Notes for ECE 369 - Discrete Mathematics for Computer Engineering

Ezekiel Ulrich

August 29, 2023

These are lecture notes for fall 2023 ECE 36900 at Purdue. Modify, use, and distribute as you please.

Contents

Course Introduction

This course introduces discrete mathematical structures and finite-state machines. Students will learn how to use logical and mathematical formalisms to formulate and solve problems in computer engineering. Topics include formal logic, proof techniques, recurrence relations, sets, combinatorics, relations, functions, algebraic structures, and finite-state machines. For more information, see the syllabus.

Equations

1. De Morgan's Theorem:
   $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
   $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
2. Modus ponens (mp)
   p
   $p \to q$
   $\therefore q$
3. Modus tonens (mt)
   $p \to q$
   $\neg q$
   $\therefore \neg p$

Propositional Logic

We often wish that others would be more logical, tell the truth, or shower. While studying formal logic cannot help with the latter (in fact, studies have shown a negative correlation between hygiene and studying formal logic) it is a useful way to define what the first mean two. In a formal logic model, we have two constructs:

- **Statements/propostion**: A statement or a proposition is a sentence that is either true or false. Propositions are often represented with letters of the alphabet. For example: "q: the more time you spend coding, the less time you have to buy deodorant."
- **Logical connectives**: Used to connect statements. For example, "and" is a logical connective in English. It can be used to connect two statements, e.g. "the person next to me smells like dog *and* looks like a dog" to obtain a new statement with its own truth value.

  Here are common logical connectives in Boolean logic:

Logical Connective	**Symbol**
Negation (NOT)	$\neg$ or $'$
Conjunction (AND)	$\wedge$
Disjunction (OR)	$\vee$
Exclusive OR (XOR)	$\oplus$
Implication	$\rightarrow$
Biconditional	$\leftrightarrow$

Table 1: Logical Connectives in Boolean Logic

Truth table: Defines how each of the connectives operate on truth values. Every connective has one. For example, consider $\wedge$ AND:

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

Table 2: Truth table for $\wedge$

We see that p AND q is only true when p is true and q is true. Similarly, p or q is only true when p is true or q is true (or both). An important connective for discovering new truths is the implication $\rightarrow$, which basically says "if the first letter is true, then so is the second". Let p: "I live in Wiley" and q: "I have no AC". In English, the statement $p \rightarrow q$ would be stated as "If I live in Wiley, then I have no AC".

Table 3 shows the truth table for $\rightarrow$. It may not seem immediately clear why, for instance, if p and q are false, then $p \rightarrow q$ is true. If we

p	q	$p \to q$
T	T	T
T	F	F
F	T	T
F	F	T

Table 3: Truth table for $\to$

consider what this means in English, then all we know is that I don't live in Wiley. Perhaps I live in Tarkington and still don't have AC, or perhaps I live in Honors and I do. In any case the first letter isn't true, so "if the first letter is true then so is the second" stands as true. If we have the statement $p \to q$, then we call q a *necessary condition* for p. Conversely, p is a *sufficient condition* for q.

Say we have a statement such as $\neg A \vee B \to C$. This is ambiguous, since we can interpret it as either $(A \vee B) \to C$ or $A \vee (B \to C)$. The truth tables will differ in each case, so it becomes necessary to specify in what order we should apply logical connectives.

1. Parentheses "()"
2. Negation "$\neg$"
3. AND and OR "$\wedge$, $\vee$"
4. Implication "$\to$"
5. Biconditional "$\leftrightarrow$"

Rules and proofs

With each additional variable in your truth table, the number of choices grows exponentially. Specifically, if you have n statement letters, you would have 2^n choices for your truth table.

Tautology: A formula that is true in every model. Example: the Bible is infallible because the Bible itself says it is infallible.

Contradiction: A formula that is false in every model Examples: "it is raining and it is not raining", "I am sleeping and I am awake", "IU is a good school".

Confusion often arises when negating a sentence such as "the book is thick and boring". An natural inclination is to negate it thus: "the book is not thick and not boring". However, consider the truth table for this: p: "the book is thick", q: "the book is boring". We can see the last two rows are not identical, therefore the negation of "the book is not thick and not boring" is not "the book is not thick and not boring". For p to be false, either the book must not be thick *or* the book must not be boring. This is summarized by **De Morgan's Theorem**:

$$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$$

Interestingly, it is possible to prove any statement in a system where a contradiction exists. This is known as the *principle of explosion*. To see how it works, consider the following example:

1. p: Donuts are good for you.
2. q: Unicorns exist.

I'll now assume the contradictory statement "donuts are good for you and donuts are not good for you".

$$\begin{array}{ll} \neg p \wedge p & \text{(Contradiction)} \\ p \vee q & \text{(Addition)} \\ \neg p & \\ \therefore q & \end{array}$$

Ergo, unicorns exist :)

p	q	$p \wedge q$	$\neg(p \wedge q)$	$\neg p \wedge \neg q$
T	T	T	F	F
T	F	F	T	F
F	T	F	T	F
F	F	F	T	T

$$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$$

We now have a sufficient understanding of truth tables and logical connectives to come up with some useful rules. First of these are
Modus ponens (mp):

$$\begin{aligned} &p \\ &p \rightarrow q \\ &\therefore q \end{aligned}$$

and
Modus tollens (mt):

$$\begin{aligned} &p \rightarrow q \\ &\neg q \\ &\therefore \neg p \end{aligned}$$

Below are two tables for commonly used rules.

Expression	Equivalent to	Name - abbreviation
$p \vee q$ $p \wedge q$	$q \vee p$ $q \wedge p$	Commutative - comm
$(p \vee q) \vee r$ $(p \wedge q) \wedge r$	$p \vee (q \vee r)$ $p \wedge (q \wedge r)$	Associative - ass
$\neg(p \wedge q)$ $\neg(p \vee q)$	$\neg p \vee \neg q$ $\neg p \wedge \neg q$	De Morgan's Laws - De Morgan
$p \rightarrow q$	$\neg p \vee q$	Implication - imp
p	$\neg(\neg p)$	Double negation - dn
$p \leftrightarrow q$	$(p \rightarrow q) \wedge (q \rightarrow) p$	Def'n of equivalence - equ

Table 4: Equivalence rules

At this point, let us formally define an
Argument: An argument can be symbolized as

$$P_1 \vee P_2 \vee P_3 \vee ... \vee P_n \rightarrow Q$$

where P_i is called a hypothesis and Q is the conclusion. If this statement is a tautology, then the argument is *valid*. There are multiple ways we could prove a given argument is a tautology. For instance, we

From	Can derive	Name - abbreviation
$p, p \to q$	q	Modus ponens - mp
$p \to q, \neg q$	$\neg p$	Modus tollens - mt
p, q	$p \vee q$	Conjunction - con
$p \vee q, \neg p$	q	Disjunction - dis
$p \wedge q$	p, q	Simplification - sim
p	$p \vee q$	Addition - add

Table 5: Inference rules

could create a truth table and brute force an answer. However, with even four hypotheses this process is tedious, and with each additional hypothesis it becomes exponentially harder. Therefore we instead often turn to the

Proof sequence: a sequence of well-formed formulas in which each formula is either a hypothesis or the result of applying a derivation rule to earlier well-formed formulas. In practice this looks like

$$\begin{aligned}
P_1 &\quad \text{(hypothesis)} \\
P_2 &\quad \text{(hypothesis)} \\
P_3 &\quad \text{(hypothesis)} \\
&\quad \dots \\
P_n &\quad \text{(hypothesis)} \\
\text{(formula) } 1 &\quad \text{(obtained from derivation rule)} \\
\text{(formula) } 2 &\quad \text{(obtained from derivation rule)} \\
&\quad \dots \\
\text{(formula) } n &\quad \text{(obtained from derivation rule)} \\
\therefore Q &
\end{aligned}$$

Let's use all this new information in a simple proof.

$$\begin{aligned}
A &\quad \text{(hypothesis)} \\
A \to B &\quad \text{(hypothesis)} \\
B \to C &\quad \text{(hypothesis)} \\
B &\quad \text{(1, 2, mp)} \\
C &\quad \text{(4, 3, mp)} \\
\hline
\therefore C &
\end{aligned}$$

If we wish to apply our knowledge of logic to the real world, some practice in translating natural language to formal logic is necessary. Let's test it with this statement: "If chicken is on the menu, then don't order fish, but you should have either fish or salad. So if chicken is on the menu, have salad." Let C: "Chicken is on the menu", F: "You order fish", and S: "You have salad". We know that if chicken is on the

menu you don't order fish, that you should have either fish or salad, and we'd like to show that if chicken is on the menu you should have salad.

$$\begin{array}{rl} C & \text{(hypothesis)} \\ C \to \neg F & \text{(hypothesis)} \\ F \vee S & \text{(hypothesis)} \\ \neg F & \text{(1, 2, mp)} \\ S & \text{(3, 4, dis)} \\ \hline \therefore S & \end{array}$$

Predicate logic

Predicate logic: Capable of making statements about entire groups instead of individual letters. In predicate logic, propositions are expressed in terms of predicates, variables and quantifiers, the latter of which propositional logic lacks.
Quantifer: How many objects have a certain property: "for every" or "for some".
Predicate: Property that a variable may have.
Domain of interpretation: Collection of objects from which the variable is taken.
Universal quantifer: "For all": $\forall$
Existential quantifer: "There exists": $\exists$

Predicate formulas are built by combining predicates with quantifers, grouping symbols, and logical connectives seen before.

For example: $(\forall x)(\exists y)x > y$. We would read this statment as "for all x there exists a y such that $x > y$." At first glance it may seem obvious that this statement is true, but consider the domain. What if the domain is all natural numbers? Then we could let $x = 1$ (or zero depending or your definition of natural numbers) and there would be no corresponding lesser y.