Discrete Mathematics (II) Spring 2013

Lecture 3: Introduction to Logic

Lecturer: Yi Li

1 Overview

What is a mathematical proof? How can proofs be justified? are there limitations to provability? To what extent can machines carry out mathematical proofs?

We do also concern about what is the relationship between logic and computer science. Logic is profound and abstract, even much more abstract than abstract algebra. Finally, we do care about what we can learn from this course.

In this lecture, we first give a brief history of mathematical logic. Then we define a partial order and represent tree in an approach other than the way in last semester.

2 Brief History of Logic

The history of logic is very long. We can even trace back to the era before systematic arithmetic occurred. Aristotle(384-322 B.C.E.) introduced theory of syllogistic, the source of modern logic. Before 19th century, logic was the garden only for philosophers.

From then on, logic was enriched and formalized, by mathematicians, such as De Morgan (1806-71), Boole (1815-64), and Schröder (1841-1902). Frege (1848-1925) made logic systematic and helpful to mathematics. He then established a connection between arithematics and logic. In the late of 1900s, logic became the powerful tool to expel the root of the third mathematical crisis. From now on, logic is the world of mathematicians.

In this course, we only introduce essential part of first order logic, which is the corner stone of whole logic building. Furthermore, it consists of propositional logic and predict logic, the former can be emmbedded into the latter.

Based on first order logic, we can introduce seconder order logic and even much higher order logic as you wish. There are also some other type of logic system, e.g., modal logic, intuitionistic logic, and temporal logic. Classic logic only has two value 0 and 1. There is also a system called multiple-value logic.

Mathematical logic is aim to formalize the pattern of human deduction but not completely. It can be divided into two threads: syntax and semantics. Post(1897-1954) found the connection between syntax and semantics. He proved completeness and soundness theorem in propositional logic. Gödel(1906-78) proved completeness in predicate logic, which was a very important contribution. Herbrand(1908-31), who disliked existences proof, gave a construct method. Henkin tried to find a model for a given set of sentences.

Several proof systems have been introduced for different reasons. However, they have the same

function and are actually equivalent. Axiomatic system is very traditional and occurs in the most of textbook. Robbinson put up with resolution method, which continued the work of Herbrand. Beth and Smullyan construct a tableaux proof system. It is the simplest proof system and is easy to use.

Leibniz had a dream to construct a machine to find all theorems. Hilbert followed him and formalized many mathematical branch in his era. But Gödel ruined their dream by his famous incompleteness theorem. This course just covers a very small part of provable topics.

Logic aims to formalize statements and relationship between them. Actually, a programming language can also be taken as a type of formal language. Then they both follow the same approach. So this course is another training about programming. It can go further. If you are unlucky, you will learn computability, decidability, and enumerability, which is the theory of computation, the base of computer science.

3 Introduction to logic

In remaining classes, first order logic will be introduced. First order logic is composed of two parts: propositional logic and predicate logic. Propositional logic is much simpler than predicate logic. However, propositional logic sets up a framework. Predicate logic is more powerful than propositional logic. It can express much more complicated scenarios. But it still follows the same framework. A fact will be proved that propositional logic can be embedded into predicate logic as a very special form.

We will learn logic system from two aspects. One is syntax and another one is semantics. The syntax part consists of form of proposition or sentence and the proof system. Proof system defines how you can deduce some consequences from a set of sentences with some rules. There are three typical proof system. Axiom proof system is very classic, which is widely adopted by textbook for mathematical student. Resolution system is a foundation of every logic programming language, e.g. prolog. We will use tableaux method. It is more vivid and easier to use. If a sentence is false, you can even construct a counterexample based on you proof procedure.

The semantics of a proposition or a sentence means whether it is true or false. Herbrand characterizes a set of sentences in semantic approach. It finally formed into model theory. However you need not worry about the semantics of sentences while using proof system. Actually we hope that every result proved by proof system should be right and a right result should be provable.

During early study, two sides are split and taught in an twisted way. Is there any connection between them? It is sure. Completeness theorem guarantees that it has a proof if a sentence holds. Soundness theorem assure the validity of the result you derived from a proof procedure. Finally a very beautiful result, compactness theorem, will be proved.

In first order logic it has only two value: true or false. Some logic system may extend to multiple values. Once you master first order logic, it is not difficult for you. Similarly, more complicated logic system are coined for some purpose, including high order logic, modal logic, intuitional logic, and temporal logic.

Logic is developed to fix the paradox resulted from set theory, which is the foundation of modern mathematics. It tries to formalize statement in order to eliminate ambiguity and to make proof

more rigorous. In this course, you should learn how to formalize a problem and make your thinking rigorous.

4 Order

Logic for application adopts a special set of its own notations. For convenience to read text book, we just follow the terminology in text book. Then, we need change the definition of partial order, which are slightly different with the one learned in last semester. And tree is also a very important tool to our class. And it is totally represented in an order approach, as a partial order.

In *Logic for application*, *partial order* is defined as following:

Definition 1 (Partial order)**.** *A* partial order *is a set S with a binary relation $<$ on S, which is* transitive *and* irreflexive*.*

You should mind that partial order is *irreflexive* other than reflexive defined in previous semester and the most of text book on set theory.

Whatever the change of partial order, *linear order* is always the same.

Definition 2 (Linear order)**.** *A partial order $<$ is a* linear order*, if it satisfies the* trichotomy *law: $x < y$ or $x = y$ or $y < x$.*

As we know, there are many partially ordered sets. For further investigation, we can divide them into two categories, something bad and something good. Here, we define what is good as following.

Definition 3 (Well ordering)**.** *A linear order is* well ordered *if every nonempty set A of S has a least element.*

The set $\mathcal{N}$ is a reference of countable set. *countable* is defined base on set of natural number.

Definition 4 (Countable)**.** *A set A is* countable *if there is a one-to-one mapping from A to $\mathcal{N}$.*

As a rational number can be represent as a pair of two natural number. $\mathcal{Q}$ is a set of countable elements, which also contains natural number as a very small part of it.

Specially, when we can count a set clearly, here we mean a exact number, we define the following concept.

Definition 5 (Finite)**.** *A set A is* finite *if there is a one-to-one mapping from A to $\{0, 1, \ldots, n-1\}$ for some $n \in \mathcal{N}$.*

In this semester, finite is our favor and unfortunately evil too. As a student who study computer area, here we just mean building machine or coining software, infinite number of objects is always out of our control.

With these two, we can define its negative side.

Definition 6. *1. If A is not countable, it is* uncountable*.*

2. *If A is not finite, it is* infinite.

The following theorem is a base on syntax of logic.

Theorem 7. *Let A be a countable set. The set of all finite sequence of elements in A is also countable.*

Proof. We can formalize it as $S = \cup_{n\in\mathcal{N}} A^n = A^1 \cup A^2 \cup \cdots \cup A^n \cup \cdots$. Then we can construct a mapping from A^n to $\mathcal{N}$. □

5 Tree

In the last semester, we have learned tree in the way of graph. And we have already known that a relation and a graph can represent each other. Here, we will represent a tree with a partial order.

We can define a tree based on a partial order as follows:

Definition 8 (Tree)**.** *A tree is a set T(whose elements are called nodes) partially ordered by $<_T$, with a unique least element called the root, in which the predecessors of every node are well ordered by $<_T$.*

From now on, we will redefine the concepts and discuss its properties in an order approach.

Definition 9 (Path)**.** *A path on a tree T is a maximal linearly ordered subset of T.*

Definition 10 (Properties of tree)**.** 1. *The levels of a tree T are defined by induction.*

2. *The 0^{th} level of T consists precisely of the root of T.*
3. *The $k+1^{th}$ level of T consists of the immediate successors of the nodes on the k^{th} level of T.*
4. *If each node has at most n immediate successors, the tree is n-ary or n-branching.*
5. *If each node has finitely many immediate successors, we say that the tree is finitely branching.*
6. *A node with no successors is called a leaf or a terminal node.*

Theorem 11 (König's lemma)**.** *If a finitely branching tree T is infinite, it has an infinite path.*

sketch. If there is no infinite path, the tree would be finite. Split the successors of the node into two parts. One with infinite successors and the other with finite successors. □

6 Extensions on Tree

Definition 12 (Segment)**.** 1. *σ is an initial segment of τ if $\sigma \subset \tau$ or $\sigma = \tau$.*

2. *σ is an proper initial segment of τ if $\sigma \subset \tau$.*

Definition 13 (Lexicographic ordering)**.** *For two sequences σ and τ we say that $\sigma <_L \tau$ if $\sigma \subset \tau$ or if $\sigma(n)$, the n^{th} entry in σ, is less than $\tau(n)$ where n is the first entry at which the sequences differ.*

Given a tree, we can define a linear order like this, which is also a special type of lexicographic ordering.

Definition 14 (left to right ordering)**.** *Given two nodes x and y,*

1. *If $x <_T y$, we say that $x <_L y$.*
2. *If x and y are incomparable in the tree ordering, find the largest predecessors of x and y, say x' and y'*
   (a) *If x' equals y', $x \leq y$ if and only if x is left to y relative to x'.*
   (b) *Otherwise $x <_L y$ if and only if $x' <_L y'$.*

A algorithmic approach to determine two elements x and y will be given in the future. And it is left as an exercise.

Exercise

Hints: Tree is now a set T with partial order $<_T$. You should prove the problem based on this definition other its definition in Graph theory.

1. Show that antisymmetric can be implied by new definition of partial order.
2. Given an example of a finitely branching tree that is not n-branching for any n.
3. Prove that the notion of the level of a node in a tree is well defined, i.e., every node of a tree T is on exactly one level.
4. Prove that every node of a tree other than the root has exactly one immediate predecessor.
5. Consider the set of all finite sequences of natural numbers. Define an ordering $<$ as follows: $\sigma < \tau$ iff(if and only if) either σ is shorter than τ or, if not, $\sigma <_L \tau$, which means that $\sigma \subset \tau$ or $\sigma(n)$, the n^{th} entry in σ, is less than $\tau(n)$ where n is the first entry at which the sequences differ. Prove that $<$ is a well ordering.
6. Prove that $\mathcal{Z}$, all integer number, with $<$ is not well ordered. Define a order $<'$ such that $< \mathcal{Z}, <'>$ is well ordered.
7. Find some ways to make a tree linear other than lexicographic ordering.
8. Given two incomparable elements x and y, give an algorithm to determine their order.