Self-referential algorithms - New proof that \( \text{Halt}_0 \) is not decidable

Recall \( \text{Halt}_0 = \{ \langle M, w \rangle : M(w) \text{ halts} \} \)

Former proof that \( \text{Halt}_0 \) not decidable:

- Started with \( M \) assumed to decide \( \text{Halt}_0 \)
- Formed \( M' \) s.t. \( M'(\langle M' \rangle) \) acts like \( M(\langle M \rangle) \)
- Gave a many-one reduction from \( \text{Halt}_0 \) to \( \text{Halt}_0 \)

Now (Self-Reference)

We’ll form \( D_m \) such that

\( D_m(\langle D_m \rangle) \) acts the same as \( M(\langle D_m \rangle) \)

Encourage google “self-printing programs” or “quines”
Let $M$ be an algorithm that takes one input.

There is an algorithm $D_M$ such that

$D_M$ acts the same as $M(\tilde{D}_M^2)$

Proof: Recall $f', f$

$$f'(f(M_1^2), M_2^2) = \text{Gödel number of algorithm that runs } M_2(\tilde{M}_1^2).$$

$$<M_1^2, M_2^2> \mapsto f'(f(M_1^2), M_2^2)$$

Computable

$$f(M_1^2) = \text{G.n. of an algorithm that outputs } M_1^2.$$

$$f(w) = \text{G.n. of an algorithm for constant function } w$$

$$f'(M_3^2, \tilde{M}_2^2) = \text{G.n. of an algorithm that, on input } w, \text{ runs } M_3(w)$$

(2) Runs $M_2$ on the output of $M_3$. 

"Diagonal"
Proof of Theorem

Form algorithm \( E_m \) s.t.

\[
E_m(w) \text{ runs } M(f'(f(w), w))
\]

\[
E_m(\#M) \text{ runs } M(f'(f(\#M), \#M))
\]

G.n. of algorithm that runs \( N \) on \( \#M \).

Define \( D_M \) to the algorithm with Gödel number

\[
\#D_M = f'(f(\#E_m), \#E_m)
\]

So \( D_M \) ignores its input and runs

\( E_m \) on \( \#E_m \)

i.e. runs \( M \) on \( f'(f(\#E_m), \#E_m) \)

i.e. runs \( M \) on \( \#D_M \).

\[
f(11) = \begin{array}{c}
85 \quad \#M, 1, L
\end{array} \rightarrow
\begin{array}{c}
81 \quad \#A, 1, L
\end{array} \rightarrow
\begin{array}{c}
82 \quad \#B, 2
\end{array} \rightarrow
\begin{array}{c}
0
\end{array}
\]

\[
f'(f(\#E_m), \#E_m) = \text{G.n. of algorithm that runs } E_m \text{ on } \#E_m.
\]
\[ f(w) = \max \{ \text{printf} \left( \text{"w \\
 \n"} \right); \} \]
Theorem Halto is undecidable

Pf

Assume N decides Halto

Define algorithm M by

Algorithm M

Input w

Runs N on w

If N(w) rejects, then halt.
If N(w) accepts, then enter an
infinite loop and never halt.

Form DM

DM(c) halts iff M(DM^c) halts iff N(DM^c) rejects

iff DM does not halt.
(since N decides Halto)

Contradiction / QED.
Rice's Theorem

"Cannot decide what a program does by looking at its source code."

Definition: A non-trivial property of c.e. sets is a collection \( P \) of c.e. sets such that \( P \neq \emptyset \) and does not contain all c.e. sets.

Then let \( P \) be as above. Then

\[
X := \{ M \mid L^e(M) \in P \}
\]

is undecidable. Where \( L^e(M) = \text{set enumerated by } M \).

Example: \( P \) is the set of c.e. sets \( R \) s.t.

1. \( R \) is non-empty
2. \( E \) is in \( R \)
3. Every member of \( R \) is even.
Proof
Suppose \( X \) is decidable by \( M \)
Let \( M_1, M_2 \) be such \( L^e(M_1) \notin P \) and \( L^e(M_2) \notin P \).
There exist since \( P \) is nontrivial.

Algorithm \( N \)

\underline{Input} \( w \)

\underline{Run} \( M(w) \)

\underline{If} \( M(w) \) accepts,

\underline{Run} \( M_2 \) - use its outputs

\underline{If} \( M(w) \) rejects,

\underline{Run} \( M_1 \) - use its outputs.

Form \( DN \) (Diagonal Theorem)

\( DN \) enumerates \( L^e(M_2) \) if \( M(\overline{DN}) \) accepts
So \( L^e(DN) \notin P \) if \( M(\overline{DN}) \) accepts

\( DN \) enumerates \( L^e(M_1) \) if \( M(\overline{DN}) \) rejects
\( L^e(DN) \notin P \) if \( M(\overline{DN}) \) rejects

Contradiction by choice of \( M \).
Contradiction since $M$ decides $X$. 
So $M(N) \text{ accepts }$ if $L_e(N_2) \in P$. 
So $M(D_N) \text{ accepts }$ if $L_e(D_N) \in P$.

#

Can use a shorthand notation $M(N)$ to denote $M(N)$.

In PDF - Read about computably inseparable c.e. sets.

Independent reading