| | |
|---|---|
| **Applied Logic** | **Lecture 1** |
| **CS 486 Spring 2005** | **Tuesday, January 25, 2005** |

## What is Logic?

What is logic, the topic of this course? There are at least two answers to that question.

- General logic: critical examination of how we acquire "scientific" knowledge, and of that knowledge itself, principles, general laws, and theories.
- Formal logic: study of sound "reasoning", based on form more than content.

In this course, we focus on this section definition of logic. Formal logic goes back to the philosopher Aristotle (384–322 BC), who studied logic in the form of syllogisms, such as the following standard example:

> All men are mortal,
> Socrates is a man;
> Therefore, Socrates is mortal.

If we the two premises (all men are moral, and Socrates is a man) are true, then we are forced, if we are rational, to acknowledge that the conclusion (Socrates is mortal) is also true.

This view of logic remained essentially unchanged well past the Renaissance.

In the 19th century, De Morgan (1806–1876) and Boole (1815–1864) started to think of logic as an "algebra of mental operations". This led to the mathematization of logic, the study of logic using mathematical techniques. This modern variant of formal logic is often called symbolic logic, logistic, or mathematical logic.

## Applications

Beyond its use as a study of reasoning, logic has uses in many fields.

(1) In Mathematics, logic is often used for the study of the foundations of the field.

- With Russell and Whitehead at the beginning of the 20th century, there was this hope to reduce all of mathematics to logic (*Principia Mathematica*). While this project arguably failed, they managed to capture a lot of mathematics using essentially logical ideas. This led to the study of Set Theory as a foundation of mathematics, a still active area of research.
- The search for an "algorithmic" decision procedure for determining the true statements in the logic of *Principia Mathematica* gave birth to Computer Science: it forced mathematicians to examine what exactly one means by an algorithm and a mechanical computation (recursive functions and Turing machines are two frameworks that came out of those investigations).
- A field of logic called Model Theory has applications to mainstream mathematical questions, as recent results on abstract algebra questions using model-theoretic techniques have shown.

(2) In Computer Science, we generally deal with computational issues associated with logic.

- Automated verification of systems and software.
- Logic can be used as a programming language (logic programming, Prolog).
- Many uses in AI: reasoning systems, robots.
- Logic is intimately connected with *computation*. A deep result, the Curry-Howard isomorphism, says, very roughly, that to every formula corresponds a type, and that a proof of a formula corresponds to a program that computes a value of the type corresponding to the formula.

(3) In Philosophy, where logic was first studied, it remains central to much of modern analytic philosophy.

- Study of other forms of sound reasoning, such as inductive reasoning.
- Connection between logic and language; understanding the meaning of an English sentence is informed by logic. Epimenides the Cretan said: "All Cretans are liars". How is one supposed to understand this sentence, whose meaning is naively paradoxical?

In this course, we study logic from a Computer Science perspective, that is, with an eye towards computational issues (procedures for deciding truth of formulas, proof methods, etc).

## Propositional Logic

Propositional logic is, in some sense, the simplest logic. It captures reasoning with the basic logical connectives *not*, *and*, *or*, and *implies*.

Compare the following English sentences:

(1) A foo is a bar.

(2) If a foo is both a bar and a baz, then a foo is a bar.

Is the first sentence stating a truth? It's hard to tell, because, intuitively, it depends on what foos and bars are. In contract, the second sentence seems to be stating a truth, despite us not knowing what foos, bars, and even bazes are. Why is this? This is what propositional logic is meant to help answer.

To study logic, rather than using English, we use an abstraction, in the form of a *formal language*.

The language of propositional logic is built from the following symbols:

$\neg$ (read "not"; sometimes written $\sim$)

$\wedge$ (read "and"; sometimes written $\&$)

$\vee$ (read "or")

$\Rightarrow$ (read "implies"; sometimes written $\supset$)

We also assume a denumerable set $PVar = \{p_1, p_2, p_3, \dots\}$ of propositional variables. (I will generally use $p, q, r, s, \dots$ to range over propositional variables.) Additionally, the symbols $($ and $)$ are used as delimiters.

The *formulas* of propositional logic (roughly, the sentences of the language) are given by the following inductive definition:

- A propositional formula $p$ is a formula.
- If $\varphi$ is a formula, then $\neg\varphi$ is a formula.
- If $\varphi$ and $\psi$ are formulas, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, and $(\varphi \Rightarrow \psi)$ are formulas.

I use $\varphi, \psi, \dots$ to range over formulas. I will often drop the outer parentheses around a formula, since there is no ambiguity in that case.

**Example 1** If we read the propositional variable $p$ as "a foo is a bar" and the propositional variable $q$ as "a foo is a baz", then the formula

$$(p \wedge q) \Rightarrow p$$

represents the English sentence (2) above. □

The following unique decomposition lemma, taken from Smullyan, is needed with the above definition of formulas to ensure that we can always tell, for a given formula, how it is built up from "smaller" formulas.

**Lemma 2** *For every formula $\varphi$, one and only one of the following propositions hold:*

*(a)* *$\varphi$ is a propositional variable.*

*(b)* *There is a unique formula $\psi$ such that $\varphi$ is $\neg\psi$.*

*(c)* *There are unique formulas $\psi_1$ and $\psi_2$ and a unique binary connective $b$ such that $\varphi$ is $(\psi_1\ b\ \psi_2)$.*

*Proof.* Left as an exercise. $\square$

There is an alternative approach to defining formulas that does away with the need for a unique decomposition lemma, which is more in the CS spirit.

First, we define the concept of a *labeled disjoint union*.

Define $l{:}A + r{:}B$ as the set of all pairs of $\langle l, a\rangle$ (for $a \in A$) and $\langle r, b\rangle$ (for $b \in B$). Intuitively. we tag every element of $A$ with an $l$, and every element of $B$ with a $r$. Thus, giving an arbitrary element of $l{:}A + r{:}B$, we can always tell whether if it is an element of $A$ (if it is tagged by $l$) or an element of $B$ (if it is tagged by $r$). This construct obviously generalizes to more than two "summands".

With this in mind, we can define the set of formulas $\mathit{Form}$ as:

$$\begin{aligned}\mathit{Form} = &\mathit{var}{:}\mathit{PVar} + \mathit{not}{:}\mathit{Form} + \mathit{and}{:}\mathit{Form} \times \mathit{Form} + \\ &\mathit{or}{:}\mathit{Form} \times \mathit{Form} + \mathit{imp}{:}\mathit{Form} \times \mathit{Form}.\end{aligned}$$

Thus, a typical element of $\mathit{Form}$ is $\langle \mathit{var}, p\rangle$, or $\langle \mathit{and}, \varphi, \psi\rangle$, where $\varphi$ and $\psi$ are themselves in $\mathit{Form}$.

**Example 3** Returning to the foo-bar-baz example sentence (2) above, if we once again interpret $p$ as "a foo is a bar" and $q$ as "a foo is a baz", then we can rewrite the sentence as the formula

$$\langle \mathit{imp}, \langle \mathit{and}, \langle \mathit{var}, p\rangle, \langle \mathit{var}, q\rangle\rangle, \langle \mathit{var}, p\rangle\rangle.$$

$\square$

If you are familiar with the concept, not that this really just defines an Abstract Syntax Tree of formulas.

Formula decomposition can now be done via a *case expression*, as

```
**case** $\varphi$ **of**
    $\langle var, p\rangle \longrightarrow \ldots p \ldots$
    $\langle not, \psi\rangle \longrightarrow \ldots \psi \ldots$
    $\langle and, \psi_1, \psi_2\rangle \longrightarrow \ldots \psi_1, \psi_2 \ldots$
    $\langle or, \psi_1, \psi_2\rangle \longrightarrow \ldots \psi_1, \psi_2 \ldots$
    $\langle imp, \psi_1, \psi_2\rangle \longrightarrow \ldots \psi_1, \psi_2 \ldots$
**end**.
```

We will see examples of this construct next class, when we define valuations for formulas.

Unless specified otherwise, we will take formulas as being defined using the second definition above, but we will write them using the original notation in terms $\neg$,$\wedge$,$\vee$,$\Rightarrow$. This is not a problem, as long as we remember that $(\varphi \Rightarrow \psi)$ is just an alternative notation for $\langle imp, \varphi, \psi\rangle$, for a suitably translated $\varphi$ and $\psi$.