Equational Logic and Term Rewriting: Lecture I

Logic Summer School

Ranald Clouston
Australian National University
5–9 December 2011

Why so many logics?

You all know classical propositional logic. Why would we want anything more?

One reason is that we might want to change some basic logical property, such as our possible truth values:

- With classical logic, propositions are either true or false;
- With fuzzy logic, propositions might be true, false, or any number of values in between!

Alternatively, we might have noticed that some piece of grammar (from natural language, say, or a computer programming language) has logical structure. We then define a logical connective for this piece of grammar, along with proof rules to reason about it, and perhaps mathematical semantics to precisely express its meaning.

Logical connectives

Once you start searching for grammar with logical structure, you find it all over the place.

- Example: modal logic, which you’ll learn about from Wednesday, is about grammar that modifies propositions: “certainly”, “possibly”, “eventually”, “I believe that”...

We end up with a whole zoo of logical connectives - you need to pick and choose!

Over the next five lectures, we’re going to make things a bit easier by discussing a logic with one connective only:

The equals sign

Equational logic - not so easy

In fact, this logic with equals only – equational logic – is quite interesting:

- It’s powerful enough to express propositional logic - both classical and intuitionistic varieties;
- It underlies the mathematical field of universal algebra;
- Many mathematicians spend their career studying subfields of universal algebra, such as group theory;
- It’s used as the basis of programming and specification languages;
- The proof rules seem simple, but equational logic is undecidable;
- The most popular algorithm for reasoning about equational logic, term rewriting, is one of the most prominent topics in computational logic.
Abstract and Universal Algebra

The mathematical field of abstract algebra tries to abstract out the underlying algebraic ‘rules’ of various mathematical systems.

For example, how is multiplication of real numbers like multiplication of matrices? Can we apply the same algebraic manipulations to both, or are different rules needed?

Once we have found these rules, we can prove theorems using only these rules. Because this point of view is so abstract, our theorems might then apply to a whole bunch of very different mathematical structures!

Often (but not always) the rules can be presented in the form of equations. The main tool for reasoning about them is then equational logic.

The study of mathematical structures defined by equations is called universal algebra.

Example I - Monoids

A monoid is a set \( M \) equipped with a distinguished element \( e \in M \) and a function \( \cdot : M \times M \to M \)...

(For readability, we’ll write \( \cdot (x, y) \) infix as \( x \cdot y \).)

... such that the following equations hold:

\[
\begin{align*}
(x \cdot y) \cdot z &= x \cdot (y \cdot z) \quad (\text{Associativity}) \\
e \cdot x &= x \quad (\text{Left identity}) \\
x \cdot e &= x \quad (\text{Right identity})
\end{align*}
\]

- Example: numbers (\( \mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R} \), etc.) with addition - \( e \) is 0.
- Example: numbers with multiplication - \( e \) is 1.
- Non-example: numbers with subtraction - not associative, no left identity.

Example II - Groups

A group is a monoid \( (G, e, \cdot) \) with an extra function \( \cdot^{-1} : G \to G \)...

(We’ll write \( \cdot^{-1}(x) \) postfix and superscripted as \( x^{-1} \).)

... obeying the additional equations:

\[
\begin{align*}
-x^{-1} \cdot x &= e \quad (\text{Left inverse}) \\
x \cdot x^{-1} &= e \quad (\text{Right inverse})
\end{align*}
\]

(This gives a five equation definition. In fact there exists an equivalent three equation definition, but this will do for now).

- Example: \( \mathbb{Z}, \mathbb{Q}, \mathbb{R} \) with addition - \( x^{-1} = -x \).
- Example: \( \mathbb{Q}, \mathbb{R} \) with multiplication - \( x^{-1} = \frac{1}{x} \).
- Non-examples: \( \mathbb{N} \) with addition; \( \mathbb{N}, \mathbb{Z} \) with multiplication.

What about commutativity?

All the examples we’ve seen so far also obey the equation:

\[
x \cdot y = y \cdot x \quad (\text{Commutativity})
\]

This isn’t part of the standard definition because there are interesting examples where it doesn’t hold, e.g. matrix multiplication.

A monoid with commutativity is called (surprise!) a commutative monoid.

A group with commutativity is called an Abelian group, after Neils Henrik Abel (1802-1829), one of the founders of group theory:
**Example III - Lattices**

A lattice is a set $L$ equipped with a partial order (reflexive, transitive, antisymmetric relation) $\leq \subseteq L \times L$, where every finite set of objects has a greatest lower bound (or meet), and a least upper bound (or join).

On the left we have a lattice; on the right we do not:

![Lattice Example](image)

It turns out we can specify this lattice requirement exactly via equations!

**Example IV - Heyting Algebras**

A Heyting Algebra (after Arend Heyting (1898-1980)) is a mathematical model for intuitionistic propositional logic, defined via equations.

Take a lattice $(H, \land, \lor)$ and add two distinguished elements $0, 1 \in H$, a function $\to : H \times H \to H$ written infix, a function $\neg : H \to H$, and the equations:

- $x \land 1 = x$ (Identity / ‘Top’ of the Lattice)
- $x \lor 0 = x$ (Identity / ‘Bottom’ - defines a bounded lattice)
- $x \land (x \to y) = x \land y$
- $y \land (x \to y) = y$
- $x \to (y \land z) = (x \to y) \land (x \to z)$ (equations for implication)
- $\neg x = x \to 0$ (definition of negation)

**Example V - Boolean Algebras**

A Boolean Algebra (after George Boole (1815-1864)) is a mathematical model for classical propositional logic, got by adding to Heyting Algebra the single equation

$$\neg \neg x = x$$

or equivalently,

$$x \lor \neg x = 1$$

There are other ways to present Boolean Algebra as well (if you look up some textbooks and Wikipedia, you’ll find a range); they should all be equivalent, although this might not be immediately obvious!

Hopefully, you are persuaded by now that equations can define some pretty interesting structure. But what exactly is equational logic?
Signatures

Before we get to equations, we need to ask what the things are that we are equating. To get to them, we need to introduce signatures.

A signature $\Sigma$ is a pair $(F_\Sigma, |\cdot|_\Sigma)$, where

- $F_\Sigma$ is a set, whose elements we call function symbols;
- $|\cdot|_\Sigma$ is a function $F_\Sigma \to \mathbb{N}$, called the arity function.

We'll generally leave the subscripts off the arity function.

If $|f| = 0$ then $f$ is a constant; if 1 then $f$ is unary; if 2 then binary, etc.

e.g. the signature $G$ for groups has $F_G = \{ e, \cdot, -1 \}$ (for identity, multiplication, and inverse), with $|e| = 0$, $|\cdot| = 2$, and $|-1| = 1$.

Terms

Say we have a signature $\Sigma$ and an infinite set $\text{Var}$ of variables.

Then the $\Sigma$-terms are defined by
t :: = x | f t ... n terms

Both look like gibberish - how can we tell which is acceptable?

Parsing Terms

The $G$-term $\cdot \cdot -1 x -1 y x$ parses unambiguously as

The putative $G$-term $\cdot \cdot -1 x -1 y$ doesn't parse at all (go ahead; try!).

Our well-defined term still isn't very attractive - we'll often use infix and postfix notation, operator precedence and brackets, to get something more like

$$(x^{-1} \cdot y^{-1}) \cdot x$$

Equations and Theories

An equation over a signature $\Sigma$ is simply a pair of $\Sigma$-terms with the symbol $\approx$ between them:

$$t \approx t'$$

(From now we'll use the squiggy $\approx$ for the logical connective 'equals', and the normal '=' for 'real' equality.)

A $\Sigma$-theory is simply a set of equations over $\Sigma$. We call equations in this context axioms.
A Note on Equational Theories

One simple but important point to make is that we are only interested in equational logic in the presence of a theory.

Why? Because as you might imagine, \( t \) and \( t' \) will be provably equal in the ‘empty theory’ if and only if they’re literally the same term - not a very interesting property.

Conversely, if we introduce equational axioms for, say, groups, we get a situation interesting enough to have sustained generations of mathematicians for almost two hundred years!

This isn’t true of all logics - it is genuinely interesting that, say, \((p \land \neg p) \rightarrow q\) is true in the ‘empty theory’ for propositional logic.

Structures

So much for syntax; what are our semantics?

A \( \Sigma \)-structure \( M \) is a pair \( (M, M[\llbracket - \rrbracket]) \), where

- \( M \) is a set, called a carrier set;
- \( M[\llbracket - \rrbracket] \) is a map, called an interpretation, from function symbols in \( F_{\Sigma} \) to functions on \( M \) so that
  \[
  |f| = n \Rightarrow M[f]:M^n \rightarrow M.
  \]

We’ll usually write \( M[\llbracket f \rrbracket] \) as \( \llbracket f \rrbracket \).

If \( f \) is a constant then its interpretation \( \llbracket f \rrbracket \) is an element of \( M \); if \( f \) is unary then \( \llbracket f \rrbracket \) is a function \( M \rightarrow M \), and so forth.

Structures ctd.

Let’s try to find a structure for the group signature \( G \), where the carrier set is some two element set \( \{a, b\} \).

\( \llbracket e \rrbracket \in \{a, b\} \), so let’s arbitrarily set it to be \( a \).

\( \llbracket -1 \rrbracket \) is a function \( \{a, b\} \rightarrow \{a, b\} \); there are four to choose from. Only one will end up giving rise to a group (which one?) but for the moment we don’t care about the equational axioms so any will do.

\( \llbracket \cdot \rrbracket : \{a, b\} \times \{a, b\} \rightarrow \{a, b\} \) is best presented as a Cayley table (after Arthur Cayley (1821-1895)), e.g.

<table>
<thead>
<tr>
<th>[]</th>
<th>(a)</th>
<th>(b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>(a)</td>
<td>(b)</td>
</tr>
<tr>
<td>(b)</td>
<td>(b)</td>
<td>(a)</td>
</tr>
</tbody>
</table>