EECS/MATH 1019
First Class
Sections 1.1–1.3: Some Basic Logic

January 10, 2023

Instructor: Prof. Neal Madras, Dept. of Mathematics & Statistics
S616 Ross, `madras@yorku.ca`

See eClass for my Student Office Hours, course outline, and everything else you need to know about this course.

In fact, almost everything relating to the course will be in eClass, including class slides, recordings, and readings for upcoming classes. It also has a list of additional problems for practice, which you should work on but not submit.

Course: LE/EECS 1019 NW 3.00 = SC/MATH 1019 NW 3.00
Discrete Mathematics for Computer Science

N = section (there are 3 sections this winter)
W = Winter term
3.00 = number of credits
This course is cross-listed between two Faculties:
LE = Lassonde Engineering; SC = Science
This course is required for students majoring in Computer Science and related programs. Engineering students normally take EECS 1028 3.00 instead.
Course credit exclusions: LE/EECS 1028 3.00, SC/MATH 1028 3.00, SC/MATH 2320 3.00

Prerequisites: SC/MATH 1190 3.00 or two 4U math courses including MHF4U (Advanced Functions)
You are responsible for making sure that you have the right prerequisites for each course you take.

We will use some **online polling** during class. This polling is anonymous, and will not be graded in any way. Any device that connects to the internet will let you participate.

Let's try a poll.

Which best describes you?
(A) Majoring in a Computer Science program
(B) Majoring in Engineering or another Lassonde program
(C) Majoring in a program in Mathematics and Statistics
(D) Majoring in another Science program
(E) Undecided or other

Google form at https://forms.gle/fBH48UAsLQZRudxY9
(sent in email yesterday; also on eClass page)

Textbook: Kenneth H. Rosen, *Discrete Mathematics and Its Applications* (8th edition), 2019, McGraw Hill.
Online homework: Connect
The Connect package is required for doing the homework. Connect also comes with the electronic version of the text. At the York Bookstore, you can buy bundle of the paperback text together with Connect, or you can buy Connect only with only the e-text (purchase online).

We will have approximately weekly homework assignments. The first one will be posted very soon, to be due Sunday January 22. In Connect, you will be able to submit answers (early) and then have a second chance to correct errors and submit again (same deadline).
In future weeks, some homework will be written up and submitted for marking by teaching assistants.

Breakdown of the course grade:

Assignments	20 %	approximately weekly
Two midterm tests	40 %	dates to be confirmed
Final exam	40 %	April; to be scheduled by Registrar

Where to go for help in this course:

- Instructor: by email or Student Office Hours in S616 Ross (times to be posted in eClass)
- Teaching Assistants (graduate students) in Math Stats Lab, S525 Ross. Hours to be announced. Starts Jan. 16.
- PASS (Peer Assisted Study Sessions) for EECS/MATH 1019 organized through Bethune College. Details will be announced.
- Peer tutoring at Bethune College (if available).

Although this is a Math course, it lays the foundations for future courses in Computer Science. Each part has applications in computer science as well as in other fields:

(a) Logic and proofs (verifying program correctness; needed for EECS 1090)
(b) Sets, functions, sums (basic terminology, used in most courses with mathematical components)
(c) Growth of functions (describes efficiency of algorithms)
(d) Induction and recursion (used to specify syntax and implement recursive algorithms)
(e) Recurrence relations (calculations arising from (c) and (d))
(f) Relations (relational databases and SQL)
(g) Graphs and trees (e.g. data structures, the Internet, social networks)

(Remark: Some parts may be more familiar to you than others.)

Some Basic Logic

Reference: Sections 1.1–1.3

A proposition is a sentence that is either true or false.

Examples (We often provide a label to identify a proposition)

U: Toronto is in Canada.

V: Ontario is in France.

W: Green laughs are tall. (Neither true nor false)

Y: She is a doctor. (The truth value depends on who *she* is.)

A: $7 > 3$.

B: Is π greater than 4?

C: $10^9 > 9^{10}$.

D: $x > 7$. (The truth value depends on the value of x.)

The sentences U, V, A, and C are propositions. The others are not.

If x is a real number, then we can write $D(x)$ to denote the statement "$x > 7$."

Then we can talk about the truth values of $D(6)$, $D(7)$, and $D(8)$, for example.

($D(8)$ is true; $D(6)$ and $D(7)$ are false.)

We shall discuss sentences with variables, like $D(x)$, in Section 1.4.

E: $3 < \pi < 4$. This means "$3 < \pi$ and $\pi < 4$".

Truth table for *AND* and *OR*:

Suppose P and Q are statements.

P	Q	P and Q P∧Q	P or Q P∨Q
T	T		
T	F		
F	T		
F	F		

Negation of a statement *P*: "not P" or "¬ P"

The statement ¬P is true precisely when P is false. For example:

B: $\pi > 4$; then $\neg B$: $\pi \leq 4$.

E: $3 < \pi < 4$; then ¬E: $\pi \leq 3$ *or* $\pi \geq 4$.

(See p. 27: $\neg(P \wedge Q)$ has same truth value as $(\neg P) \vee (\neg Q)$;
$\neg(P \vee Q)$ has same truth value as $(\neg P) \wedge (\neg Q)$; and
$\neg(\neg P)$ has same truth value as P.)

When we use the word “or” in speaking, sometimes we allow both possibilities, but sometimes not.

“I will work on the assignment on Monday or Tuesday.”
“With the $7 lunch, you can have a chicken sandwich or a felafel wrap.”
XOR: Exclusive “or”. Sometimes we say “either A or B” to exclude the possibility of both.

P	Q	P OR Q $P \vee Q$	P XOR Q $P \oplus Q$
T	T	T	F
T	F	T	T
F	T	T	T
F	F	F	F

Implication

Medical knowledge tells us that any one of the following can occur:

Joan has pneumonia and has a fever,

Joan does not have pneumonia and does not have a fever, or

Joan does not have pneumonia but does have a fever,

and that the following **cannot** occur:

Joan has pneumonia but does not have a fever.

We can summarize the above knowledge by the sentence

If Joan has pneumonia, then Joan has a fever.

The corresponding truth table is

P: Joan has pneumonia	Q: Joan has a fever	If P then Q
T	T	T
F	F	T
F	T	T
T	F	F

We use the notation $P \to Q$ to mean "P implies Q", which can also be expressed as "If P, then Q".
(E.g., if Joan has pneumonia, then Joan has a fever)

Other ways to express $P \to Q$:
Q if P
Q whenever P
P is a sufficient condition for Q (i.e., *P is enough to ensure Q*)
Q is a necessary condition for P (i.e., *we can't have P without Q*)
P only if Q (*the only way that P can be true is if Q is also true*)

Truth table:

P	Q	P→Q	$(\neg P)$ or Q
T	T	T	T
F	F	T	T
F	T	T	T
T	F	F	F

$P \to Q$ has the same truth value as $(\neg P) \vee Q$. In the terminology of Section 1.3, $P \to Q$ and $(\neg P) \vee Q$ are *logically equivalent*.
That is, $P \to Q \equiv (\neg P) \vee Q$.

Poll: Which of the following is logically equivalent to $P \to Q$?

(A) $Q \to P$

(B) $(\neg P) \to (\neg Q)$

(C) $(\neg Q) \to (\neg P)$

(D) all of the above

(E) none of the above

Answer: (C). (the "contrapositive" of $P \to Q$).
E.g. "If it is chocolate, then I will eat it" is logically equivalent to "If I won't eat it, then it's not chocolate!"
Notice that both assertions allow for the possibility of things that I will eat that are not chocolate (which is one reason that I am still healthy).
Both are equivalent to "It is not chocolate, or I will eat it."

How can we prove that $(\neg Q) \to (\neg P)$ is logically equivalent to $P \to Q$?

One way is to complete the truth table for each statement.

It can also be proved by logical arguments, as follows.

$$\begin{aligned}
(\neg Q) \to (\neg P) &\equiv (\neg(\neg Q)) \vee (\neg P) && \text{(recall earlier slide)} \\
&\equiv Q \vee (\neg P) && (\text{since } \neg(\neg Q) \equiv Q) \\
&\equiv (\neg P) \vee Q && (A \vee B \equiv B \vee A) \\
&\equiv P \to Q && \text{(again, by earlier slide).}
\end{aligned}$$

Here is one more kind of "compound" proposition:

$P \leftrightarrow Q$ means $((Q \to P)$ and $(P \to Q))$

That is, ([P if Q] and [P only if Q])

Hence $P \leftrightarrow Q$ is often read as "P if and only if Q"

P	Q	$P \to Q$	$Q \to P$	$P \leftrightarrow Q$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

Precedence of logical operators: See sections 1.1.5 (and 1.4.6).
In particular, negation has highest priority.
E.g. $\neg P \to Q$ means $(\neg P) \to Q$, and does NOT mean $\neg(P \to Q)$.
For clarity, it is often safer to use more parentheses than fewer.

Satisfiability (subsections 1.3.5–1.3.7)

Suppose p, q, r, and s are statements. We can write a compound statement like $C : (p \to (r \vee \neg s)) \wedge ((\neg q \vee r) \leftrightarrow s)$ and think of p, q, r, and s as *logical variables* that can each be either True or False. We say that the compound statement C is

- a **tautology** if C is always True (for all possible truth values of p, q, r, and s)
- a **contradiction** if C is always False (for all possible truth values of p, q, r, and s)
- **satisfiable** if C is not a contradiction (that is, if it is True for some choice of truth values of p, q, r, and s)

E.g. The given statement $C : (p \to (r \vee \neg s)) \wedge ((\neg q \vee r) \leftrightarrow s)$ is satisfiable because, it is True when all of p, q, r and s are True, or when p is False and r and s are both True, ...
C is not a tautology because C is False when all of p, q, r, and s are False. (Check this!)
(So C is a **contingency**, i.e. neither tautology nor contradiction.)

We defined a compound statement to be **satisfiable** if it is True for some choice of the truth values of its logical variables.

Consider the following two compound statements:

$$\begin{array}{ll} (A) & (p \wedge \neg q \wedge r) \vee (\neg p \wedge q \wedge r) \vee (p \wedge q \wedge \neg r) \\ (B) & (p \vee \neg q \vee r) \wedge (\neg p \vee q \vee r) \wedge (p \vee q \vee \neg r) \end{array}$$

Poll: Is it easier to see whether (A) or (B) is satisfiable?

For (A) to be True, it suffices to choose truth values so that any one expression in parentheses is True. E.g., the first expression $(p \wedge \neg q \wedge r)$ is True when p is True, q is False, and r is True. With these values, (A) must be true.

(B) is less straightforward. It is not hard with only 3 variables, but with 100 variables and 1000 groupings (with $\vee$ inside the parentheses, and $\wedge$ between the parentheses), a satisfiability problem of type (B) becomes much harder. It is not known whether there is an "efficient" algorithm for deciding satisfiability (i.e., much faster then checking all 2^{100} possible truth assignments). Satisfiability is "one of the hardest problems in NP" (see Algorithms course).

Next class: Read Sections 1.4 and 1.5.

Homework 1 will be posted in Connect, due Sunday January 22 at 11:59 pm.