- 5.4

Russell's Paradox and the Halting Problem

P. Danziger

1 Russell's Paradox

With the advent of axiomatic theory it seemed reasonable that it should be possible to write a book, which started with some basic axioms, and then derived all of mathematics from these axioms. At the beginning of this century Alfred Whitehead (1861 - 1947) and Betrand Russell (1872 - 1970) attempted to write just such a book, *Principia Mathematica,* which actually appeared in three volumes from 1910 to 1913. The scope of this work was incredible, the proof that $1 + 1 = 2$ does not appear until page 362.
However Russell and Whitehead ran into a problem, they kept running into inconsistencies. Every time they tried to fix an inconsistency it would pop up again elsewhere. Though they did eventually publish, the work was flawed in that it was incomplete, there were mathematical theorems it could not address from the fundamental axioms.
The basic inconsistency that they found is known as Russell's paradox. Russell provided the following simple puzzle, known as the barbers paradox, to exemplify the problem:

> In a certain town there is a male barber who shaves all those men, and only those men, who do not shave themselves.
> The question is, who shaves the barber?

If the barber does not shave himself, then he should, since he shaves all those men who do not shave themselves. On the other hand the barber *only* shaves such men, and hence cannot shave himself.
Sets can have other sets as members, e.g. $\{1, \{2, 3\}\}$.
What about a set that conatains itself as a member.
As an example consider the universal set of everything.
Since this set conatains everything it must contain itself.
While the property of self containment is unusual it is not, in and of itself, paradoxical.
We obtain a paradox when we consider 'the set of all sets which are not members of themselves'.

$$R = \{ \text{ Sets } A \,|\, A \notin A\}$$

The question is, is R a member of itself?
R cannot not be a member of itself, but it must be. This is known as Russell's paradox.
If we go back to the definition of set given earlier, we can actually find a way out of this dilemma.

> A *set* is a collection of objects, such that it is possible to state unequivocally whether any given object is in the set or not.

Since we have an object, R, for which it is not possible to state unequivocally $R \in R$ or $R \notin R$, we must conclude that, by this definition, R is not a set.

2 Gödel's Theorem

The final word comes from the Austrian mathematician Kurt Gödel (1906 - 1978).
If it is possible to prove two mutually contradictory statements from a set of axioms, then this set of axioms is called *inconsistent,*
If there exists a theorem which **cannot** be proved or disproved from a set of axioms, then this set of axioms is called *incomplete.*
Gödel's theorem (1931) may be succinctly stated as follows.

> Every formal axiomatic system, with a finite number of axioms, is either incomplete or inconsistent.

What is truly incredible about this is that Gödel managed to use a formal mathematical approach to show that mathematics can never be both complete and consistent.
See *Gödel, Escher, Bach* by D. Hofstadter for a more complete discussion of Gödel's theorem.
Of course in mathematics we eshew inconsistency and always go for incompleteness.

3 The Halting Problem

We wish to design an algorithm, H which will take as input an algorithm, A and a potential input to that algorithm w and decide whether A eventually halts on the input w.
The question of whether such an algorithm H exists or not is known as the halting problem. The existence of such an algorithm would be very useful for checking whether our programs ever enter an infinite loop. In fact one of the ramifications of Russell's paradox is that no such algorithm can exist.
It first is necessary to realise that an algorithm is, ultimately, a sequence of characters, presumably in some appropriate programming language. This sequence can be used as an input to another algorithm. Indeed by asking the halting problem we implicitly assumed that the algorithm A could be an input for H. This is reminiscent of sets within sets.
In particular if A expects an algorithm for input, w could be the ecoding of another algorithm.

Theorem 1 *There is no Algorithm which will take as input an algorithm, A and a potential input to that algorithm w and decide whether A eventually halts on the input w.*

Proof:(By Contradiction)
Suppose that such an algorithm exists, $H(A, w)$.
So $H(M, w)$ outputs $\begin{cases} \textit{Halts} & \text{if } M \text{ Halts on input } w \\ \textit{Loops} & \text{if } M \text{ does not halt on input } w \end{cases}$
Now we design a new algorithm D, which uses H as a 'subroutine'. The input to D is the encoding of an algorithm M. On input M, D runs H on $< M, M >$. i.e. H determines the outcome of running the algorithm M on an encoding of itself. D then reverses the output from H and acts as follows
$D(M)$ $\begin{cases} \text{Loops forever} & \text{if } M \text{ Halts on input } M \\ \text{Halts} & \text{if } M \text{ does not halt on input } < M > \end{cases}$
We now run D on itself to derive a contradiction:

$$D(D) \begin{cases} \text{Loops forever} & \text{if } D \text{ Halts on input } D \\ \text{Halts} & \text{if } D \text{ does not halt on input } D \end{cases}$$

But this says that D halts, then it loops forever and that if it halts that it loops forever. We have derived a contradiction, so our original assumption, that H exists is wrong. $\square$

Has it crashed, or is it just taking a long time?