UNIVERSITY OF CAMBRIDGE

Introductory Logic

Lecture 1: Background Knowledge – Revision

Alan Mycroft

Computer Laboratory, University of Cambridge, UK
http://www.cl.cam.ac.uk/~am21

MPhil in ACS – 2011/12

Course outline

- A primer course on mathematical logic to prepare students for research in Theoretical Computer Science.
- Concentrates on logic for modelling. Model theory rather that proof theory.
  (Compare the course on Automated Reasoning which—put crudely—uses logical formulae as a data structure in a tool and inference steps as syntactic manipulations of this).
- E.g. did you know there are countable models of the reals?
- Book: Enderton “A Mathematical Introduction to Logic" (2nd ed.).

Lecture Outline

- Sets, set operations
- Cardinality, infinities, countability
- Effectiveness, recursive enumerability, recursive sets

Sets

A *set* is a collection of elements. Notation:

- $\{1, 4, 9\}$
- $\{5, \{7, 8\}, \{\{2\}\}, -57.34\}$
- $\{0, 1, 2, 3, \ldots\}$ (be careful here)
- $\{\}$ (a.k.a. $\emptyset$).
- $\{x \mid P(x)\}$ where P is a formula (see later) involving x.

Just saying sets are like this is *wrong*. Russell's Paradox.

- Consider $S = \{x \mid x \notin x\}$. Then ask "Is $S \in S$?" (contradiction whether you suppose yes, or no).
- There is no "set of all sets" even if there is a set of all real numbers.

Be careful when saying things formally!

Sets more carefully

To avoid Russell's paradox we need to be more careful.

- Finite sets, written down are OK, e.g. $\{1, 4, 9\}$.
- Unless you're very formal then writing $\{0, 2, 4, 6, \ldots\}$ is OK. Note that this is an infinite set, all of whose elements are finite (but unbounded).
- Sets can be made *from other sets* by construction. So $\{x \in S \mid P(x)\}$ is always when S is already a set.

Define relations on sets:

- $X \subseteq Y$ means (for all x) $x \in X \Rightarrow x \in Y$.
- $X \supseteq Y$ means $Y \subseteq X$
- $X = Y$ means $X \subseteq Y$ and $X \supseteq Y$.

At the moment 'means', 'forall', '$\Rightarrow$', 'and' are rather informal. We'll use symbols $\wedge$, $\vee$, $\neg$, $\forall$, $\exists$, $\Rightarrow$ but only define them later.
Note that for all sets X we have $\emptyset \subseteq X$ (vacuous reasoning).

Definition and Equality

Beware the use of '$=$' both as equality and as a definition. E.g. the following is ugly:

- $(X = Y) = (X \subseteq Y \wedge X \supseteq Y)$

You might prefer 'iff' (if and only if) for definition for things which are true or false. (Other symbols include $\stackrel{def}{=}$, $\stackrel{\Delta}{=}$ especially on values or sometimes $\equiv$).

But the key idea is that equality '$=$' is an operator within the formal system I'm talking about, whereas definition 'iff' is part of the human language we use to talk about the system. (Later we will say object- and meta-language).

Operations on sets

- $X \cap Y = \{x \mid x \in X \wedge x \in Y\}$
- $X \cup Y = \{x \mid x \in X \vee x \in Y\}$
- $X \setminus Y = \{x \mid x \in X \wedge x \notin Y\}$ (some authors use $X - Y$).
- $X \times Y = \{(x, y) \mid x \in X \wedge y \in Y\}$ (cartesian product); this also explains e.g. $\mathbb{R}^3$ for 3-tuples of reals.
- $X + Y = X \times \{0\} \cup Y \times \{1\}$ (disjoint union)
- $X \to Y$, the set of functions from set X to set Y (see later)
- $\mathcal{R}(X, Y)$ (slightly non-standard), the set of relations between X and Y.
- $\mathcal{P}(X) = \{Y \mid Y \subseteq X\}$ (power set)

We do *not* define X^c the complement of X (all the things not in X) unless it's very clear that this is really $U \setminus X$ for some local 'universe' of discourse.

‘Big’ operations on sets

We also have iterated operations on sets: $\bigcap$, $\bigcup$.
Recall

$$\sum_{i=1}^{i=n} i = \sum_{i \in \{1,\ldots,n\}} i = \frac{n(n+1)}{2} \qquad \text{noting} \qquad \sum_{i \in \{\}} i = 0$$

When $\mathcal{S}$ is a set of sets, we similarly have

- $\bigcup \mathcal{S} = \{x \mid x \in \text{ some } X \in \mathcal{S}\}$.
- $\bigcap \mathcal{S} = \{x \mid x \in \text{ every } X \in \mathcal{S}\}$. This second case is only valid when $\mathcal{S} \neq \{\}$ (because otherwise this would be the set of all sets which doesn’t exist).

Notation: we generalise this notation (as in $\sum$ above) to e.g.

$$\bigcup_{X \in \mathcal{S}} X = \bigcup \mathcal{S} \qquad \text{often a more-convenient form}$$

Common names for sets

- $\mathbb{N} = \{0, 1, 2, \ldots\}$ the natural numbers (computer scientists and many logicians start at zero – see next slide)
- $\mathbb{Z} = \{0, 1, -1, 2, -2, \ldots\}$ the integers
- $\mathbb{Q}$, the rational numbers (fractions), you might try defining them as $\{(x, y) | x \in \mathbb{Z}, \in \mathbb{N} \setminus \{0\}\}$ but note that $(1, 2)[= \frac{1}{2}] = (2, 4)[= \frac{2}{4}]$
- $\mathbb{R}$, the real numbers
- $\mathbb{Q}$, complex numbers (these just are $\mathbb{R} \times \mathbb{R}$).
- $\mathbb{B} = \{\mathit{true}, \mathit{false}\}$

Making sets by induction

We can also define a set as "the smallest set having a given set of elements".
For example, suppose we say that: "$0 \in S$" and "whenever $x \in S$ then $x + 1 \in S$" then this property is satisfied by $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ and more.
But $\mathbb{N}$ is the smallest such set.
We can define many sets, including $\mathbb{N}$, by "the smallest set such that".
But beware: we need to be careful about what the construction rules say to avoid Russell's paradox.

Sets can define everything

Just like a database is a data-structure built up out of tables built of values build of bits, so everything can be defined as a set.
Here are (common ways) of doing this (note I'm using '$=$' as a definition)

- $(x, y) = \{x, \{x, y\}\}$ (exercise: why does this mean that $(x, y) = (x', y')$ iff $x = x' \wedge y = y'$?)
- $0 = \{\}, 1 = \{0\}, 2 = \{0, 1\}$, etc.
- (informally) a real number, like π can be defined a set of rationals below it and tending to it: $\{3, 3.1, 3.14, 3.14159\}$ and a rational is just a pair of integers ....

Note that even if we do define (say) tuples as a pair of sets, then we try to avoid using the actual definition as much as possible and concentrate on the defined set's *properties*. Think *hidden implementation details* in object-oriented programming or abstract data types.

Sets: number of members

For a finite set (and later for infinite sets) write $|X|$ (or $card(X)$ for the number of elements of X.
Note that we have:

- $|\{\}| = 0$
- $|X + Y| = |X| + |Y|$
- $|X \times Y| = |X| \times |Y|$
- $|X \to Y| = |Y|^{|X|}$
- $\max(|X|, |Y|) \leq |X \cup Y| \leq |X| + |Y|$
- $0 \leq |X \cap Y| \leq \min(|X|, |Y|)$

Defining sizes of infinite sets requires a bit more care ...

Relations, Functions

Write $\mathcal{R}(X, Y)$ for the set of relations between X and Y. [Non-standard, but useful, notation] Its members relate elements of X to those of Y. Note that a relation is just a set of pairs (think city pairs and with airline links). There are operations on relations, e.g. composition but we'll not use them here.

Hence $\mathcal{R}(X, Y) = \mathcal{P}(X \times Y)$. Every possible set of pairs *is* a relation.

A function is just a special case of a relation which has exactly relates every $x \in X$ to exactly one $y \in Y$, so

$$X \to Y = \{f \in \mathcal{R}(X, Y) \mid \forall x \in X \quad \exists! y \in Y \quad (x, y) \in f\}$$

Here I've written $\exists!$ for "there exists a unique". While this is often convenient, it's important to know how to do it properly (next slide). We write $f(x)$ for this unique element.

Functions, bijections

Write $X \to Y$ for the set of functions from X to Y. A relation $f \in \mathcal{R}(X, Y)$ is a function if:

- $(\forall x \in X)(\exists y \in Y) \quad (x, y) \in f$
- $(\forall x \in X)(\forall y, y' \in Y) \quad (x, y) \in f \wedge (x, y') \in f \Rightarrow y = y'$

Note the slight mathematical 'coding' in the last rule expressing uniqueness of the image of f.

- A function f is injective iff $(\forall x, x' \in X) \quad f(x) = f(x') \Rightarrow x = x'$
- A function f is surjective iff $(\forall y \in Y)(\exists x \in X) \quad f(x) = y$.
- A function is bijective if (iff!) it is injective and surjective.

Existence of a bijection in $X \to Y$ means that X and Y have the same number of elements (which we use to define the size of infinite sets).

Countability

A set S is *countable* if there is a bijection $\mathbb{N} \to S$.

There are uncountable sets: one is $\mathbb{R}$.

Cantor's theorem: for every set S there is no bijection between S and $\mathcal{P}(S)$. This means that there are an infinite number of different sizes of infinite sets – as $\mathbb{N}, \mathcal{P}(\mathbb{N}), \mathcal{P}(\mathcal{P}(\mathbb{N})), \mathcal{P}(\mathcal{P}(\mathcal{P}(\mathbb{N}))), \ldots$ are all of different cardinality.

In this course, we just distinguish three sizes of sets: finite sets (including the empty set), countably infinite sets, and uncountable sets. Beware: some books use 'countable' for 'countably infinite' and some for 'finite or countably infinite'.

Beware infinity

Various unexpected things happen for infinite sets (intuitively because we have to describe them with only a finite number of symbols and so there are hence things we cannot say precisely).
For example, for finite sets X, Y of the same cardinality every injection $X \to Y$ is a surjection and vice-versa.
But $f(x) = x + 1$ is an injection $\mathbb{N} \to \mathbb{N}$ which is not a surjection.

Similarly (details off this course) if we have a *finite* set ordered with $<$, then every time I find an finite ascending chain $x_1 < x_2 < \cdots < x_n$ of size n then I can find a descending chain $y_1 > y_2 > \cdots > y_n$ of the same length. But $\mathbb{N}$ has many infinite ascending chains, but no infinite descending chain!

Recursive Enumerability

Just because a set S is countable (there is a bijection $f : \mathbb{N} \to S$) does not necessarily mean that f is calculable by an (idealised) computer.

We say a set is *effectively enumerable* or *recursively enumerable* if the f corresponds to a computable function.

For example, every subset of $\mathbb{N}$ is countable (exercise). It's easy to effectively enumerate the halting Turing machines too. But it's impossible to effectively enumerate the non-halting Turing machines.

Given a subset S of a countable set such as $\mathbb{N}$ it's clear that its complement $\mathbb{N} \setminus S$ is also a subset of $\mathbb{N}$ and hence countable. It's also simple to define a *mathematical function* $\chi_S : \mathbb{N} \to \{0, 1\}$ which has $\chi_S(x) = 1$ iff $x \in S$.

But these are less easy to do by machine ...

Recursive Sets, Kleene's Theorem

A subset S of $\mathbb{N}$ is *recursive* [unusual use of the word at first] if there is a computable function $\mathbb{N} \to \mathbb{B}$ giving membership of S (like χ above).

Theorem (Kleene): a set is recursive iff both it and its complement are recursively enumerable.

The way to think about this is: that an effective enumeration of S is a machine which emits the elements of S in order, and similarly we have a machine which emits the elements of $\mathbb{N} \setminus S$ in order. So to test (effectively) whether x is a member of S we run both machines in parallel, and stop (and say 'yes' or 'no' as appropriate) when the member appears in either output list. We know this always terminates.