NOWHERE TO HIDE

— NOTES FOR ADVANCED COMPUTATIONAL MODELS —

We will say that a numerical function $\widehat{f} : \mathbb{N} \to \mathbb{N}$ *simulates* a function $f : D \to D$ over an arbitrary domain D with respect to an injective representation $\rho : D \twoheadrightarrow \mathbb{N}$ if $\rho(f(n)) = \widehat{f}(\rho(n))$ for all $n \in \mathbb{N}$. (This definition can be extended to partial functions and to wider arity than 1.)

The question we address is whether, with this definition, an (intuitively) uncomputable function can be simulated by a recursive (computable) function if do not restrict the representation to be (intuitively) effective. We show that—under reasonable assumptions—such an anomaly is impossible, lending credence to the above definition of simulation.

We need to be careful, since a computable function can in fact simulate an uncomputable one if the representation itself does uncomputable work. For example, consider any standard enumeration TM_n, $n = 0, 1, \ldots$, of Turing machines. Define the following uncomputable functions: $\bar{h} : \mathbb{N} \to \mathbb{N}$ enumerates (the numbers of) those machines that do not halt on the empty tape; $h : \mathbb{N} \to \mathbb{N}$ enumerates those that do. So $h(\mathbb{N}) \uplus \bar{h}(\mathbb{N}) = \mathbb{N}$, where $h(\mathbb{N})$ is the image $\{h(n) : n \in \mathbb{N}\}$ of h and $\bar{h}(\mathbb{N})$ is the image of $\bar{h}$. Then the uncomputable function

$$H(n) \quad := \quad \begin{cases} \min \bar{h}(\mathbb{N}) & \text{if } \mathrm{TM}_n \text{ does not halt} \\ \min h(\mathbb{N}) & \text{if } \mathrm{TM}_n \text{ halts} \end{cases}$$

is simulated by the computable parity function ($n \bmod 2$) under the following (uncomputable) *bijective* representation:

$$\rho(n) \quad := \quad \begin{cases} 2\bar{h}^{-1}(n) & \text{if } \mathrm{TM}_n \text{ does not halt} \\ 2h^{-1}(n) + 1 & \text{if } \mathrm{TM}_n \text{ halts} \end{cases}$$

Note that the successor function cannot be simulated by any computable function under this nefarious representation. (Exercise: Prove or disprove this claim.)

In the opposite direction, it is quite clear that a non-computable function can always simulate a recursive one. For example, the identity function $\imath$ on the naturals is tracked by the uncomputable function

$$\begin{aligned} \widehat{\imath}(2n) &:= n \\ \widehat{\imath}(2n+1) &:= \mathit{halt}(n) \end{aligned}$$

under the (computable) injective representation $\rho : n \twoheadrightarrow 2n$ (where *halt* is the halting function).

To circumvent the hiding of supernatural power in a representation, without resorting to a circular definition of effectiveness, let's say that an injective numerical representation $\rho : D \twoheadrightarrow \mathbb{N}$ is *good* if there is some finite (or countable) set of functions C over D via which all of D is reachable, that is, each $x \in D$ is the value of at least one (ground) term over (the signature of) C. Furthermore, each $c \in C$ is simulated under ρ by a *recursive* numerical functions $\widehat{c}$. This condition guarantees that representations for all the elements of the domain can be effectively generated.

The key to everything is that a good representation always has a recursive definition (as we saw above for successor):

$$\rho(c(x_1, \ldots, x_\ell)) \quad = \quad \widehat{c}(\rho(x_1), \ldots, \rho(x_\ell))$$

for each $c \in C$ (at least one of which must be a constant of arity 0). For this to constitute an effective definition, we need to be able to extract the arguments of $c(x_1, \ldots, x_\ell)$ given its value in D. Assuming the ability to check equality over D, one can effectively generate all terms over C in some order, remembering how they have been constructed, so that when the domain value $c(x_1, \ldots, x_\ell)$ is encountered, one knows which $c \in C$ was used to construct it and also what the values of its arguments were. Furthermore, the inverse function $\rho^{-1} : \mathbb{N} \to D$ can be computed for every number in the image $\rho(\mathbb{N})$ of the representation, by enumerating the elements of D and looking at their values under ρ.

For example, consider the constructors zero, 0, and successor, s, of the naturals. Suppose these are mapped to $\widehat{0}$ and $\widehat{s}$, respectively, under a good injective representation $\rho : \mathbb{N} \twoheadrightarrow \mathbb{N}$. Then ρ can be defined recursively as follows:

$$\begin{aligned} \rho(0) &= \widehat{0} \\ \rho(s(n)) &= \widehat{s}(\rho(n)) \end{aligned}$$

Therefore, any $g : \mathbb{N} \to \mathbb{N}$ that is simulated by a recursive $\widehat{g} : \mathbb{N} \to \mathbb{N}$ under this representation must itself be recursive, since

$$g(n) \quad = \quad \rho^{-1}(\widehat{g}(\rho(n)))$$

is the composition of computable functions.

The point is that in general, under the above (lax) assumptions, if the simulating function $\widehat{h}$ is computable, then the simulated function h can in fact be programmed effectively over the combined domain $D \uplus \mathbb{N}$:

$$h(c(x_1, \ldots, x_\ell)) \quad = \quad \rho^{-1}(\widehat{c}(\rho(x_1), \ldots, \rho(x_\ell)))$$

for each $c \in C$. For this to work, we presume an effective equality test for D. This makes it possible to extract the x_i, to which the computable ρ, ρ^{-1}, and $\widehat{c}$'s can be applied.

It seems right, then, to say, in general, that a function h over an arbitrary domain D is *not* computable iff there is a good bijective representation $\pi : D \leftrightarrow \mathbb{N}$ under which h is simulated by an *uncomputable* numeric function. We show that

(1) no uncomputable function (in this sense) over any domain is ever simulated by a recursive function for any good injective representation and that (2) every computable function can be simulated by a recursive function regardless of which good representation is used.

(1) Let $h' : \mathbb{N} \to \mathbb{N}$ be the function that simulates h under π, that is, $\pi(h(x)) = h'(\pi(x))$ for all $x \in D$. Similarly, let $c' : \mathbb{N} \to \mathbb{N}$ be the recursive simulation of $c \in C$.

If $\widehat{h}$, the function that simulates h under ρ, is recursive, then h' must also be, since

$$h'(n) = \pi(h(\pi^{-1}(n))) = \pi(\rho^{-1}(\widehat{h}(\rho(\pi^{-1}(n)))))$$

Let $\tau = \pi^{-1} \circ \rho$. We will show that τ is good, from which it follows, by the previous discussion, that $h' = \tau \circ \widehat{h} \circ \tau^{-1}$, which is simulated by $\widehat{h}$, is recursive. Hence that h is effective by our proposed definition.

Define (looking at the unary case by way of example)

$$\widehat{c}_k(n) := \tau(c'_k(\tau^{-1}(n)))$$

To show that τ is effective, we can work with any standard numerical encoding of terms over C, rather than with elements of D, and compute π^{-1} and ρ as indicated above.

(2) Let $g : D \to D$; let g be mapped to recursive g' under some effective bijection π; and let ρ be any effective representation. (In particular, we have seen that good representations can be computed effectively.) The recursive function

$$\widehat{g}(n) := \rho(\pi^{-1}(g'(\pi(\rho^{-1}(n)))))$$

simulates g, since

$$\begin{aligned}
\widehat{g}(\rho(x)) &= \rho(\pi^{-1}(g'(\pi(\rho^{-1}(\rho(x)))))) \\
&= \rho(\pi^{-1}(g'(\pi(x)))) \\
&= \rho(\pi^{-1}(\pi(g(x)))) \\
&= \rho(g(x)))
\end{aligned}$$