St. Francis Xavier University
Department of Computer Science

CSCI 544: Computational Logic
Lecture 1: Introduction
Winter 2022

1 Computers and Logic

Computers as we know them are relatively recent innovations in the history of humanity. Only a few decades ago were we able to harness electricity and machinery in such a way to allow us to perform calculations mechanistically. However, the underlying rules that govern the operation of this electricity and machinery to take us from an input question to a valid, correct output answer have existed since antiquity.

Formal logic (or just *logic*) is the study of reasoning and truth by way of rule systems. Logic is uniquely positioned at the intersection of computer science, mathematics, and philosophy: computers use logic to perform calculations, mathematicians use logic to study the properties and characteristics of abstract systems, and philosophers use logic to reason about knowledge and meaning. While the philosophical perspective on logic is interesting, in this course we'll focus on logic through the lenses of computer science and mathematics.

While the foundations of logic were largely built up from antiquity to around the nineteenth century by philosophers, mathematicians began to take a keen interest in logic around the late nineteenth century when the prevailing research focus turned to studying foundations of mathematics. Mathematicians such as George Boole, Augustus De Morgan, Gottlob Frege, Giuseppe Peano, and Charles Peirce published some of the earliest work on mathematical logic from the mid to late 1800s. As this body of work grew more popular, mathematicians began to feel concerned that mathematics had not been developed on a solid-enough foundation, and so mathematical logic quickly grew into a field of its own. This early work culminated in the publication of the first volume of *Principia Mathematica* in 1910 by Bertrand Russell and Alfred Whitehead: a rigorous attempt to formalize the foundations of mathematics using pure logic. Around the same time, David Hilbert developed his program to formalize all of mathematics according to a ground set of logical axioms. Unfortunately, not long after, Kurt Gödel would publish a paper titled *On Formally Undecidable Propositions of Principia Mathematica and Related Systems* in 1931, effectively showing that there are fundamental limitations preventing us from perfectly modelling mathematics using logic. While this dashed the hopes of Hilbert's program, it did leave the door open just enough for mathematicians to be able to formalize smaller or weaker mathematical systems.

Later in the twentieth century, as work on the foundations of mathematics began to lose steam and the notion of computing machines started to dominate more of the discourse, mathematical logic found new life in the work that would go on to form the foundations of computer science. Gödel's work would inspire and inform the work of early computer scientists, such as Alonzo Church, Stephen Kleene, and Alan Turing, as researchers dove into computability theory and recursion theory. Electrical engineers would join in on the fun, too, as electronic components that were capable of modelling logical operations and calculations were produced, combined, and miniaturized to form digital computers.

These days, logic is not strictly tied to computation. Informally speaking, logic appears in our everyday lives whenever we make a factual (or non-factual) claim. For example, the claim "it's cloudy outside" can be evaluated logically by simply looking out the window: if, indeed, it is cloudy at the time that claim is made, then the claim is true. Similarly, the claim "Beethoven's Symphony No. 9 was composed in C major" is false (since it was composed in D minor). On the other hand, claims like "StFX is a great school" cannot be evaluated logically, since the truth of such claims varies based on the opinion of the person evaluating that claim.[1]

[1] Obviously, asking any StFX student to evaluate this claim will result in overwhelming agreement that it is indeed true.

In the mathematical world, however, we don't deal in ambiguity and "gray areas". In mathematics, everything is ideally true or false, and logic can be used to determine definitively whether or not a given claim can be proved from a base set of axioms, or claims that we take to be true always. This is reflected in the (classical) computational world as well: bits are either on or off, 1 or 0, true or false. There is no in-between value. While this all-or-nothing approach might not perfectly reflect the state of the real world, mathematics and computer science have, in a sense, always been designed to deal in the abstract. Thus, for our purposes, we take this approach to be "good enough".

1.1 The Language of Logic

As we proceed in our study of logic, we'll make use of a wide array of vocabulary and symbols. You may have seen some of these words and symbols before, but for the purposes of review, we'll cover everything we need to start out in this section.

The claims that we referred to in the previous paragraphs are more accurately called *statements* or *propositions*. Every proposition has a truth value: *true* or *false*. We often denote propositions using variables like p, q, and r. If needed, we can access an infinite set of propositional variables by using subscripts: p_1, p_2, p_3, and so on.

Note that a proposition is *atomic*; it cannot be divided into any smaller component. If we were to connect multiple propositions together, we would create a *formula*.

To connect multiple propositions together to form a formula, we use a set of *logical connectives*. The most common logical connectives are as follows:

- The *negation* connective, $\neg p$, is a formula that is true when proposition p is false, and vice versa.
- The *conjunction* connective, $p \wedge q$, is a formula that is true when both propositions p and q are true, and false otherwise.
- The *disjunction* connective, $p \vee q$, is a formula that is false when both propositions p and q are false, and true otherwise.[2]
- The *conditional* (or *implication*) connective, $p \Rightarrow q$, is a formula that is false when p is true and q is false, and true otherwise.
- The *biconditional* (or *equivalence*) connective, $p \Leftrightarrow q$, is a formula that is true whenever p and q share the same truth value.

We may summarize the truth values of each logical connective in a *truth table*, where T denotes "true" and F denotes "false":

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Going beyond logical connectives, we can also introduce new symbols called *quantifiers* that allow us to evaluate not just whether or not a proposition is true, but how often the proposition is true:

- The *universal* quantifier, $\forall$, intuitively indicates that a proposition is always true.
- The *existential* quantifier, $\exists$, intuitively indicates that a proposition is true in some cases.

We will discuss logical connectives and quantifiers in much greater depth later, but for now it suffices to know that such symbols exist.

[2] Take care to remember that the disjunction connective is not exclusive; $p \vee q$ is true if p is true, q is true, or both p and q are true. Exclusive disjunction is a different logical connective.

2 The Big Three Pairs

In our study of logic, we'll often see concepts and notions that naturally go hand in hand: propositions and predicates, syntax and semantics, soundness and completeness. In this section, we'll briefly define and discuss each of these notions just to become acquainted with them in preparation for future lectures.

2.1 Propositions and Predicates

Just like in formal language theory, where we can consider different classes of languages depending on how much expressive power we desire, we can consider different "levels" of difficulty when it comes to logical systems.

At the most basic level, we focus purely on propositions and logical connectives, which gives us *propositional logic* or the *propositional calculus*. The simplicity of propositional logic makes it an ideal starting point for our study, since it's easy for us to reason about propositional logic and to implement ideas in propositional logic on a computer.

Going one step further, if we consider both logical connectives and quantifiers together with a special form of proposition called a *predicate*, then we get *predicate logic* or the *predicate calculus*. Predicate logic is sometimes referred to as *first-order logic*. A predicate is essentially a proposition in which we can change the value of one or more constants in the proposition. (You may now see why quantifiers are important: the truth value of a predicate may change depending on the value of the constants within.)

Given the name "first-order logic", you may (correctly) assume that we can go even further and introduce second-order logic, third-order logic, and so on. Indeed, it's possible for us to extend these notions—for example, by quantifying over relations instead of constants or variables—though our study of logic in this course will only take us as far as the first order.

2.2 Syntax and Semantics

In logic, it's important to distinguish what we write from what our writings mean. In the previous section, we defined a variety of symbols and variables and acribed to them certain names and meanings. In a way, these symbols and variables constitute a *formal language* that we can use to reason about logic. The specification of this formal language and its constitutent parts define the *syntax* of a logical system.

Going beyond plain definitions, we can talk about the *meaning* behind propositions. Depending on the logical system we're using, we can define a *model* wherein certain propositions are taken to be true. With this model, we can then assign truth values to other propositions. Broadly speaking, this defines the *semantics* of a logical system.

For each of the logical systems we'll discuss in this course, we'll discuss both the syntax and the semantics of the system in great depth. It's all the more important to consider both sides of this coin, since (for example) it's possible for some propositions to be equivalent semantically, but not syntactically. Being comfortable with both syntax and semantics will allow us to identify such situations and get as much use out of a logical system as we can.

2.3 Soundness and Completeness

Since the semantics of a logical system may vary depending on the model we choose to use, it's important for us to establish when exactly a proposition is true no matter what changes we make; in other words, we want to establish the property of truth based purely on logical form and nothing else. If we have a proposition that is true no matter the model we choose, then we say that proposition is *valid*. In a similar vein, if we take a collection of propositions S and another proposition t, and if t is true whenever all of the propositions in S are true (again, no matter the model we choose), then we say that t is a *logical consequence* of S.

In the context of a logical system, if we're able to show that some proposition t is a logical consequence of one or more propositions using the rules of that system, then we say the proposition t is *provable*. With these

notions of validity (or truth) and logical consequence (or provability), we're able to prove two important properties of logical systems:

- A logical system is *sound* if everything that is provable in the system is true. In other words, the logical system only proves valid propositions.
- A logical system is *complete* if everything that is true has a proof in the system. In other words, the logical system is able to prove every valid proposition.

Returning to our history lesson from the beginning of this lecture, the crucial blow that Gödel dealt to Hilbert's program was showing that any consistent logical system (that is, any system in which it is impossible to prove both a proposition and its negation at the same time) is incomplete. This result, known as Gödel's *first incompleteness theorem*, establishes that there exists a proposition in our system of arithmetic that can be neither proved nor disproved, defeating any chance of formalizing all of mathematics using logic.

Altogether, one of our goals in studying formal logic is to define and investigate logical systems that (i) are able to handle the notions of both validity and logical consequence, and (ii) are both sound and complete. Fortunately, for the systems we will study in this course, it's possible to achieve all of these goals.