Causality Arguments [second draft]*

Edgar Daylight†

February 22, 2022

`d(`n`) = if` $a \to|^n$ `then` a `else d(`$n+1$`)`

`And` $a \to|^n$ `means expression` a `converges in` $\leq n$ `steps`

`Observation: d(0) converges if and only if` a `converges`

I recently came to the conclusion that the "Observation" in this snippet — as presented by Felice Cardone and myself in a slightly more involved form at a PROGRAMme workshop in June 2019 — hinges upon a conflation of two accounts:

1. a *dynamic* (step-by-step) analysis of computability
2. a *static* understanding of computability

Cardone and I believed that the "if and only if" observation follows mathematically from the rest of the snippet without reasoning beyond *every* conceivable finite bound; that is, without reasoning *at* infinity. Examining computability solely in terms of (increasing) finite bounds is akin to 1., while reasoning *at* infinity is akin to 2. Since I have always wanted to detect a conflation in my own work, after having done so [5, 7] in other sources, I shall eagerly discuss it in the present chapter.

1 Terminology

Cardone and I eschewed actual infinities, i.e., we were not *actualists.* Rather, we were *potentialists.* Moreover, one of us was a *liberal potentialist* and the other was a *strict potentialist.* (Or so I shall assume in the present exposition. In reality, neither Cardone nor I were consistently championing one specific philosophical position.) Abiding by an analysis of Øystein Linnebo and Stewart

*The title will have to be modified.

†The author, also known as Karel Van Oudheusden, was financed by SFB 1187 "Medien der Kooperation" (Siegen University) and by ANR-17- CE38-003-01 "PROGRAMme" (Lille University).

Shapiro pertaining to first-order logic, a *liberal potentialist* never takes exception to the Law of the Excluded Middle, meaning that his logic of potential infinity is classical logic [16]. The *strict potentialist*, in contrast, does expect every truth to be "made true" at some stage (e.g., of the computation at hand). Universal quantification over all natural numbers with classical logic presupposes actual infinity for the *strict potentialist,* but not for the *liberal potentialist* [16, p. 184].

These stipulations come from Linnebo & Shapiro's recent analysis in which they conceptually connect the intellectual positions of Aristotle and Plato with, respectively, dynamic terminology and procedures (on the one hand) and static, mathematical truths (on the other hand). Extrapolating to developments in computer science, an *operational* semantics and a *denotational* semantics arguably belong to, respectively, account 1. and account 2. Likewise, an *intensionalist* perspective and an *extensionalist* vantage point typically comply with, respectively, the first and the second account. Yet, the weasel words "arguably" and "typically" in the previous sentences (and in the following paragraphs) indicate my reluctance to proclaim any definite mastery of the concepts at hand.[1] Although the various dichotomies mentioned in this chapter overlap each other, they are not *per se* identical to each other.[2]

Finally, I also distinguish between *causal* and *abstract*, since this ontological distinction prevails in the philosophical literature on Turing machines. (The Appendix provides key examples.) The word "causal" fits in a stage-by-stage account of a process (cf. 1.), while the word "abstract" typically belongs in a Platonic exposition (cf. 2.). Pivoting on this dualism, two separate realms can now be entertained:

> ***A.*** The territory of causal computations of the computer scientist — in which, at best, potentially-infinitely long computations are permitted. The computer scientist reasons operationally about buildable machinery that can be extended whenever required by an external agent.
>
> ***B.*** The non-causal map of the computability theorist — in which completed infinities are welcomed. The theorist examines computations with abstract objects in a Platonic realm of number-theoretic functions.

The *A*-*B* divide serves a heuristic role: some historical players fit into this scheme and others less so. To illustrate this point, I briefly sketch a contrast between Martin Davis's 1958 book on computability [4] and the 1993 approach of Robert Constable and Scott Smith [1]. Davis took "computations" to be syntactic and he examined them *with* abstract objects. Constable & Smith, however, took a computation *itself* to be abstract, i.e., they examined computations *as* abstract objects.[3] Placing Davis in realm *B* is a bit more natural

[1] It is, for example, perfectly possible to reason Platonically (i.e., statically, so to speak) about an operational semantics.

[2] I thank Julian Rohrhuber for conveying this insight to me.

[3] Which is presumably the standard approach in *denotational* semantics. This reflection, along with others, come from personal correspondence with Felice Cardone.

than placing Constable & Smith in that same realm, although all three authors favored an extensionalist narrative (cf. realm B). Moreover, positioning Davis in realm B would need further explication, because Davis *also* reasoned, albeit informally, inside realm A. Conceptual complications abound.

The divide between realms A and B can be recast as a difference between engineering and pure mathematics. The words "causal computation," used to describe realm A, can be directly or indirectly associated with material devices. In this regard, I mention Carl Petri who took the standard view on Turing machines to be in accord with an outdated Newtonian understanding of the world; that is, he placed Martin Davis's 1958 book inside realm B. Petri took his engineering version of "Turing machines" to belong to realm A and in accordance with modern physics [19]. The tape of his Turing machinery was potentially infinitely long, and a careful reading of his work reveals that he eschewed number theoretic functions, undecidability claims, and the like.

2 Historical Reflections

The emerging narrative in my recent writings is, that, several computer scientists have a tendency to conflate the territory of causal computations with the non-causal map of the theorist, wanting to have the best of both worlds: buildable Turing machines subject to undecidability results from pure mathematics.[4] This is a category mistake of the first order, as in my reading Carl Petri [19] and Paul Henry [13] have pointed out before. For example, the notion of undecidability pertaining to digital machinery is only a coherent notion in the formal, non-causal setting of actual infinity, not when reasoning causally with potential infinity. Causal Turing machines — that is, buildable Turing machines — belong to realm A, which is separated from realm B of mathematical Turing machines and pure mathematics. Only the latter, mathematical category contains undecidable problems. These thoughts are crisply formulated by Henry:

> The machines we have considered here, the combination of the Euclidean straightedge and compasses, Descartes' machine and the Turing machines, have all of them had something to do with the foundations of mathematics. I have insisted upon the fact that those machines have had a theoretical function and that there was no need to materially construct them for them to be operational from that point of view. Furthermore, I said that in reality they cannot be constructed. Now, I can add that it is precisely as "machines impossible to materially construct" that they give rise to impossible problems. [13, p. 121, original quotes, my emphasis]

In contrast to Petri and Henry, many computer scientists do not consistently distinguish between (say) Petri's causal Turing machines and Davis's Platonic

[4] The present paragraph borrows largely from one of my articles (currently under peer review).

Turing machines. Only the latter allow for extensionality, which is, so I shall maintain, a prerequisite for deriving an impossibility result. To be more precise, these computer scientists actually believe that their "potentially infinite" Turing machines suffice in Davis's number-theoretic setting. I present various examples and clarifications below.

At the outset of any discussion pertaining to Turing computability, I assume all parties agree that any Turing machine M must be able to accept, as input, a suitable encoding of *any* natural number. Hence, M's input string can be arbitrary long. Many computer scientists in realm A will insist that some external agent can always extend M's tape when required,[5] thereby apparently avoiding the need of an infinitely long tape. The theorist in realm B will, in adherence to recursive function theory, stress that the notion of an 'external agent' is foreign to any formal definition of a Turing machine. An infinitely long tape is a *conditio sine qua non* for the computability theorist to get the mathematics right.

At this point the computer-science critic might insist that in standard presentations of Turing machines, the machine's tape is assumed to be only potentially infinitely long. At first glance, this claim seems to be persuasive, as also the following words from Linnebo & Shapiro seem to indicate:

> The notion of potential infinity is still with us, perhaps in a more subtle form. It is now a commonly held view in linguistics that languages are infinite. Noam Chomsky, for example, once wrote that a grammar projects from a finite corpus to a "set (presumably infinite) of grammatical sentences" (p. 15). It is, we think, more natural to think of a language as potentially infinite. For another example, one of us asked a teacher about the infinite tape of a Turing machine. He was told that we do not have to think of the tape as (actually) infinite. It is enough to live near a tape factory. [16, p. 187]

I shall return to Linnebo & Shapiro's "tape factory" towards the end of this chapter. At second glance, it should be noted that the textbooks of Martin Davis [4], Stephen Kleene [15], John Hopcroft and Jeffrey Ullman [14], Marvin Minsky [17], and David Harel [12] all explicitly refer to an actual infinity. For example, Kleene wrote:

> To allow for unlimited storage of such information, we consider as separate the machine proper and a peripheral storage facility, which we will take to be an infinite "tape". [15, p. 233]

While potential infinity seems to suffice constructively, the "tape" has to be infinitely long nonetheless. For else Kleene's impossibility proof cannot get off

[5] Petri only agreed partially with this reasoning: once the tape becomes too long, one is required to chop the Turing machine into two smaller machines that communicate with each other asynchronously across a network. Furthermore, it is *not* possible to peruse the *global* state of the computer network, there is no notion of *global* time and the like. Newtonian physics is substituted by an Einsteinian Weltanschauung. Hence, reasoning extensionally about the entire network in some Laplacian framework makes no sense either.

the ground, a proof which is by contradiction and extensional [15, p. 245-246]. Similar observations hold for, say, Davis (1958) and Constable & Smith (1993). In sum: actual infinity and classical logic are salient in these expositions.

The critic continues, however, by remarking that David Hilbert in the 1920s only accepted potential infinities in order to solve the *Entscheidungsproblem*. Alan Turing in 1936 was thus not allowed to exploit actual infinity. By implication, the critic insists, the tape of a Turing machine is *not* infinitely long, only potentially so. I briefly respond as follows: Turing showed that, even if one resorts to Hilbert's Platonic realism, even then the *Entscheidungsproblem* is unsolvable. So it remains unsolvable inside Hilbert's finitistic Program as well.[6] Turing's overall argument was like Georg Cantor's in that both required an actual infinity.[7] I also refer to Guido Gherardi's "Alan Turing and the Foundations of Computable Analysis" [11, p. 397, 407], in which he not only attributes an *infinite* tape to Turing's 1936 exposition, he also shows how Turing explicitly relied on the law of the excluded middle (classical logic) in a 1937 correction to his 1936 article [24, 25].

The critic persists one last time, by mentioning the writings of Wittgensteinian scholar Juliet Floyd [8, 9]. According to Floyd, the critic remarks, Turing's 1936 reasoning was non-extensional. Specifically, Floyd takes Turing's 1936 diagonal argument to be constructive in nature, akin to Wittgenstein's 1947 eloquent exposition of what a diagonal argument came to mean to him:

> Let $N = F(k, n)$ be the form of the law for the development of decimal fractions.
>
> N is the nth decimal place of the kth development.
>
> The diagonal law then is: $F'(n) := F(n, n)$
>
> To prove that $F'(n)$ cannot be one of the rules $F(k, n)$, assume it is the 100th.
>
> Then the formation rule of F'(1) runs $F(1, 1)$, of F'(2) runs $F(2, 2)$ ... etc.
>
> But the rule for the formation of the 100th place of F'(n) will run $F(100, 100)$; that is, it tells us only that the hundredth place is supposed to be equal to itself, and so for $n = 100$ it is not a rule.
>
> The rule of the game runs "Do the same as ..." — and in the special case it becomes "Do the same as you are doing".[8]

My response consists of three observations. First, Floyd's writings are illuminating: they allowed me to grasp details in Turing's 1936 paper for the first time.

[6] I thank Erhard Schüttpelz for sharing this key insight with me in private correspondence in the spring of 2021.

[7] To recapitulate: how could Turing prove the *impossibility* of the *Entscheidungsproblem* by staying *inside* Hilbert's Program? He couldn't. He had to *step out of it* and show that *even then* the *Entscheidungsproblem* is unsolvable. For broader coverage, see both Alexander Zenkin's 2004 article [27] and a set-theoretical rebuttal of that article: *https://www.youtube.com/watch?v=qhZgABFnd0s*

[8] This Wittgensteinian fragment comes from Floyd [8, p. 35-36].

Second, I agree with Floyd and most specialists that Wittgenstein was not a Platonist, but I disagree with her when it comes to Turing. I take Turing to have been an idealist,[9] which would clarify the observations made above (cf. Turing's infinite tape) as well as other findings conveyed, e.g., by Kjeld Schmidt and Stuart Shanker [22, 23]. The crux, then, is that a diagonal argument as presented above meant something else to Wittgenstein than to Turing in 1936. For the former no actual infinity was in play; but there are no indications to suggest this was also the case for Turing, quite to the contrary. Wittgenstein's interpretation of his diagonal argument hinges on an operational notion of following a rule: a distinction can be made before and after one arrives at, say, the 100th place in the argument above. In a Platonic setting, however, this talk makes no sense: the entire diagonal already exists, extensionally; that is, no distinction is in order between the prescriptive (what the rule dictates) and the descriptive (what the rule has achieved).[10] To recapitulate, Wittgenstein would not have claimed that his diagonal argument had any extensional import. He would therefore have disagreed with Turing, Alonzo Church, Davis, et al. that "there exist" number-theoretic functions (let alone incomputable ones). Third, in her latest book on Wittgenstein, Floyd does remark — and aptly so from my vantage point — that at least part of Turing's 1936 paper is extensional (after all), for else Turing would not have been able to conceptually connect his automatic machines to the number-theoretic functions of Church [10].

As a final response to the critic, I venture to say that most computer science professors, today, teach Cantorian diagonal arguments by reasoning *both* operationally and extensionally.[11] It should be noted that Davis in 1958 was careful to separate his informal, operational commentary on computability from his formal, extensionalist account. Alas, conflating a step-by-step analysis of computability with a static understanding is precisely what I did myself in 2019, as I shall now clarify.

3 Analyzing the Snippet

Impossibility proofs from Turing (1936), Davis (1958), Kleene (1967), and others hinge on actual infinity and classical logic. Can the same be said of the mathematical snippet provided in the introduction of this chapter? Today I fail to see how that snippet can make any sense without appealing to an actual infinity. Specifically, the snippet's "only if" claim pertains to nontermination

[9] I recently gave a talk on this topic, entitled: "Church's Reception of Turing's 1936 Paper: A Philosophical Angle," at the 6th International Conference on the History and Philosophy of Computing, (ETH Zürich, 2021); extended abstract is available here: *https://hapoc2021.sciencesconf.org/resource/page/id/18*

[10] And it is this Russellian blending of the prescriptive and the descriptive which I have associated with Turing's position, in connection with a 1939 exchange between Turing and Wittgenstein [6]. Although Bertrand Russell tried to change his own ways, he remained a Platonist for years [18].

[11] I question whether such accounts are philosophically admissible, but I do not delve into these matters here.

(as we shall see) and, thus, expresses the *impossibility* of termination.

According to all parties involved, the snippet conveys code which is equivalent to that of a specific Turing machine M, which consists of two tapes:

- Tape 1 is used to evaluate expression a
- Tape 2 is the control tape and stores n's current value

In colloquial language, mathematical Turing machine $M(a)$ takes expression a as input and subsequently computes $d(0)$; that is, M halts with expression a as output or M fails to halt. Loosely speaking, M is akin to a universal Turing machine: it evaluates (or simulates) any expression (or machine description) a.

At this point the theorist in realm B will already insist that machine M is infinitely large. Since *any* expression a should be encodable on Tape 1, and since no notion of 'external agent' is permitted in computability theory proper, it follows that Tape 1 is infinitely long. But, playing the advocate of the computer scientist in realm A, no fundamental reason has been given so far to discount the possibility of concocting an alternative, agent-based account of Turing-machine theory in compliance with the philosophy of the computer scientist. The theorist will have to indicate *when* and *where* the computer scientist *literally* appeals to an actual infinity.

The theorist therefore asks the computer scientist to consider the "only if" part ($\Rightarrow$) of the "observation" in the original snippet:

$$\texttt{d(0) converges} \Rightarrow a \texttt{ converges}$$

How do we prove this implication? Since $d(n)$ is defined reductionally in terms of a, both the theorist and the computer scientist agree to proceed by perusing the contraposition:

$$a \texttt{ diverges} \Rightarrow \texttt{d(0) diverges}$$

As an *actualist*, the theorist has no objection to this move. Computer scientists who are *liberal potentialists* will not object either. But those computer scientists who are *strict potentialists* will complain. According to them, the "only if" part of the "observation" cannot be constructively proved without further givens.[12]

We are thus left with the actualist and the liberal potentialist. Now, "a diverges" means, by definition, that for any natural number m, we have: $a \rightarrow|^m$ does not pertain. Similarly, "$d(0)$ diverges" means that for <u>any</u> natural number m, we have: $d(0) \rightarrow|^m$ does not pertain; that is, $d(0)$ eventually calls on $d(m+1)$. So, what we do is we take an *arbitrary* value m and prove that $d(m)$ will call $d(m+1)$, which is easy to prove due to the *if-then-else* construct in the code snippet and our knowledge that $a \rightarrow|^m$ does not pertain, for any

[12] Attempting to do so nonetheless would require a definition of 'functional equivalence' (or of 'simulation') — as in 'one program is functionally equivalent to another' (or as in 'one machine simulates another') — which would require engagement with divergent behavior once again, thereby putting the cart before the horse. See, e.g., Constable & Smith [1, p. 107] for a technical explication of essentially the same issue.

m. So far so good for both the actualist (theorist) and the liberal potentialist (computer scientist).

Also, both actors agree to use induction to conclude what they had set out to prove in the first place: that $d(0)$ will eventually call $d(n)$ *for all natural numbers* n. At this point, however, the actualist will remark that Tape 2's length (which stores the values of n) must definitely be infinite. The potential liberalist immediately counters by appealing, once again, to his 'external agent' who will 'extend' the Tape 2 when needed. This time, however, the actualist has a more refined argument up his sleeve. He will stress that, for the specific a under scrutiny, it is a priori known (to both himself and the computer scientist) that Tape 2 will be extended indefinitely; that is, an actual infinite number of times. (For both actors have jointly proved that $d(0)$ diverges for the specific a under consideration, and divergence of $d(0)$ means that *every* natural number n will be stored on Tape 2 at some point.) So, if the potential liberalist in A sticks to the interventions of his 'external agent,' then the theorist in B can rightfully claim that he has formally specified *strictly more information* about the computation at hand (by appealing to an infinitely long tape) in comparison to the symbols jotted down on paper by the liberal potentialist. For, the latter lacks the mathematical apparatus to express the full-fledged nature of the computation at hand; instead, he has to appeal in some vague way to some external agent.

To recapitulate, two observations can be made. On the one hand, when a number-theoretic function f on some argument x is *told* to all parties involved that it is undefined — that is, $f(x) = \perp$ — then this *means* nontermination for both the actualist and the liberal potentialist.[13] On the other hand, to mathematically stipulate that a particular computation — i.e., that of $d(0)$ in our running example — diverges, is to specify that it never halts. And to rigorously jot that insight down, one has to express a property of *all* natural numbers. The actualist can do that without appealing to some external agent because his tape is infinitely long to begin with. The liberal potentialist who wishes to be equally rigorous on paper, will not only have to formally spell out the details of his external agent, he will have to specify the *infinitely many* interventions of that agent with regard to the computation at hand. But doing precisely *that* requires an appeal to a completed infinity after all.

4 The "Tape Factory"

I have argued that nothing less than a completed infinity will do, in order to get the mathematics of a nonterminating Turing-machine computation correct. Yet, it should be noted that Linnebo & Shapiro side with the liberal potentialist; that is, they champion the notion of an agent (external to the Turing machine's tape) who visits a "tape factory" on a regular basis [16, p. 187]. These authors

[13] Assuming that all parties have agreed to semantically treat $\perp$ in the same way, for else we are comparing apples to oranges. That is, if $\perp$ can mean something other than termination for, say, the actualist, then the same freedom should also be granted to the potentialist.

believe that potential infinity is all that is required, just like I did in 2019. I shall now explain why their advocacy of liberal potentialism is contradicted by their own analysis [16, p. 178].

Linnebo & Shapiro eloquently convey a distinction between the actualist (on the one hand) and "potentialists of all stripes" (on the other hand) in connection with what is known as the plural comprehension scheme:

$$\exists c.\, used(c) \rightarrow \exists r. \forall c.\, [c \prec r \leftrightarrow used(c)]$$

This ordinary unrestricted P-Comp scheme — in which $c \prec r$ reads: "c is one of some objects r" — conveys the following information:

> Provided that there is at least one used tape cell of the computation (which we are investigating extensionally), there is a tape region r of cells that comprise all and only the used tape cells.

A tape cell c is *used* if and only if the Turing machine has positioned its "head" over cell c during the computation at hand.

It should be noted that I am merely applying Linnebo & Shapiro's reasoning in the context of Turing machines; that is, the formulas presented here come (modulo a change in names of variables) from their own paper. Moreover, the antecedent in the previous formula is always true in the present discussion, since every Turing-machine computation has at least one used tape cell. Therefore, we obtain the simplified formula $\exists r. \forall c.\, [c \prec r \leftrightarrow used(c)]$, which conveys, that:

> There is a tape region r of cells that comprise all and only the used tape cells — again, when reasoning extensionally.

In our running example pertaining to $d(0)$ and its divergent behavior, r represents a mathematical object that is *infinitely* large. That, in brief, is the position of the actualist (i.e., the theorist in realm B).

Coming now to potentialists of all stripes and particularly the liberal potentialist in realm A, I shall follow Linnebo & Shapiro and conclude that he does not — in fact, he cannot — accept the P-Comp scheme. The liberal potentialist accepts a formula in classical logic when its modal translation into a well-specified modal logic (as detailed in the paper by Linnebo & Shapiro) holds true. Specifically, a necessary condition for the potentialist to accept the previous formula (in classical logic) is that the following formula (in modal logic) is true:

$$\Diamond \exists r.\, \Box \forall c.\, [c \prec r \leftrightarrow used(c)]$$

Informally, this modal predicate states that:

> It is possible for there to be some tape region r — at some finite stage of computation, for we are now reasoning intensionally (step-by-step) — which *necessarily* comprises all and only the used tape cells.

But this statement is clearly false in general. For example, if the modal formula holds true in our running example, then there should be a finite threshold r of tape cells which the diverging computation $d(0)$ uses and only uses. But this is clearly not the case (for all parties involved). So here we have, to quote Linnebo & Shapiro, "a clear logical difference between actualism, on the one hand, and the two forms of potentialism, on the other" [16, p. 178].

To recapitulate, the P-Comp scheme is acceptable to the actualist (realm B), but not to the liberal potentialist (realm A). The conclusion, once again, is that the actualist can formally express strictly more properties of Turing-machine computability than the potentialist. Specifically, diverging behavior is no obstacle to the actualist but it is to the liberal potentialist, provided my readership accepts the analysis of Linnebo & Shapiro. That analysis, however, was construed independently of the present discussion and by two scholars who favor(ed) potential infinity to begin with.

References

[1] R.L. Constable and S.F. Smith. Computational foundations of basic recursive function theory. *Theoretical Computer Science*, 121:89–112, 1993.

[2] B.J. Copeland and O. Shagrir. Do accelerating Turing machines compute the uncomputable? 21(2), 2011.

[3] A.E. Curtis-Trudel. Why do we need a theory of implementation? 2020. Forthcoming in BJPS.

[4] M. Davis. *Computability and Unsolvability*. McGraw-Hill, New York, USA, 1958.

[5] E.G. Daylight. *Turing Tales*. Lonely Scholar, 2016.

[6] E.G. Daylight. Addressing the Question "What is a Program Text?" via Turing Scholarship. 43(4):87–91, 2021.

[7] E.G. Daylight. The Halting Problem and Security's Language-Theoretic Approach: Praise and Criticism from a Technical Historian. *Computability*, 10(2):141–158, 2021.

[8] J. Floyd. Wittgenstein's Diagonal Argument: A Variation on Cantor and Turing. In P. Dybjer, S. Lindström, E. Palmgren, and B.G. Sundholm, editors, *Epistemology versus Ontology*, pages 25–44. Springer, 2012.

[9] J. Floyd. Turing on "Common Sense": Cambridge Resonances. In J. Floyd and A. Bokulich, editors, *Philosophical Explorations of the Legacy of Alan Turing*, pages 103–149. Springer, 2017.

[10] J. Floyd. *Wittgenstein's Philosophy of Mathematics*. Cambridge University Press, 2021.

[11] G Gherardi. Alan Turing and the Foundations of Computable Analysis. *The Bulletin of Symbolic Logic*, 17(3):394–430, September 2011.

[12] D. Harel. *The Science of Computing: Exploring the Nature and Power of Algorithms*. Addison-Wesley, 1987, 1989.

[13] P. Henry. Mathematical Machines. In H. Haken, A. Karlqvist, and U. Svedin, editors, *The Machine as Metaphor and Tool*. Springer-Verlag, 1993.

[14] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[15] S.C. Kleene. *Mathematical Logic*. John Wiley and Sons, Inc., 1967.

[16] Ø. Linnebo and S. Shapiro. Actual and Potential Infinity. *Noûs*, 53(1):160–191, 2019.

[17] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc, 1967.

[18] R. Monk. Russell. In R. Monk and F. Raphael, editors, *The Great Philosophers: From Socrates to Turing*, pages 307–348. Phoenix, 2000.

[19] C.A. Petri. Communication with automata. Technical report, Griffiss Air Force Base, New York, Technical Report RADC-TR-65-3777, Vol 1, Suppl. 1 (1966).

[20] G. Piccinini. *Physical Computation: A Mechanistic Account*. Oxford University Press, 2015.

[21] M. Rescorla. A theory of computational implementation. *Synthese*, 191:1277–1307, 2014.

[22] K. Schmidt. Dispelling the Mythology of Computational Artifacts. In K. Schmidt, editor, *Cooperative Work and Coordinative Practices*, pages 391–413. Springer-Verlag, London/New York, 2011.

[23] S. Shanker. *Wittgenstein's remarks on the foundations of AI*. Routledge, 1998.

[24] A.M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, 2nd series*, 42:230–265, 1936.

[25] A.M. Turing. On computable numbers, with an application to the Entscheidungsproblem. A correction. *Proceedings of the London Mathematical Society, 2nd series*, 43, 1937.

[26] J. Woodward. Mechanistic explanation: Its scope and limits. *Proceedings of the Aristotelian Society, Supplementary Volumes*, 87:39–65, 2013.

[27] A. A. Zenkin. Logic of Actual Infinity and G. Cantor's Diagonal Proof of the Uncountability of the Continuum. *Review of Modern Logic*, 9(3-4):27 – 82, 2004.

A What is a Turing Machine?

The philosophy of computing is a flourishing research field. In a recent survey article, entitled: "Why do we need a theory of implementation?" [3], Curtis-Trudel provides conceptual diversity with regard to the question: 'What is a Turing machine?' For example, he sketches contrasting positions formulated by Piccinini, Copeland & Shagrir, and Rescorla. Piccinini is quoted as follows:

> The tape [of the Turing machine] and processing device are explicitly defined as spatiotemporal components. [20, p. 119-120]

One school of thought indeed takes a Turing machine to be spatiotemporal (irrespective of Piccinini's own nuanced position on this topic). In this context, Curtis-Trudel aptly injects the "Turing-machine realism" [3, Sec. 6.1] of Copeland & Shagrir into his narrative. These two authors described (but did not *per se* advocate) an "extra ontological level .... with Turing machines 'having' **causal** features" [2, p. 234, my emphasis]. I mention in passing that I use the word 'causal' in compliance with Curtis-Trudel's "causal" and "causal-mechanical" terminology; that is, with the understanding that the notion of 'cause' is more basic than 'mechanism,' cf. [26, p. 45].

In stark contrast to the positions illustrated so far, another school of thought casts a Turing machine as an abstract model of a physical system. In Rescorla's words:

> To describe a physical system's computational activity, scientists typically offer a computational model, such as a Turing machine or a finite state machine. Computational models are abstract entities. They are not located in space or time, and they do not participate in causal interactions. [21, p. 1277]

In sum, there is no agreement among philosophers of what a Turing machine entails.

All aforementioned philosophers, along with the present author, seem to share a dualistic philosophical outlook on science and technology, distinguishing between non-causal (abstract) objects and causal objects. The latter need not be physical *per se* according to some scholars. In the present chapter a causal Turing machine is treated as a mathematical object.