What is Logic?

The study of logic has evolved for nearly 2,500 years, since Aristotle's work (350 BCE) and writings of the Stoics about the logos. For the Greeks, logic was the study of correct argument expressed in a syllogism. Leibniz (1666) explained the potential of logic to create a universal language for codifying human knowledge and mechanically resolving scientific and legal questions. He accepted the idea that the mind works by computation. Boole built on those ideas and created a calculus of logical operations in 1854, now known as Boolean algebra. We will start the course with a modern version of his calculus, called propositional logic. This logic expresses one of the deepest open questions in mathematics and computer science, finding the intrinsic computational difficulty of determining whether a formula in propositional logic can be true. This is the famous P=NP question, one of the seven Millennium Prize Problems in mathematics worth one million dollars.

Boole believed that his calculus captured many of the laws of thought, and while his was a very mathematical analysis, it was inadequate to express much of the reasoning actually used day to day in mathematics. His calculus was however well suited to describe hardware circuits and has been used for decades in industrial circuit design. In 1879 Frege created the predicate calculus as a way to avoid errors in mathematical reasoning and proof. His calculus extends Boole's and is the basis of modern logic. It is a central part of all logic courses because it expresses most mathematical arguments as well as reasoning used in science and law. In this course the topic is called first-order logic, the title of the required textbook as well. Most logic courses teach students how to express ideas in first-order logic, and this logic is also used to study the semantics of natural language and to investigate some of the paradoxes about truth in natural language such as the meaning of the assertion "This sentence is false", a version of the Liar paradox. Some paradoxes lead to contradictions others have led to extensions of logic.

Russell created a logical theory to express all mathematical concepts based on Frege's analysis. His logic, called type theory, was also designed to avoid Russell's paradox, a paradox which arose in a certain application of Frege's logical foundation for arithmetic and lead to a contradiction. Russell and Whitehead used type theory in their famous book Principia Mathematica published in 1910 and purporting to provide a rigorous contradiction-free foundation for all mathematics. In 1931, Godel proved a remarkable incompleteness theorem that neither Principia Mathematica nor any equivalent theory could prove all true statements of the theory unless it was contradictory. Godel's proof used a kind of self reference seen in the Liar paradox, but in a non-contradictory way. This major discovery raised philosophical questions about the limits of human knowledge that remain with us today, as do various logical paradoxes, some already known to the

Greeks. Godel's result can be interpreted to say that no machine can know all truths of this theory, but could a human know them? Are minds more potent than machines? These issues became more urgent with the advent of powerful digital computers and the possibility of machine intelligence. We will examine such questions when we study type theory. Church simplified Russell's type theory to create a version called simple type theory -- also called higher-order logic. We will study higher-order logic in the course. Godel's theorem applies to this logic as well.

Church and Turing in 1936 laid the foundations for computer science by defining equivalent notions of computability – Church for software, Turing for hardware. Their ideas were used to make precise the insights of Brouwer from 1900 that mathematics is based on fundamental human intuitions about numbers and human computation. This use of logic was able to shed light on the long standing philosophical question about the basis for mathematical truth and role of computation in human thought, a question already studied by Leibniz. In particular, Turing's work explained the core of Godel's famous incompleteness theorem, and it led to his predictions about machine intelligence and his famous Turing test for intelligence. He agreed with Leibniz that the mind works by computation. On the other hand, Turing's work raised the question of whether human computation ("effective computation") is the same as mechanical computation. The Church/Turing Thesis is that they are the same. Brouwer disagreed. We will present a modern version of Turing's explanation of Godel's theorem in the last section of the course. The ideas that shed light on these important philosophical issues have also been extremely useful in computer science for specifying properties of software systems and proving them as we will show in the topic called Computational Type Theory.

Major Course Topics

Propositional Logic

First-Order Logic

Higher-Order Logic

Computational Type Theory