The halting problem in general is the problem of deciding, given the specification \( e \) (e.g., program) of a computable function and a number \( n \), whether the computation of the function on input \( n \) halts, i.e., produces a result. Famously, Alan Turing proved that this problem itself cannot be solved by a computable function, i.e., the function

\[
h(e, n) = \begin{cases} 
1 & \text{if computation } e \text{ halts on input } n \\
0 & \text{otherwise},
\end{cases}
\]

is not computable.

In the context of partial recursive functions, the role of the specification of a program may be played by the index \( e \) given in Kleene’s normal form theorem. If \( f \) is a partial recursive function, any \( e \) for which the equation in the normal form theorem holds, is an index of \( f \). Given a number \( e \), the normal form theorem states that

\[
\varphi_e(x) \simeq U(\mu s \, T(e, x, s))
\]

is partial recursive, and for every partial recursive \( f : \mathbb{N} \rightarrow \mathbb{N} \), there is an \( e \in \mathbb{N} \) such that \( \varphi_e(x) \simeq f(x) \) for all \( x \in \mathbb{N} \). In fact, for each such \( f \) there is not just one, but infinitely many such \( e \). The halting function \( h \) is defined by

\[
h(e, x) = \begin{cases} 
1 & \text{if } \varphi_e(x) \downarrow \\
0 & \text{otherwise.}
\end{cases}
\]

Note that \( h(e, x) = 0 \) if \( \varphi_e(x) \uparrow \), but also when \( e \) is not the index of a partial recursive function at all.

**Theorem rec.1.** The halting function \( h \) is not partial recursive.

**Proof.** If \( h \) were partial recursive, we could define

\[
d(y) = \begin{cases} 
1 & \text{if } h(y, y) = 0 \\
\mu x \, x \neq x & \text{otherwise.}
\end{cases}
\]

From this definition it follows that

1. \( d(y) \downarrow \) iff \( \varphi_y(y) \uparrow \) or \( y \) is not the index of a partial recursive function.

2. \( d(y) \uparrow \) iff \( \varphi_y(y) \downarrow \).

If \( h \) were partial recursive, then \( d \) would be partial recursive as well. Thus, by the Kleene normal form theorem, it has an index \( e_d \). Consider the value of \( h(e_d, e_d) \). There are two possible cases, 0 and 1.

1. If \( h(e_d, e_d) = 1 \) then \( \varphi_{e_d}(e_d) \downarrow \). But \( \varphi_{e_d} \simeq d \), and \( d(e_d) \) is defined iff \( h(e_d, e_d) = 0 \). So \( h(e_d, e_d) \neq 1 \).
2. If \( h(e_d, e_d) = 0 \) then either \( e_d \) is not the index of a partial recursive function, or it is and \( \varphi_{e_d}(e_d) \uparrow \). But again, \( \varphi_{e_d} \simeq d \), and \( d(e_d) \) is undefined iff \( \varphi_{e_d}(e_d) \downarrow \).

The upshot is that \( e_d \) cannot, after all, be the index of a partial recursive function. But if \( h \) were partial recursive, \( d \) would be too, and so our definition of \( e_d \) as an index of it would be admissible. We must conclude that \( h \) cannot be partial recursive. 

\[ \square \]

Photo Credits

Bibliography