Intuitionistic/Constructive Logic

Douglas S. Bridges

Department of Mathematics & Statistics,

University of Canterbury,

Christchurch, New Zealand

Constructive?

Nonconstructive:

$$\neg\forall_{x\in S}\neg P(x) \rightarrow \exists_{x\in S}P(x).$$

Constructive: a proof of $\exists_{x\in S}P(x)$ must embody algorithms for the construction/computation of the desired object x and for showing that $P(x)$ holds.

Three ways to approach computability in mathematics:

1) Use classical computability theory:

The logic allows "decisions" that cannot be made by any real computer, so we need a clearly specified type of algorithm.

Three ways to approach computability in mathematics:

1) Use classical computability theory:

The logic allows "decisions" that cannot be made by any real computer, so we need a clearly specified type of algorithm

2) Use classical *proof mining*:

This requires a heavy logical analysis in order to extract (admittedly often very good) constructive estimates from classical proofs. There may be serious limitations on the kinds of classical theory that can be attacked by this method.

3) Use intuitionistic logic:

This

automatically takes care of the problem of noncomputational "decisions", and

enables us to work, with any mathematical objects, in the familiar style of the analyst, algebraist, geometer, ...

Using intuitionistic logic, we can

— clarify distinctions of meaning obscured by classical logic, and

— allow results to have a wider range of interpretations (including recursive ones) than their counterparts proved with classical logic.

The BHK interpretation

Modern intuitionistic logic is based on the *BHK-interpretation** of the connectives

$$\vee \text{ (or)}, \wedge \text{ (and)}, \rightarrow \text{ (implies)}, \neg \text{ (not)}$$

and quantifiers

$$\exists \text{ (there exists)}, \quad \forall \text{ (for all/each)}.$$

Note that it is *provability*, rather than an *a priori* notion of truth, that is fundamental to the constructive approach.

*Brouwer-Heyting-Kolmogorov

- $P \vee Q$: either we have a proof of P or else we have a proof of Q.

- $P \vee Q$: either we have a proof of P or else we have a proof of Q.

- $P \wedge Q$: we have both a proof of P and a proof of Q.

- $P \vee Q$: either we have a proof of P or else we have a proof of Q.

- $P \wedge Q$: we have both a proof of P and a proof of Q.

- $P \to Q$: by means of an algorithm we can convert any proof of P into a proof of Q.

- $P \vee Q$: either we have a proof of P or else we have a proof of Q.

- $P \wedge Q$: we have both a proof of P and a proof of Q.

- $P \to Q$: by means of an algorithm we can convert any proof of P into a proof of Q.

- $\neg P$: assuming P, we can derive a contradiction (such as $0 = 1$); equivalently, we can prove $(P \to (0 = 1))$.

- $\exists_x P(x)$: we have (i) an algorithm which computes a certain object x, and (ii) an algorithm which, using the information supplied by the application of algorithm (i), demonstrates that $P(x)$ holds.

- $\exists_x P(x)$: we have (i) an algorithm which computes a certain object x, and (ii) an algorithm which, using the information supplied by the application of algorithm (i), demonstrates that $P(x)$ holds.

- $\forall_{x \in A} P(x)$: we have an algorithm which, applied to an object x and a proof that $x \in A$, demonstrates that $P(x)$ holds.

- $\exists_x P(x)$: we have (i) an algorithm which computes a certain object x, and (ii) an algorithm which, using the information supplied by the application of algorithm (i), demonstrates that $P(x)$ holds.

- $\forall_{x \in A} P(x)$: we have an algorithm which, applied to an object x and a proof that $x \in A$, demonstrates that $P(x)$ holds.

Note that in the interpretation of the statement $\forall_{x \in A} P(x)$, the proof of $P(x)$ will normally use both the data describing the object x and the information supplied by a proof that x belongs to the set A.

Under this interpretation, you cannot expect to prove such statements as the following:

LPO For each binary sequence $a \equiv (a_n)_{n \geqslant 1}$ either $a_n = 0$ for all n, or else there exists N such that $a_N = 1$.

LLPO For each binary sequence $a \equiv (a_n)_{n \geqslant 1}$ with at most one term equal to 1, either $a_n = 0$ for all even n, or else $a_n = 0$ for all odd n.

LEM: The law of excluded middle.

Axiomatic logic

Classical logic: primitive connectives $\neg$ and $\rightarrow$; primitive quantifier $\forall$.

Axioms for classical propositional logic:

PC$_1$ $p \rightarrow (q \rightarrow p)$

PC$_2$ $(p \rightarrow (q \rightarrow r)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow r))$

PC$_3$ $(\neg p \rightarrow q) \rightarrow ((\neg p \rightarrow \neg q) \rightarrow p)$

Intuitionistic logic: primitive connectives $\vee$, $\wedge$, $\longrightarrow$, and $\neg$; primitive quantifiers $\exists$ and $\forall$.

Heyting's axioms (1930) for intuitionistic propositional logic:

1. $p \longrightarrow (p \wedge p)$

2. $(p \wedge q) \longrightarrow (q \wedge p)$

3. $(p \longrightarrow q) \longrightarrow (p \wedge r \longrightarrow q \wedge r)$

4. $(p \longrightarrow q) \longrightarrow ((q \longrightarrow r) \longrightarrow (p \longrightarrow r))$

5. $b \leftarrow (d \leftarrow b)$

6. $(a \vee d) \leftarrow ((b \leftarrow d) \leftarrow b)$

7. $d \leftarrow (a \vee b)$

8. $(a \vee d) \leftarrow (b \vee d)$

9. $((a \leftarrow r) \vee (b \leftarrow r)) \leftarrow ((a \vee b) \leftarrow r)$

10. $\neg d \leftarrow (b \leftarrow d)$

11. $(d \leftarrow (b \vee d)) \leftarrow ((b \leftarrow \neg d) \leftarrow \neg d)$

Not surprisingly, Heyting's axioms for intuitionistic *predicate* logic are also more complicated than their classical-logic counterparts, since they take $\exists$ and $\forall$ as independent, primitive quantifiers.

A quote reportedly from Hilbert to Blumenthal on a train journey:

> "One must be able to say at all times—instead of points, straight lines, and planes—tables, chairs, and beer mugs"

Let's put the "beer mugs" bit into practice—now!