Non-Existence of Entities (Sci.American 1980s)

There are objects/entities which one can describe but which can't exist (maybe because their description is "faulty"), one example:

Describe really large numbers, using n symbols, e.g. $n = 3$. Maybe this could be 999, better 9^{9^9}, or (hexadecimal) F^{F^F}, ...

LARGEST $n \in \mathbf{N}$ DESCRIBED BY AT MOST 43 SYMBOLS

$7 + 3 + 9 + 2 + 2 + 4 + 2 + 7 = 36$ + 7 spaces $\Rightarrow$ 43 symbols

Thus, we can't have the largest number described with 45 symbols:

LARGEST $n \in \mathbf{N}$ DESCRIBED BY AT MOST 45 SYMBOL S+1

Halting Problem for Register Machines

Definition. A register machine H **decides the Halting Problem** if for all $e, a_1, \ldots, a_n \in \mathbb{N}$, starting H with

$$R_0 = 0 \qquad R_1 = e \qquad R_2 = \ulcorner [a_1, \ldots, a_n] \urcorner$$

and all other registers zeroed, the computation of H always halts with R_0 containing 0 or 1; moreover when the computation halts, $R_0 = 1$ if and only if

> the register machine program with index e eventually halts when started with $R_0 = 0, R_1 = a_1, \ldots, R_n = a_n$ and all other registers zeroed.

Halting Problem for Register Machines

Definition. A register machine H **decides the Halting Problem** if for all $e, a_1, \ldots, a_n \in \mathbb{N}$, starting H with

$$R_0 = 0 \qquad R_1 = e \qquad R_2 = \ulcorner [a_1, \ldots, a_n] \urcorner$$

and all other registers zeroed, the computation of H always halts with R_0 containing 0 or 1 ; moreover when the computation halts, $R_0 = 1$ if and only if

> the register machine program with index e eventually halts when started with $R_0 = 0, R_1 = a_1, \ldots, R_n = a_n$ and all other registers zeroed.

Theorem No such register machine H can exist.

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

- Let H' be obtained from H by replacing $START \rightarrow$ by

  $$START \rightarrow \boxed{Z ::= R_1} \rightarrow \boxed{\begin{matrix} push\ Z \\ to\ R_2 \end{matrix}} \rightarrow$$

  (where Z is a register not mentioned in H's program).

- Let C be obtained from H' by replacing each $HALT$ (& each erroneous halt) by

  $$\begin{tikzcd} {} \arrow[r] & R_0^- \arrow[r, bend left] \arrow[d] & R_0^+ \arrow[l, bend left] \\ & HALT & \end{tikzcd}$$
  .

- Let $c \in \mathbb{N}$ be the index of C's program.

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows: (assuming $R_0 = 0$ and $R_2 = 0$)

C started with $R_1 = c$ eventually halts

if and only if

H' started with $R_1 = c$ halts with $R_0 = 0$

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $R_1 = c$ eventually halts

if and only if

H' started with $R_1 = c$ halts with $R_0 = 0$

if and only if

H started with $R_1 = c, R_2 = \ulcorner [c] \urcorner$ halts with $R_0 = 0$

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $R_1 = c$ eventually halts

if and only if

H' started with $R_1 = c$ halts with $R_0 = 0$

if and only if

H started with $R_1 = c, R_2 = \ulcorner [c] \urcorner$ halts with $R_0 = 0$

if and only if

$prog(c)$ started with $R_1 = c$ does not halt

$prog(c)$ means the program given by the number c.

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $R_1 = c$ eventually halts

if and only if

H' started with $R_1 = c$ halts with $R_0 = 0$

if and only if

H started with $R_1 = c, R_2 = \ulcorner [c] \urcorner$ halts with $R_0 = 0$

if and only if

$prog(c)$ started with $R_1 = c$ does not halt

if and only if

C started with $R_1 = c$ does not halt

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $R_1 = c$ eventually halts

if and only if

H' started with $R_1 = c$ halts with $R_0 = 0$

if and only if

H started with $R_1 = c, R_2 = \ulcorner [c] \urcorner$ halts with $R_0 = 0$

if and only if

$prog(c)$ started with $R_1 = c$ does not halt

if and only if

C started with $R_1 = c$ does not halt

Contradiction!

Enumerating computable functions

For each $e \in \mathbb{N}$, let $\varphi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the unary partial function computed by the RM with program $prog(e)$. So for all $x, y \in \mathbb{N}$:

$\varphi_e(x) = y$ holds iff the computation of $prog(e)$ started with $R_0 = 0, R_1 = x$ and all other registers zeroed eventually halts with $R_0 = y$.

Thus

$$e \mapsto \varphi_e$$

defines an **onto** function from $\mathbb{N}$ to the collection of all computable partial functions from $\mathbb{N}$ to $\mathbb{N}$.

An uncomputable function

Let $f \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the partial function $\{(x, 0) \mid \varphi_x(x)\uparrow\}$.

Thus $f(x) \; = \; \begin{cases} 0 & \text{if } \varphi_x(x)\uparrow \\ \mathit{undefined} & \text{if } \varphi_x(x)\downarrow \end{cases}$

An uncomputable function

Let $f \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the partial function $\{(x, 0) \mid \varphi_x(x)\uparrow\}$.

Thus $f(x) = \begin{cases} 0 & \text{if } \varphi_x(x)\uparrow \\ \mathit{undefined} & \text{if } \varphi_x(x)\downarrow \end{cases}$

f is not computable, because if it were, then $f = \varphi_e$ for some $e \in \mathbb{N}$ and hence

- if $\varphi_e(e)\uparrow$, then $f(e) = 0$ (by def. of f); so $\varphi_e(e) = 0$ (by def. of e), i.e. $\varphi_e(e)\downarrow$
- if $\varphi_e(e)\downarrow$, then $f(e)\uparrow$ (by def. of e); so $\varphi_e(e)\uparrow$ (by def. of f)

Contradiction! So f cannot be computable.

(Un)decidable sets of numbers

Given a subset $S \subseteq \mathbb{N}$, its **characteristic function** $\chi_S \in \mathbb{N}{\to}\mathbb{N}$ is given by: $\chi_S(x) \triangleq \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{if } x \notin S. \end{cases}$

(Un)decidable sets of numbers

Definition. $S \subseteq \mathbb{N}$ is called (register machine) **decidable** if its characteristic function $\chi_S \in \mathbb{N} \rightarrow \mathbb{N}$ is a register machine computable function. Otherwise it is called **undecidable**.

So S is decidable iff there is a RM M with the property: for all $x \in \mathbb{N}$, M started with $R_0 = 0$, $R_1 = x$ and all other registers zeroed eventually halts with R_0 containing 1 or 0; and $R_0 = 1$ on halting iff $x \in S$.

(Un)decidable sets of numbers

Definition. $S \subseteq \mathbb{N}$ is called (register machine) **decidable** if its characteristic function $\chi_S \in \mathbb{N}\to\mathbb{N}$ is a register machine computable function. Otherwise it is called **undecidable**.

So S is decidable iff there is a RM M with the property: for all $x \in \mathbb{N}$, M started with $R_0 = 0, R_1 = x$ and all other registers zeroed eventually halts with R_0 containing 1 or 0; and $R_0 = 1$ on halting iff $x \in S$.

Basic strategy: to prove $S \subseteq \mathbb{N}$ undecidable, try to show that decidability of S would imply decidability of the Halting Problem.

For example...

Claim: $S_0 \triangleq \{e \mid \varphi_e(0)\downarrow\}$ **is undecidable.**

Claim: $S_0 \triangleq \{e \mid \varphi_e(0){\downarrow}\}$ **is undecidable.**

Proof (sketch): Suppose M_0 is a RM computing χ_{S_0}. From M_0's program (using the same techniques as for constructing a universal RM) we can construct a RM H to carry out:

$$let\ e = R_1\ and\ \ulcorner[a_1,\ldots,a_n]\urcorner = R_2\ in$$
$$R_1 ::= \ulcorner(R_1 ::= a_1)\ ; \cdots ;\ (R_n ::= a_n)\ ;\ prog(e)\urcorner\ ;$$
$$R_2 ::= 0\ ;$$
$$run\ M_0$$

Then by assumption on M_0, H decides the Halting Problem. **Contradiction.** So no such M_0 exists, i.e. χ_{S_0} is uncomputable, i.e. S_0 is undecidable.

Claim: $S_1 \triangleq \{e \mid \varphi_e \; total \; function\}$ **is undecidable.**

Claim: $S_1 \triangleq \{e \mid \varphi_e \; total\ function\}$ is undecidable.

Proof (sketch): Suppose M_1 is a RM computing χ_{S_1}. From M_1's program we can construct a RM M_0 to carry out:

$$\begin{array}{l} let\ e = R_1\ in\ R_1 ::= \ulcorner R_1 ::= 0\,;\,prog(e)\urcorner\,; \\ run\ M_1 \end{array}$$

Then by assumption on M_1, M_0 decides membership of S_0 from previous example (i.e. computes χ_{S_0}). **Contradiction.** So no such M_1 exists, i.e. χ_{S_1} is uncomputable, i.e. S_1 is undecidable.