An interesting fixed-point condition

The first step is to construct a pair of Godel numberings in which natural numbers represent possible Turing machines, and sentences in the langauge, respectively.

If n is a natural number, we denote by $\sigma(n)$ a sentence in the language which encodes n in the usual way. The length of the sentence with Godel number n is denoted $l(n)$, and the length of the sentence $\sigma(n)$ will not be notated but we will choose σ so that this length is logarithmic in n (for example using binary expansions).

We introduce a notion of proofs in the language, and we will ensure that that propositions which have a proof in the language also are true in the real world.

We introduce a predicate P, which we intuitively interpret as meaning that the sentence with Godel number a is a proof of the sentence with Godel number b; and we introduce rules of deduction, so that for any two actual natural numbers a, b, $P(\sigma(a), \sigma(b))$ logically implies, according to a combination of these rules, that the sentence with Godel number a is a proof of the sentence with Godel number b, and which uses only these same rules of deduction. In accordance with the previous paragraph, these rules of deduction are restricted to be among those which we believe to be logically valid in the real world.

We also include the axiom "For all a, b $P(a, b)$ implies b."

We introduce a function symbol T (the letter T stands for 'truncation') and we intuitively interpret $T(m, i, x)$ as the Godel number of the output string when Turing machine with Godel number m is given input string with Godel number x and performs i steps. (Here 'input string' is terminology about computer programs instead of Turing machines, this is inessential).

When the Godel number of the ouptut string in this situation is t, we arrange to include axioms about the action of Turing machines such that that $T(\sigma(m), \sigma(i)) = \sigma(t)$ is provable, and we alter the rules of deduction so that again $P(\sigma(a), \sigma(b))$ logically implies that sentence numbered a is a proof of sentence numbered b even if these sentences include the symbol T. One can check that adjusting the rules of deduction here only needs to be done once, this is a well-founded process.

We introduce the predicate symbol Q and an axiom saying that for all m, n, x $Q(m, n, x)$ is logically equivalent to $P(T(m, n, x), x)$. Intuitively this means that the n'th truncation of the m'th Turing machine when given input x outputs a proof of x. Again we go back and modify P and the axiomatization as needed. This is well-founded as the axiom we've just introduced is unchanged. Here m, n, x are variables.

Definition. We say that the *polynomial fixed-point condition* is true if there is a two-variable function s such that for every natural number m, firstly $l(s(m, n))$ is bounded above by a polynomial in the length of $\sigma(n)$, and secondly the sentence "X if and only if not $Q(\sigma(m), \sigma(n), \sigma(s(m, n)))$ " has a proof, in the language, whose length is also bounded above by a polynomial in the length of $\sigma(n)$, where X is the sentence with Godel number $s(m, n)$.

If the fixed point condition is true, we can for any pair of natural numbers n and m, construct a formal proof, in the language, of the the sentence with Godel number $s(m, n)$. The proof can start with the assumption that the sentence with Godel number $s(m, n)$ is false. Then $Q(\sigma(m), \sigma(n), \sigma(s(m, n)))$ is true, then use the axiom providing the logical equivalence with $P(T(\sigma(m), \sigma(n), \sigma(s(m, n))), \sigma(s(m, n)))$. Then finally using quantifier elimination and the the axiom "For all a, b, $P(a, b)$ implies b" to deduce the truth of the sentence itself with Godel number $s(m, n)$.

For each pair of natural numbers m, n, let $f(m, n)$ be the Godel number of the sentence which is the proof described in the previous paragraph.

We will say that a Turing machine with Godel number a is a 'degree c proof finder' if for all a, b and for all $i > (l(a)+l(b))^c$ we have $P(a, b)$ implies $P(T(a, i, b), b)$. This is not implication in the language, we just mean that whenever $P(a, b)$ is true also $P(T(a, i, b), b)$ must be true.

Theorem. If the polynomial fixed-point condition is true, then for all c there is no degree c proof finder.

Proof. For each number n, $f(a, n)$ is the Godel number of a sentence which is a proof of the sentence with Godel number $s(a, n)$. Therefore, for every number n, $P(f(a, n), s(a, n))$ is true. Because of our hypothesis about a we know that $P(T(a, i, s(a, i), s(a, n))$ is true for $i > (l(f(a, n)) + l(s(a, n))^c$. This is equivalent to $Q(a, i, s(a, n))$. By one of the requirements of the polynomial fixed-point condition, the sentence with Godel number $f(a, n)$ has length bounded above by a polynomial in $\sigma(n)$. Also that $l(s(a, n))$ is bounded above by a polynomial in the length of $\sigma(n)$. Then the sum has logarithmic growth, and for sufficiently large n $n \geq (l(f(a, n))+l(s(a, n)))^c$. Choose such an n. Then $Q(a, n, s(a, n))$ is true. This is provably equivalent to the negation of the sentence with Godel number $s(a, n)$ Then the sentence with Godel number $f(a, n)$ is a proof in the language of a sentence which is false in real life. But we were careful to arrange that proofs in the language only use deductions that are valid in real life.

Remark. A weak form of the polynomial fixed-point condition would be to state the condition without the polynomial bound. This would be similar to Godel's first proof. Once m, n are fixed, denoting by $g(F)$ the Godel number of each formula F with one free variable, we define H so that whenever F is a formula with one free variable, $H(g(F))$ = "Not $P(T(m, n, F(g(F)), F(g(F)))$." Then $H(g(H))$ = "Not $P(T(m, n, H(g(H))), H(g(H))$"