philosophies

MDPI

Article

Computability, Notation, and *de re* Knowledge of Numbers

Stewart Shapiro[1],*, Eric Snyder [2] and Richard Samuels [1]

1 Department of Philosophy, The Ohio State University, Columbus, OH 43210, USA; samuels.58@osu.edu
2 Department of Philosophy, Ashoka University, Haryana 131029, India; eric.snyder@ashoka.edu.in
* Correspondence: shapiro.4@osu.edu

Abstract: Saul Kripke once noted that there is a tight connection between computation and de re knowledge of whatever the computation acts upon. For example, the Euclidean algorithm can produce knowledge of *which number* is the greatest common divisor of two numbers. Arguably, algorithms operate directly on syntactic items, such as strings, and on numbers and the like only via how the numbers are represented. So we broach matters of *notation*. The purpose of this article is to explore the relationship between the notations acceptable for computation, the usual idealizations involved in theories of computability, flowing from Alan Turing's monumental work, and de re propositional attitudes toward numbers and other mathematical objects.

Keywords: *de re* knowledge; notation; number; idealization

check for updates

Citation: Shapiro, S.; Snyder, E.; Samuels, R. Computability, Notation, and ***de re*** Knowledge of Numbers. *Philosophies* **2022**, *7*, 20. https://doi.org/10.3390/philosophies7010020

Academic Editors: Diane Proudfoot and Zhao Fan

Received: 27 December 2021
Accepted: 13 February 2022
Published: 18 February 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Saul Kripke [1] (p. 344) noted that there is a tight connection between computability and de re knowledge of natural numbers:

"... the notion of computability is best seen as providing a procedure for knowing *which number* is the value of the function." (emphasis added)

The purpose of, say, the Euclidean algorithm is to determine the identity of the greatest common divisor of two given numbers. A use of the algorithm to determine which number is the greatest common divisor of twelve and sixty-four seems to presuppose some sort of de re knowledge of the input numbers (twelve and sixty-four) and it produces de re knowledge that the result (four) is the greatest common divisor of those two numbers.

We thus broach issues related to two of the most vexed matters in philosophy. One is the so-called *access problem* for abstract objects, such as numbers. This is often traced to Paul Benacerraf [2]. If numbers are abstract, then how can we know anything about them? The issue is usually cast in terms of mathematical truths, and thus knowledge de dicto How do we manage to know that every number has a successor, that there is no largest prime number, etc.? Hartry Field [3] puts this as a problem for accounting for a (supposed) correlation between our mathematical beliefs and the (alleged) truth of those beliefs. Our problem here is not exactly that, but rather how we manage to have de re knowledge of particular numbers, such as twelve, sixty-four, and four. We clearly do have such knowledge, Field-style fictionalism aside. We seem to know which number is the greatest common divisor of twelve and sixty-four. How do we manage this? One would think that we can, and indeed we have, characterized computability, through the pioneering work of Alan Turing citeTuring, Alonzo Church [4], and others in the 1930s, without general solutions to these deep metaphysical and epistemological problems concerning mathematics.

The other vexed matter concerns de re propositional attitudes generally. Consider the question of what it is to know who a given person is. W. V. O. Quine [5,6], among others, pointed out that, in general, such matters are highly context-sensitive, and, in particular, interest-relative. For example, in asking who someone is, we sometimes know the name and are asking for the face: “Which of these people is Susan?”. Other times, we know the face and want the name: “Who is the person sitting over there, eating an apple?”. Sometimes, we know what a person did, and want a name: “Who stole all that the money from the train?”. Sometimes, we know the name and the face, and want something else: “See that woman over there; her name is Sally Brown. Who is she?”. Possible answers, in various contexts, might be “Pat’s partner”, “a professor of music at a local college”, etc.

Quine is famously skeptical that a fruitful account of de re propositional attitudes is even possible. Kripke [1] suggests that Quine overstates the case. And, to be sure, there are serviceable accounts of de re propositional attitudes in linguistic semantics. The accounts all allow for context sensitivity, and interest relativity, in the indicated ways. It is generally recognized that context sensitivity is a fact of linguistic life, and a large number of tools have been developed to accommodate this feature of natural languages. But the Turing-Church notion of computability in place today is not context sensitive, let alone interest relative, in these ways, or at least it is not usually taken to be. There is, for example, no context sensitivity in the notion of a Turing computable or recursive function, and these notions are taken to be extensionally equivalent to computability, via the co-called Church-Turing Thesis (or Theses).

De re propositional attitude reports concern ways that ordinary objects and people are *represented*, in language or in thought (if those are different). And here we encounter another vexed matter, this time in cognitive science and the philosophy of mind. What is representation, and how is it accomplished? In the present case, the issues concern how natural numbers are represented. Extreme (epistemic) Platonism aside, we have de re knowledge or belief about particular numbers only after we manage to represent them in language or thought. We thus broach matters of *notation*.

2. Notation

Shapiro [7] (p. 14) argues that computability applies directly only to functions on syntactic entities, such as strings on an alphabet:

> “Mechanical devices engaged in computation and humans following algorithms do not encounter numbers themselves, but rather physical objects such as ink marks on paper. As strings are the relevant abstract forms of these physical objects, algorithms should be understood as procedures for the manipulation of strings, not numbers. Furthermore, mathematical automata, such as Turing machines, which are the abstract forms of computation devices, have only appropriately constituted strings for inputs and outputs. It follows that, strictly speaking, computability applies only to string-theoretic functions and not to number-theoretic functions.”

To be sure, the general notion of computability does apply to number-theoretic functions. This invokes a notation, a function from a (presumably decidable) set of strings onto numbers. Typical presentations of Turing machines (including that of Turing [8]) use so-called *unary* notation, where a given number n is denoted by a sequence of n, or sometimes $n+1$, strokes. Other common notations are binary, decimal, hexadecimal, Roman numerals, scientific notation, etc.

Shapiro [7] points out that not any notation—not any function from strings to numbers—will do. For example, let X be any non-recursive set of numbers, say the set of codes of truths of first-order arithmetic. Intuitively, the characteristic function of X should not be computable.[1] Let the members of X be:

$$\{a_0, a_1, \ldots\},$$

in any particular order. And let the members of the complement of X be

$$\{b_0, b_1, \ldots\},$$

again, in any particular order. Now define a notation d_X as follows: if n is any natural number, then let the decimal numeral for $2n$ denote a_n, and let the decimal numeral for $2n+1$ denote b_n.

The characteristic function for X is computable via the notation d_X. The usual algorithm that determines whether a given numeral denotes an even number in the decimal notation also determines whether the number denoted by that numeral via d_X is a member of X.

At this point, one option is to concede that the notion of computability, as applied to number-theoretic functions, is relative to a notation, a kind of context-sensitivity. The characteristic function for X is computable via d_X but not computable via unary notation, decimal notation, etc., while the even numbers are computable via unary notation, decimal notation, and the like, but not via d_X (see [9]).

A better option, we submit, is to follow standard practice and resist this context-sensitivity, and thus to find fault with the notation d_X. The burden of Shapiro [7] is to articulate and defend a notion of *acceptable* notation, one that sanctions unary notation and decimal notation, but does not allow d_X. Along the way, an informal criterion is proposed, one that invokes the notion of de re knowledge of numbers. If a given notation is acceptable, then:

1. "The [agent] should be able to write numbers in the notation. If he has a particular number in mind, he should (in principle) be able to write and identify tokens for the corresponding numeral;"
2. "The [agent] should be able to read the notation. If ... given a token for a numeral, [the agent] should (in principle) be able to determine what number it denotes" [7] (p. 18).

Michael Rescorla [10] (p. 254) takes issue with the claim that, primarily, computability applies to syntactic entities, but we need not engage that matter here.[2] He defined a "semantics" for a set of symbols to be "a bijective mapping ... from the symbols [on]to the natural numbers." A "semantics" is thus what we call a "notation". Rescorla insists that an acceptable "semantics" must *itself* be "computable", despite the fact that a "semantics" is a function *onto* the natural numbers. The notion of a computable "semantics" is then glossed in terms of de re knowledge of numbers:

> "A semantics for some set of symbols is computable just in case there exists a mechanical procedure for computing *what number* a given symbol denotes ... [The] procedure succeeds only when the [agent] can *understand* the symbolic representations he manipulates. The [agent] need not know in advance which number a given symbol represents, but he must be capable, in principle, of determining *which number* the symbol represents ... [And if an agent] manipulates syntactic items that possess a noncomputable semantics, then he cannot mechanically determine *which number* a given symbol denotes" [10] (pp. 260–262, first and last two emphases added).

So we now turn to matters de re.

3. De re Attitudes toward Numbers

Kripke's Whitehead lectures [12] deal with de re propositional attitudes toward natural numbers. He calls a term t denoting a given quantity a *buck-stopper*, for a given person (at a given time), if the question "How much is t?" makes no sense, for that person (at that time). It is not merely a matter of not being able to give an answer that is any more informative than the buck-stopper. The idea is that if a speaker asserts or hears a sentence using a buck-stopper, for that person, then she knows *which quantity* is being considered. That is why it makes no sense to ask "But how much is that?", in that context, for that person. In the case of quantities, buck-stoppers thus appear to be what David Kaplan [13] calls "vivid designators".[3]

Clearly, the the notion of "buck-stopper" is context-sensitive if anything is. What counts as a 'buck-stopper', in a given context, depends both on the quantity it denotes and the state of the person at the time. Consider terms for distance. Suppose that someone is in Paris, during the pandemic, and is chatting over the phone with a friend in Kansas. Her friend asks about the social distancing conventions in place in Paris, and she tells him that people are to stay two meters apart from each other. He might then ask "How far apart is that?", and she might answer "A little over 6 and a half feet". In all likelihood, further inquiry on the part of the friend in Kansas is silly. The final statement in the dialogue contains a buck-stopper for him. He now knows de re what the relevant distance is. Of course, it could go in the other direction, if someone else is in the U.S. speaking to someone else in Paris, and the latter does not know how far "six feet" is. Something similar applies to currency (US Dollars, Euros) and temperature (Fahrenheit, Celsius), weight, volume, etc.

There are also buck-stoppers and non-buck-stoppers concerning ordinary numbers, independent of their use to measure distance, currency, temperature, etc. Indeed, there are buck-stoppers for pure arithmetic. And the context sensitivity is manifest. For someone who is taught it, unary notation is fine as a buck-stopper for sufficiently small natural numbers, say those less than 5. But if given a unary numeral for a larger number, say ||||||||||||||||||||||||, it would be fair for an interlocutor to ask, "What number is that?" For most educated folks, from Western societies, decimal notation would do as a buck-stopping answer here: 24.

Rescorla [14] (pp. 201–202) argues:

> "Unary notation is a poor vehicle for numerical computation. For instance, if forced to compute over a large number as presented in unary notation, any normal human would immediately translate into some more legible notation, such as Arabic decimal notation, scientific notation, etc., ... [U]nary representation is basically useless for normal computation involving large numbers.
>
> Unary notation is inefficient, because its demands upon storage space rise alarmingly with the size of numerical inputs. Other notations allow the thinker to represent large numbers much more efficiently."

For ordinary human beings, of course, Rescorla's observations are correct. No one would try to compute, say, 52^4 in unary notation. The answer is 7,311,616, and so the result of the computation would have (exactly) that many strokes. The same goes (or almost goes) for Roman numerals: the result would be 7311 M's followed by DCXVI. In both cases, it is surely reasonable to ask, "but what number is that?".

Tyler Burge [15] (p. 73) points out that even decimal notation will not work for sufficiently large numbers: "One needs to do some figuring, calculating, grouping, or simplifying of a thirty-seven figure numerical name to grasp which number it names." The following is an approximation for Avogadro's number: 602214090000000000000000.

It seems fair for someone to ask: "What number is that?". So the buck has not stopped. As Burge notes, it helps a bit, in this case, to group the digits in the usual manner: $602, 214, 090, 000, 000, 000, 000, 0004$.

But scientific notation is better: 6.0221409×10^{23}. For those familiar with this notation, the buck is perhaps stopped.

We might add that some numbers do not have buck-stopping notations at all. Consider a product of two primes, each with sixty digits. A (representation of a) number such as that might be sent from one computer to another to set up a secure digital transaction. Even if one were to gaze at a page that contained a decimal numeral for that number, it seems sensible to ask "But what number is this?" And no answer seems to be forthcoming, at least none that stops the buck—given that we need the exact number here, and not an approximation. Even scientific notation would have to list all of the 119 or 120 digits, as every one of them is relevant for the task at hand.

Rescorla [14] (p. 201) also points out that the standard notations, scientific, decimal, unary, etc., will not do for agents that do not know those notations:

"Someone might be familiar with unary notation but not Roman numeral notation. Thus, someone might believe that numerical function f [is computable relative to unary notation] without having the conceptual resources to contemplate [whether it is computable relative to Roman notation]. Even if one has the requisite conceptual resources, one might rationally believe that f [is computable relative to unary notation] while doubting that it [is computable relative to Roman], or vice versa."

These observations are all correct, and they do point to an interesting batch of questions concerning the semantics of reports of de re propositional attitudes about numbers. However, as above, we might be loath to conclude, against standard practice nowadays, that computability itself is relative to notation (against [9]).

4. Idealization

We submit that the the usual idealizations involved in theorizing about computability, along with what is known about the semantics of numerical terms, provide the key to resolving these issues.

From the beginning—that is, from the 1930s—theorists have focused on *human* computation, people calculating functions by following algorithms.[4] Only later was attention turned to machine or physical computation. And from the beginning, the focus has always been on *idealized* human agents. This is made manifest by speaking of what our agents can do "in principle".

Without these idealizations, there are many recursive functions that are not computable in any realistic sense. For example, a standard Ackermann function (defined via a double recursion) is not computable: one cannot compute its value at, say $\langle 5, 5\rangle$, for the simple reason that (so far as can be determined) the entire physical universe does not contain enough material to express this output, let alone compute it.[5]

The idealizations are familiar: we assume that the (human) agent will not run out of time, attention, or material, and will follow the instructions faithfully and accurately. We imagine agents that are immortal and infallible, but otherwise human (whatever that might mean). Similarly, we assume that each Turing machine has a (potentially) infinite tape, and that there are no bounds on the number of states it can have.[6]

Similar idealizations are standard in mathematics, and have been well before Turing, Church, and others started pondering the limits of computability. The first postulate of Euclid's *Elements* is: "A straight line segment can be drawn joining any two points", and the third is: "Given any straight line, a circle can be drawn having the segment as radius and one endpoint as center". No limits are specified on how far apart the endpoints might be from each other. The geometer does not worry about whether it is possible to draw a line between two points that are so far apart that no one can connect them in her lifetime, nor whether it is physically possible to build a straightedge that is big enough to draw the given line (say one from the center of gravity of the solar system to the center of gravity of a distant star). Our focus here is on what sorts of notations are acceptable for our ideal agents, those not encumbered with limits in attention span, lifetime, materials, and the like.

This shows that for present purposes—determining the limits of what is computable—matters of feasibility and efficiency raised by Rescorla, Burge, and others are simply not relevant. We cannot very well care *at all* about the demands of storage space if we are to declare the Ackermann function computable. As noted above, the universe does not contain enough storage space to compute this function for even small inputs.

The conclusion of Shapiro [7] is that a given notation N for natural numbers is "acceptable" just in case the successor function, given that notation, is computable. That is, a notation N is acceptable if and only if there is an algorithm on strings such that, given a numeral in N for a natural number n, produces the N-numeral for $n + 1$. Equivalently, the proposal is that N is acceptable just in case the following set of pairs of strings is effectively decidable: $\{\langle n, m\rangle \mid m$ is the N-numeral for the successor of the number denoted by N-numeral $n\}$.

The usual suspects—unary, decimal, binary, hexadecimal, and Roman numerals—are all acceptable, in this sense. And, of course, the above notation d_X is not.

Jack Copeland and Diane Proudfoot [17] (p. 251) give pride of place to unary notation, proposing "Turing's Notational Thesis":

> "Any job of work that can be done by a human computer engaged in numerical calculation can be carried out equivalently by a human computer employing [unary] notation."

In light of the standard idealizations, Turing's Notational Thesis is equivalent to the proposition that computations use a notation that is acceptable in the sense of Shapiro [7].

5. Connecting the Dots

It remains to show that the main proposal of Shapiro [7] and what Copeland and Proudfoot [17] call "Turing's Notational theses" are correct, in the relevant sense. Given the idealizations, do algorithms using stroke notation, or an acceptable notation in the sense of Shapiro [7], deliver knowledge of which number is the output of the given function for the given inputs?

We noted that, in general, de re propositional attitudes are context sensitive and interest relative, at least for for ordinary humans. We do not propose a counterfactual analysis of what sort of goals and desires our ideal agents would have, given that they follow algorithms flawlessly and have no finite bounds on their time, attention spans, and storage, and yet are still human—whatever that might mean. Indeed, one might think that idealized agents such as these are not even the sorts of creatures that *can* have de re propositional attitudes.[7]

We suggest that, in the case of natural numbers, one can still make sense of a buck-stopping notation for our ideal agents. This would be a notation in which, for a given numeral M, the identity of the number denoted by M flows directly and (more or less) immediately from the structure of M and the nature of the natural numbers themselves. If that holds, then, given the idealizations in place, it does not make sense to go on and ask "But which number is that?".

Settling this, however, requires insight into what the natural numbers *are*, and so we encounter yet another vexed philosophical issue, one that has troubled philosophers since ancient times. We submit, however, that for present purposes, the idealizations make the issue tractable and do indeed privilege unary notation, as well as acceptable notations in the sense of Shapiro [7]. Our modest proposal is to examine the role of number expressions in ordinary contexts, and then invoke the idealizations.

It is generally recognized that there are three primary uses for numerals. There are interesting empirical and conceptual questions concerning how these uses are learned, in what order, and how the various semantics for the different uses relate to each other, but we need not engage those issues here (see, for example, [18,19]). In the context of the standard idealizations, all three of these primary uses suggest a special or privileged role for unary notation and for the successor function on strings that denote numbers, the central item in acceptable notations according to Shapiro [7].

One use for numerals is to specify *cardinalities*, as in "Jupiter has four moons" and "A baseball defense consists of nine players". A cardinal number canonically answers a "How many?" question. An expression such as "the (cardinal) number of" expresses a function that takes a set, group, concept, property, plurality—anything with elements or members—and delivers the the size of that set, group, concept, property, plurality, etc. A cardinal number is the value of this function for some set, group, concept, property, plurality, etc. Frege's [20,21] logicist account of arithmetic takes natural numbers to *be* (finite) cardinals, as does the abstractionist program of Bob Hale and Crispin Wright (e.g., [22]), Neil Tennant's [23] neo-logicism, and many others. The cardinal notion also seems to be the one that children learn first, via counting (to the delight of their parents and grandparents).

With unary notation, the numeral for each number n is a sequence of n (or sometimes $n + 1$) vertical strokes. So the connection between each numeral and the corresponding cardinality is directly displayed. With the usual idealizations in place, there is no serious question about which number a given unary numeral denotes. Surely, if we are putting aside questions of time, space, memory, and the like, then unary notation is a buck-stopper for cardinal numbers. If one wants to know which cardinal number a given unary numeral denotes, just look at the number of strokes it contains (or one less than the number of strokes it contains).

It might be added that the typical way that we determine the cardinality of a small (but not too small) collection is to count it, using what Paul Benacerraf [24] dubbed "transitive counting". One recites numerals, typically decimal numerals, while pointing to each item in the collection being counted. This suggests a central role for the successor function on the numerals. To engage in transitive counting, one must have a way of reciting the next numeral after any given one. In other words, the agent must deploy the successor function on the numerals being used.

Rescorla [10] (p. 269) concedes this:

> "This proposal [from Shapiro [7]] receives powerful support from the crucial role the natural numbers play in counting. Typically, we measure cardinalities by enumerating elements of some numerical notation in ascending order. This procedure only works if the successor operation is computable relative to the notation ... [This] reflects an inherently desirable property of notations."

In sum, given that we are invoking the idealizations and setting aside matters of lifetime, attention span, and memory, if the successor function is computable for a given notation, and the agent knows the procedure for (transitive) counting, then numerals in that notation are buck-stoppers. There is no serious question concerning which number a given numeral in that notation denotes.

A second use of numerals, in ordinary language, is as an *ordinal*. In English, ordinals are denoted with expressions such as "third", "ninth", and "sixty-fourth". Sometimes, ordinary numerals are used, as in "Jack is contestant four", and "Bachelor number three".

One can think of "the ordinal number of" as a function, albeit one a little more complex than "the cardinal number of". It applies to an object with respect to a (finite) linear ordering that includes the object. It delivers the place of the object in the ordering. So to say that Joe is one's sixth child is to say that Joe occupies the sixth position among the indicated children, in birth-order. An *ordinal number* is the value of the function for some object-ordering pair.[8]

Again, a numeral in unary notation is a sequence of strokes. The strokes display an *ordering*, say left to right. So each unary numeral directly represents the corresponding ordinal. The last (i.e., rightmost) stroke in the numeral is in the requisite position in the displayed ordering for the corresponding ordinal. So, imposing the standard idealizations, unary numerals are buck-stoppers for ordinal numbers as well.

It is often noted that the procedure of (transitive) counting also invokes an ordinal. When someone counts a collection, she, in effect, imposes an ordering on it, given by the order in which the objects are counted. The last numeral mentioned is thus the ordinal of the last object counted, within the imposed order. Of course, that same numeral also designates the cardinality of the collection. To be sure, the particular order chosen for the count does not matter—every ordering (of the right kind) will produce the same result[9]—but any use of transitive counting does invoke an order, and thus an ordinal. And, as noted, counting presupposes the successor relation on the numerals: the agent must always know how to determine the next numeral. So here, too, we have a special role for the successor function on the notation used in the procedure.

The third use of numerals may be called *numerical*. It is used to talk about numbers themselves. Consider statements such as:

- Six is a number;
- Five is prime;
- The first four perfect numbers are 6, 28, 496, and 8128;
- Nine is Susan's favorite number;

- There is a lot of folklore concerning the number seven.

The natural numbers satisfy the Dedekind–Peano axioms. These are often characterized in a formal language with a symbol '0' for zero and one-place function symbol 's' for the successor function.

1. $N0$;
2. $\forall m(Nm \rightarrow Nsm)$;
3. $\forall m \forall n((sm = sn \wedge Nm \wedge Nn) \rightarrow m = n)$;
4. $\forall m(Nm \rightarrow sm \neq 0)$;
5. For any property F of numbers, if F holds of zero, and for any natural number n, if F's holding of n implies that F holds of the successor of n, then F holds of all natural numbers.

The first axiom is that zero is a natural number; the second is that every number has a unique successor; the third is that the successor function is one-to-one; the fourth is that zero is not the successor of a natural number; and the fifth is the induction principle.

Taken together, the Dedekind–Peano axioms characterize the natural numbers as forming an ω-sequence. Arguably, these are the central features involved in the numerical uses of numerals (if not also the cardinal and ordinal uses). So here, too, we see a privileged role for the successor function. From this perspective, a natural number *just is* either zero or the result of applying the successor function to zero a finite number of times.

The formal languages for arithmetic contain what can be taken as a *canonical* notation for the numbers. A number n is denoted by a string of n s's, followed by a 0. This is clearly a variant on unary notation, using the sign for the successor function instead of a stroke. So the canonical numeral for a number n directly displays which number n denotes—the canonical numeral displays just how many times one applies the successor function to zero in order to arrive at the indicated number. And, given the idealizations, this numeral is a buck-stopper.[10]

6. Beyond the Natural Numbers

We close with brief accounts of other mathematical domains. A buck-stopping notation for an integer, at least for our ideal agents, would be a buck-stopping notation for a natural number, together with a sign, '$+$' or '$-$'. So there seems to be nothing further to making sense of matters of computation, and de re propositional attitudes for those mathematical entities. Similarly, a rational number is a ratio of integers. So a suitably idealized buck-stopping notation for a given rational number is $\frac{m}{n}$, where m is a buck-stopper for an integer, n a buck-stopper for a natural number other that zero, and the two numbers are relatively prime. One can perhaps develop a buck-stopping notation for finite sets or finite sequences of natural numbers, using common coding techniques.

After this, however, things do not go nearly as smoothly. Within mathematics, the natural numbers, the integers, the rational numbers, and finite sets or sequences of those, are more the exception than the rule. For this reason, philosophy of mathematics should not focus exclusively on these systems.

Consider the real numbers. There is no hope for a canonical notation for real numbers. Indeed, no language with a finite (or countable) alphabet can contain a name for every real number, let alone a canonical name, let alone a buck-stopping name, even allowing the idealizations. Even the languages of our idealized agents contain only countably many denoting expressions. Yet there does seem to be an intuitive sense of a given function being computable, and for even actual humans to have de re propositional attitudes toward at least some real numbers. We seem to know de re, for example, that the volume of a (Euclidean) sphere with a radius of one meter is $\frac{4\pi}{3}$ cubic meters.

One might begin by restricting attention to so-called recursive real numbers. But how are those denoted? One might think of a (recursive) real number as given by a Turing machine that prints its decimal expansion on an initially blank tape. Using this notation, however, the addition function, on recursive real numbers, is not itself recursive. So, via Church's Thesis, addition in not computable. Indeed, the addition function on such numbers is equivalent to the halting problem.[11]One might bite this bullet, and declare that addition, on these real numbers, is not computable, but one might wonder if there are any non-trivial computable functions.

In treatments of computation over real numbers, it is more common to think of a (recursive) real number as given by an effective Cauchy sequence of rational numbers. Then there are indeed straightforward algorithms for addition, multiplication, exponentiation, and the like—in most cases, the operations can be performed pointwise, on the corresponding rational numbers in the sequences. However, with both decimal expansions and Cauchy sequences, it is not at all clear that we are gaining de re knowledge of individual (recursive) *real numbers*, as it is not decidable whether two different Turing machines (say) converge to the same number. If we agree on a canonical notation for Turing machines (or recursive function), then we may be gaining de re knowledge of Turing machines, but that is hardly a gain, since, for these purposes, Turing machines are equivalent to natural numbers.

In the usual presentations of the complex numbers, as the algebraic closure of the real numbers, a complex number is indiscernible from its conjugate. Anything true of $a + bi$ will also be true of $a - bi$. So there does not seem to be any sense to be made of de re knowledge of one of the square roots of -1, as opposed to the other one (i.e., i vs. $-i$, see [27]).

Things are even worse in Euclidean geometry. The space is entirely homogeneous: every point is indiscernible from every other point; every line is indiscernible from every other line; every line segment from any congruent line segment, etc. There simply is no way to single out a given point, line, or segment and thus denote it. So it would seem that there just is no de re knowledge of individual points, lines, line segments, etc., at least not in the language of geometry.

However, it does seem intuitive to think of, say, Euclidean construction, invoking an unmarked straightedge and a compass, as a kind of computation, a sort of algorithm. And so, as above, it is natural to think of a given construction as telling us, say, *which* point is the midpoint of a given line segment, or *which* which line passes through a given point and is parallel to a given line.

Perhaps the best way to look at this is a kind of *relative* de re attitude. If we are somehow given a line segment, then we can locate (de re) the midpoint of that line segment. We draw two circles from what we are given (each with the radius of the given line segment and each with an endpoint as center) and then draw a line connecting the points where the circles intersect. The midpoint is the intersection of that line with the given one.

This, however, is not the place to make progress on this relative notion of de re knowledge. We rest content with noting the success in the case of natural numbers (and the like), via the longstanding idealizations.

Author Contributions: These authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors would like to declare that there are no conflicts of interest.

Notes

[1] The characteristic function f_X of a set X is a function f such that $fn = 1$ if $n \in X$ and $fn = 0$ otherwise. It is a corollary of the usual proof of the incompleteness theorem that the characteristic function of the set of arithmetic truths is not recursive.

[2] Rescorla [11] (p. 338) later concedes that "Turing computation over a non-linguistic domain presupposes a notation for the domain".

[3] Quine [6] (p. 9) suggests that a vivid designator "is the analogue, in the logic of belief, of a rigid designator".

4 See, for example, Turing [8] and Church [4].

5 See [16] for an amusing thought experiment involving an Ackermann-style function.

6 When machine computation was brought into the picture, similar idealizations were assumed for the "machines". It is assumed, for example, that they do not decay or otherwise break down.

7 We are indebted to an anonymous referee for pressing this issue.

8 Øystein Linnebo [25] provides an abstraction principle for ordinals, in this sense, and argues that it occupies a fundamental role in cognitive architecture. In logic and mathematics generally, an "ordinal" is sometimes taken to be the order-type of a well-ordering (or a surrogate for this, in set theory). That is a generalization of the usage of the term in focus here.

9 Gelman and Gallistel [26] label this the "Order-Irrelevance Principle".

10 In rigorous axiomatizations of arithmetic, it is, of course, common to add symbols for addition and multiplication, along with recursive axioms for those functions. This leads to non-canonical ways to denote individual numbers. For example, $ssss0 \times ssssss0$ denotes twenty-four, or ssssssssssssssssssssssss0.

11 Suppose, for example, that one of the inputs starts 0.333 . . . and the other starts 2.666 . . . . The sum of those two numbers will start with either a 2 or a 3 depending on whether there is a digit of the inputs that adds up to 10 or more. And, short of solving the halting problem, there is no way to know that (see [11]).

References

1. Kripke, S. *Philosophical Troubles*; Oxford University Press: Oxford, UK, 2011.
2. Benacerraf, P. Mathematical Truth. *J. Philos.* **1973**, *70*, 661–679. [CrossRef]
3. Field, H. *Truth and the Absence of Fact*; Oxford University Press: Oxford, UK, 2001.
4. Church, A. An unsolvable problem of elementary number theory. *Am. J. Math.* **1936**, *58*, 345–363. [CrossRef]
5. Quine, W.V.O. *The Ways of Paradox and Other Essays*; Random House: New York, NY, USA, 1966.
6. Quine, W.V.O. Intensions revisited. *Midwest Stud. Philos.* **1977**, *2*, 5–11. [CrossRef]
7. Shapiro, S. Acceptable notation. *Notre Dame J. Form. Log.* **1982**, *23*, 14–20. [CrossRef]
8. Turing, A. On computable numbers, with an application to the Entscheidungsproblem.*Proc. Lond. Math. Soc.* **1936**, *42*, 230–265.
9. Brauer, E. The dependence of computability on numerical notations. *Synthese* **2021**, *198*, 10485–10511. [CrossRef]
10. Rescorla, M. Church's thesis and the conceptual analysis of computability. *Notre Dame J. Form. Log.* **2007**, *48*, 253–280. [CrossRef]
11. Rescorla, M. The representational foundations of computation. *Philos. Math.* **2015**, *23*, 338–366. [CrossRef]
12. Kriple, S. Wittgenstein, Russell and our conception of the natural numbers. In *Mathematical Knowledge, Objects and Applications: Essays in Memory of Mark Steiner*; Ben-Menahem, Y., Posy, C., Eds.; Springer: New York, NY, USA, 2022, *forthcoming*.
13. Kaplan, D. Quantifying in. *Synthese* **1968**, *19*, 178–214. [CrossRef]
14. Rescorla, M. Copeland and Proudfoot on computability. *Stud. Hist. Philos. Sci. A* **2012**, *43*, 199–202. [CrossRef]
15. Burge, T. *Foundations of Mind*; Oxford University Press: Oxford, UK, 2007.
16. Boolos, G. A curious inference. *J. Philos. Log.* **1987**, *16*, 1–12. [CrossRef]
17. Copeland, B.J.; Proudfoot, D. Deviant encodings and Turing's analysis of computability. *Stud. Hist. Philos. Sci. A* **2012**, *41*, 247–252. [CrossRef]
18. Snyder, E.; Shapiro, S.; Samuels, R. Cardinals, ordinals, and the prospects for a Fregean foundation. *R. Inst. Philos. Suppl.* **2018**, *82*, 77–107. [CrossRef]
19. Snyder, E. *Semantics and the Ontology of Number*; Cambridge University Press: Cambridge, UK, 2021.
20. Frege, G. *The Foundations of Arithmetic*, 2nd ed.; Austin, J., Translator; Harper: New York, NY, USA, 1960.
21. Frege, G. *Gottlob Frege: Basic Laws of Arithmetic*; Ebert, P.A., Rossberg, M., Translators; Oxford University Press: Oxford, UK, 2013.
22. Hale, B.; Wright, C. *The Reason's Proper Study*; Oxford University Press: Oxford, UK, 2001.
23. Tennant, N. *Anti-Realism and Logic*; Oxford University Press: Oxford, UK, 1987.
24. Benacerraf, P. What numbers could not be. *Philos. Rev.* **1965**, *74*, 47–73. [CrossRef]
25. Linnebo, Ø. The Individuation of the Natural Numbers. In *New Waves in Philosophy of Mathematics*; Bueno, O., Linnebo, Ø., Eds.; Palgrave: London, UK, 2009; pp. 220–238.
26. Gelman, R.; Gallistel, C. *The Child's Understanding of Number*; Harvard University Press: Cambridge, MA, USA, 1978.
27. Shapiro, S. An 'i' for an i: Singular terms, uniqueness, and reference. *Rev. Symb. Log.* **2012**, *5*, 380–415. [CrossRef]