vee B$
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

They're equivalent! We can reduce the if statement's expression to simply;

```
if(x > 0 || y > 100)
```