CS 1010 Discrete Structures
Lecture 1:
Logic

Maria Francis

November 19, 2020

Mathematical Reasoning

- How to express ideas mathematically?
- A standard style that is unambiguous to interpret.
- English language can be very ambiguous.
  - For eg: *If you can solve any problem we come up with, then you get an A for the course.*
  - What happens if you can solve some problems, can you get an A?
  - What if you can not solve even a single one of the problems, can you get an A?

Introduction to Logic

- Uncertain meanings is okay in a conversation but in mathematics and programming, ambiguities can be a problem.
- Mathematicians introduced the language of logic to get around this ambiguity.
- We will also see an important open problem in computer science in this study of language of logic.
- Logic rules have applications in the design of computer circuits, in the design of computer programs, etc.

Propositions

- Basic building block of logic.
- Proposition is a declarative sentence that is either true or false but not both.
- Examples: $1 + 1 = 2$, Toronto is the capital city of Canada.
- Not examples: Read this carefully, $x + 1 = 2$
- Propositions are represented by propositional variables such as $p, q, r, s, \ldots$ or $P, Q, R, S, \ldots$.
- Truth value of a proposition is true (T) if it is a true proposition and false (F) if it is a false proposition.
- Propositional Calculus or Propositional Logic : Area of logic that deals with propositions.
- Developed first by Aristotle 2300 years ago.

Compound Propositions

- Producing new propositions from logical operators.
- Discussed first by George Boole (of Boolean algebra fame) in 1854, a British mathematician who did a lot of work in logic.
- Propositional variables are also called Boolean variables.
- NOT ($\neg$), AND ($\wedge$), OR ($\vee$), IFF ($\leftrightarrow$), IMPLIES ($\Rightarrow$/$\rightarrow$): operations that change or combine propositions.
- Precise meaning is expressed by truth tables.
- If P is a proposition then so is NOT(P).

P	NOT(P)
T	**F**
F	**T**

Compound Propositions

P	Q	P AND Q
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**F**

P	Q	P OR Q
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

P OR Q is true even if both P and Q are true.

"You may have cake, or you may have ice cream" in mathematics means you can have both!

XOR

If you want to exclude the case where both P and Q are true: Exclusive-or/ $\oplus$

P	Q	P XOR Q
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

Students who have taken calculus or computer science, *but not both*, can take this class.

IMPLIES

P IMPLIES $(\Rightarrow, \rightarrow)$ Q

P	Q	P IMPLIES Q	
T	**T**	**T**	(tt)
T	**F**	**F**	(tf)
F	**T**	**T**	(ft)
F	**F**	**T**	(ff)

An implication is true exactly when the if-part is false or the then-part is true.

A large fraction of all mathematical statements are of the if-then form.

How many rows in a truth table if there are n variables? Each variable can take 2 values (T or F) so there are 2^n different assignments/rows in the truth table.

Implicit use of IFFs

- IFF or biconditionals are not always explicitly stated, especially in casual English language.
- For eg: "If you finish your lunch, then you can have ice-cream" typically means "If you have lunch you can have ice-cream" and "You can have ice-cream *only if* if you have your lunch".
- In mathematics we have to explicitly specify if we mean the conditional statement $p \rightarrow q$ or biconditional $p \leftrightarrow q$.

False Hypothesis

- In mathematics, an implication as a whole is considered true when its hypothesis is false.
- This may seem strange since we look at a causal connection between the hypotheses and conclusions.
- Consider this statement:
  - If you followed the security protocol, then your account won't get hacked.
  - Mathematically and casually the proposition is true. There is a clear relation between protocols and account hackability.
- Consider the statement:
  - If pigs could fly, then your account won't get hacked.
  - Causally, this is false since there is no relation between pigs flying and account hackability. Mathematically this implication counts as true.
- Easier to analyze when we look at it abstractly as P,p, etc.

Equivalence of Propositions

- Two compound propositions p, q are logically equivalent $(p \equiv q)$ if for all assignments they evaluate to the same truth values, i.e. same truth table.
- p XOR $q \equiv (p \vee q) \wedge \neg(p \wedge q)$. Or equivalently,
- p XOR $q \equiv (p \wedge \neg q) \vee (\neg p \wedge q)$.
- $p \rightarrow q \equiv (\neg p \vee q)$.
    - This is a very useful way to think about the implication.
- Verify all the above equivalences using truth tables.
- Order of precedence: NOT, AND, OR. Preferred way is to use parenthesis.
- And IFF and IMPLIES take lower precedence in which IMPLIES takes higher precedence.

Contrapositive

$p \to q \equiv (\neg q \to \neg p)$ This is called the contrapositive.

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

Same truth table as $p \to q$.

Equivalence of Propositions

- Logical equivalences are an extremely useful tool in reasoning.
- But having to build truth tables do not seem like the most efficient/intuitive method.
- So let us look at how to build new equivalences from basic logical identities.
- A compound proposition that is always true is called a tautology.
  - $p \vee \neg p$
- One that is always false is called a contradiction.
  - $p \wedge \neg p$

Equivalence of Propositions

- Another way of defining equivalence of p and q:
  - $p \equiv q$ if $p \leftrightarrow q$ is a tautology.
- Note: $\equiv$ is not a logical connective and $p \equiv q$ is not a compound proposition, it is just a statement.
- $\Leftrightarrow$ is sometimes used instead of $\equiv$ to denote logical equivalence.
- So how do we infer long, complicated equivalences from logical equivalences that we have?

De Morgan's Laws

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

How to show these? Use truth tables.

Augustus De Morgan

- Augustus De Morgan in 1840s made contributions to symbolic logic.
- Extremely prolific writer, author of > 1000 articles that include biographies of Newton and Halley.
- Gave the first clear explanation of mathematical induction.
- Born in India.

Basic Logical Identities/Equivalences

Identity Laws

$$p \wedge T \equiv p$$
$$p \vee F \equiv p$$

Domination Laws

$$p \vee T \equiv T$$
$$p \wedge F \equiv F$$

Idempotent Laws

$$p \vee p \equiv p$$
$$p \wedge p \equiv p$$

Basic Logical Identities/Equivalences

Double Negation

$$\neg(\neg p) \equiv p$$

Commutative Laws

$$p \vee q \equiv q \vee p$$
$$p \wedge q \equiv q \wedge p$$

Associative Laws

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

Basic Logical Identities/Equivalences

Distributive Laws

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

De Morgan's Laws

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

Basic Logical Identities/Equivalences

Absorption Laws

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

Negation Laws

$$p \vee \neg p \equiv T$$
$$p \wedge \neg p \equiv F$$

Equivalences relating to Conditional Statements

$$p \to q \equiv \neg p \lor q$$
$$p \to q \equiv \neg q \to \neg p$$
$$p \land \neg p \equiv F$$
$$(p \to q) \land (p \to q) \equiv p \to (q \land r)$$
$$(p \to r) \land (q \to r) \equiv (p \lor q) \to r$$
$$(p \to q) \lor (p \to r) \equiv p \to (r \lor q)$$
$$(p \to r) \lor (q \to r) \equiv (p \land q) \to r$$
$$p \leftrightarrow q \equiv (p \to q) \land (q \to p)$$
$$p \leftrightarrow q \equiv \neg p \leftrightarrow \neg q$$
$$p \leftrightarrow q \equiv (p \land q) \lor (\neg p \land \neg q)$$

Basic Operators

- From the previous equivalences it is clear that AND, OR and NOT, what we call basic operators, can capture any truth table. (Verify)
- XOR, IFF, IMPLIES are called secondary operators.
- In fact from De Morgan's laws we can conclude that we do not need both OR and AND to capture all truth tables.
- $\neg(p \wedge q) \equiv \neg p \vee \neg q$ and
- $\neg(p \vee q) \equiv \neg p \wedge \neg q$
- This also will be needed: $\neg(\neg p) = p$.
- Using the logical identities you can show the equivalence of more complicated identities.

Building from basic identities

T.S.T. $\neg(p \to q)$ and $p \wedge \neg q$ are equivalent.

$$\begin{aligned}
\neg(p \to q) &\equiv \neg(\neg p \vee q) \\
&\equiv \neg(\neg p) \wedge \neg q \text{ (De Morgan Law)} \\
&\equiv p \wedge \neg q \text{ (Double Negation Law)}
\end{aligned}$$

Satisfiability

- A compound proposition is satisfiable if there is an assignment of truth values to its variables that makes it true.
- When no such assignments exists, the compound proposition is unsatisfiable.
- That is, there is no assignment of truth values that can make it true.
- A proposition is unsatisfiable iff its negation is a tautology.
- The assignment of truth values that makes a proposition true and therefore shows that it is satisfiable is called a solution of the satisfiability problem.
- To show unsatisfiability we need to show that every assignment of truth values to its variables makes it false.
- Using a truth table can be very tedious, we would prefer to reason out using identities.

SAT problem

- We are given a complicated proposition like the one below and asked to show if it is satisfiable or not.

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

- The general problem of deciding whether a proposition is satisfiable is called SAT.
- Build a truth table and see if a T appears – the problem with this approach is it grows exponentially with the number of variables.
- Is there an efficient solution to SAT?

SAT problem

- What do we mean by efficient? Something that grows polynomially in number of variables. That is something like n^{14} and not 2^n.
- No one knows the answer. In fact, if the answer is yes a lot of other problems will have efficient solutions!
- This may or may not be a good thing since cryptographic systems will break down.
- The problem of determining whether or not SAT has a poly-time solution is essentially the P versus NP problem.
- A very important open question in computer science, one of the 7 Millenium Problems.
- Solving which will mean you will win a million dollars.

SAT problem

- SAT-solvers is an active area of research.
- They find satisfying assignments with amazing efficiency even for formulas with millions of variables.
- But there are problems:
  - Which formulas work for SAT-solver methods is hard to predict.
  - For unsatisfiable formulas SAT-solvers can be less effective.