$[n_1, n_2, \ldots, n_k]$ denotes the tape configuration where the tape contains a block of n_1+1 1's, followed by a 0, followed by n_2+1 1's, followed by a 0, … followed by n_k+1 1's, on an otherwise blank (all 0) tape, and with the head at the leftmost 1

A function $f:N^k \rightarrow N$ is Turing-computable* if and only if there exists a Turing machine M such that for all $<n_1, n_2, \ldots, n_k>$: if $f(n_1, n_2, \ldots, n_k) = n$, then M, when started (in q_1) on $[n_1, n_2, \ldots, n_k]$, will halt (in q_0) with $[n]$.

Relative to an enumeration $E = M_1, M_2, \ldots$ of Turing-machines, the self-halting function $h_E:N \rightarrow N$ is the function defined as

$h_E(n) = 1$ if M_n, when started on $[n]$, halts

$h_E(n) = 0$ if M_n, when started on $[n]$, does not halt

Relative to an enumeration $E = M_1, M_2, \ldots$ of Turing-machines, the halting function $h_E:N \times N \rightarrow N$ is the function defined as

$h_E(m,n) = 1$ if M_m, when started on $[n]$, halts

$h_E(m,n) = 0$ if M_m, when started on $[n]$, does not halt

TM is the set of all Turing-machines

TT is the set of all tape-configurations (specifying tape content; head position is assumed to be at leftmost 1)

The halting function $h:TM \times TT \rightarrow \{yes, no\}$ is the function defined as

$h(M,T) = yes$ if M, when started on T, halts

$h(M,T) = no$ if M, when started on T, does not halt

Given some computational method, an effective encoding maps any $x \in X$ to a representation $[x]$ that that kind of computational method can work on.

The halting function $h:TM \times TT \rightarrow \{yes, no\}$ is MM-computable if and only if there exists an effective symbol-manipulating algorithm (i.e. computation, program, or machine) H of kind MM, such that for any M and T: H starts on [M,T], and halts on [yes] or [no], depending on whether M with T halts or does not halt respectively.

‘Effective’ = can be performed by a human

The Halting Problem: Given any Turing-machine M and tape-configuration T, can we decide whether M halts on T or not?

For any enumeration $E = M_1, M_2, \ldots$ of Turing-machines, the self-halting function $h_E:N \rightarrow N$ is not Turing-computable*

↓

For any enumeration $E = M_1, M_2, \ldots$ of Turing-machines, the halting function $h_E:N \times N \rightarrow N$ is not Turing-computable*
+ Every effective encoding of numbers can be transformed into any other effective encoding of numbers by some Turing-machine

↓

For any enumeration $E = M_1, M_2, \ldots$ of Turing-machines, the halting function $h_E:N \times N \rightarrow N$ is not Turing-computable
+ Every effective encoding [M,T] where T = [n] can be transformed into [m,n] relative to any enumeration E = M1, M2, … of Turing-machines

↓

The halting function $h:TM \times TT \rightarrow \{yes, no\}$ is not Turing-computable
+ Turing's Thesis: every effectively computable function is Turing-computable

↓

The halting function $h:TM \times TT \rightarrow \{yes, no\}$ is not effectively computable
+ There are no such things as ‘non-effective computations’ (e.g. accelerated Turing-machines)

↓

The halting function is not computable
+ Other methods to try and solve the halting problem (e.g. divine revelation) don't exist

↓

The Halting Problem is not solvable