Algorithmic Scientific Inference

Within Our Computable Expected Reality

John Case

Computer and Information Sciences Department
University of Delaware
Newark, DE 19716 USA

Abstract

It is argued that, scientific laws, including quantum mechanical ones, can be considered algorithmic, that the *expected* behavior of the world, if not its exact behavior, is algorithmic, that, then, communities of human scientists over time have algorithmic expected behavior.

Some sample theorems about the boundaries of algorithmic scientific inference are then presented and interpreted. There is some discussion about (but there are not presentations of) succinct machine self-reference proofs of these theorems and whether non-artifactual self-referential examples may exist in the world.

There is also a brief discussion regarding the possibility that the expected behavior of reality may be *in*feasibly computable.

Keywords: machine inductive inference, physics, philosophy of science

Contents

Email address: `case@cis.udel.edu` (John Case)

1. Scientific Laws

Below we describe in Section 1.1 how and why we model scientific laws in terms of algorithms, and, in Section 1.2, we provide important clarification with an example from *quantum mechanics*.

1.1. Modeling Scientific Laws

In the 1970s, I was motivated to work on the *Theory of Machine Inductive Inference*, Putnam and Gold [Put63, Gol67, Put75], thanks to the Blums' assertion [BB75, Page 125] just below.

> Consider the physicist who looks for a law to explain a growing body of physical data. His data consist of a set of pairs (x, y), where x describes a particular experiment, e.g., a high-energy physics experiment, and y describes the results obtained, e.g., the particles produced and their respective properties. The law he seeks is essentially an algorithm for computing the function $f(x) = y$.

Such an algorithm is a *predictive explanation*, Case & Smith [CS78, CS83]: if one has the good fortune to *have* such an algorithm, one can use it to predict the outcomes of the associated experiments.

Importantly, a predictive explanation must provide its predictions *algorithmically*! How else are we to get out the predictions — by magic? To be sure, in, say, physics, the laws are typically not written down including how to extract algorithmically the predictions. That is implicit and may, in some cases, be difficult. The techniques are essentially covered by computably axiomatizable mathematics, algorithmic numerical techniques, etc. Of course physicists rarely resort to axiom systems directly, but, when mathematics is formulated axiomatically, one always sees a computably decidable set of axioms. How else could formal proofs be checked, e.g., when they cite an axiom, — by magic? Of course with a formal system having a computably decidable set of axioms, the set of corresponding theorems forms a computably enumerable set.

1.2. A Quantum Mechanical Example

Here is the promised example chosen on purpose to be from quantum mechanics. Essentially from Case, et al, [CJNM94, Cas07]:

> x codes a particle diffraction experiment & $f(x)$ the resultant probable distribution or *interference pattern* on the other side of the diffraction grating. Quantum theory provides *deterministic, algorithmic* extraction of $f(x)$ from x. A program for f is, then, a *predictive* explanation or law for the set of such particle diffraction experiments.

The program/law in this case does *not* tell us deterministically where the particles go. It tells us instead, *deterministically, algorithmically*, their statistically *expected behavior*! In the case an interference pattern is generated from an experiment x where multiple particles are sent through a diffraction grating, it deterministically, algorithmically provides $f(x)$ which can be, then, a depiction of that interference pattern!

Again, for the reasons spelled out at the end of Section 1.1 just above, a *predictive explanation* must provide its predictions *algorithmically*!

2. Data Types

In this section we indicate in detail how, without loss of generality, we can and will treat the functions f such as those described above in Sections 1.1 and 1.2 just above.

A *countable set* is (by definition) one in 1-1 correspondence with (some $\subseteq$) $\mathbb{N} = \{0, 1, 2, \ldots\}$, the set of natural numbers.

My former student, Mark Fulk, [Ful85] argued that the set of distinguishable experiments *one can actually do and record* on a phenomenon is countable: lab manuals can and do contain only *finite* notations, strings, and images from a *finite* alphabet of symbols, including gray and color pixel values.

One does *not* record *measurements* such as *arbitrary infinite-precision* real numbers of volts.

Beautiful continuous-mathematics (featuring *un*countable sets such as that of the real numbers) is employed in physics many times to smooth out *feasibly* some much too complicated discrete reality, e.g., a giant cloud of electrons.

Interestingly, Maddy [Mad08] discusses the just prior paragraph, and provides a pointer, [ER06, Pages 290, 326], to cases where a continuous approximation to a discrete thermodynamic reality fails.

So, one of my working hypotheses is that reality is *discrete*. This is discussed further early in Section 3.1 below. Of course continuous mathematics is, in many cases, on a practical level, hard to replace.

In what follows, then, thanks to Gödel or code numbering an algorithmically circumscribed *countably* infinite set of experiments and outcomes for some phenomenon F, e.g., some well circumscribed particle diffraction phenomenon: we imagine coding (algorithmically) the set of experiments associated with F onto $\mathbb{N}$ and the possible outcomes into $\mathbb{N}$, and we let the function f (associated with phenomenon F) map any experiment on phenomenon F with code # x, into the code # y of the outcome of x on F: $f(x) = y$.

Hence, the *type* of our fs can and will be taken to be $\mathbb{N} \rightarrow \mathbb{N}$.

Also, since we seek *algorithmic* explanations for F, we can handle the cases only where f is also *computable*.

N.B. Our above discussion does not yet take into account error bounds on measurements, an important, crucial, practical consideration. For our approach, we can just consider that the code numbers of experiments and outcomes, *include* measurement error bounds.

3. Computability

In Section 3.1 just below is discussed my additional working hypothesis that the *expected* behavior of reality is *algorithmic*.

Then in Section 3.2 further below we explain what this has to do with human scientific endeavors.

Next, in Section 3.3, we consider objections based on apparent human creativity and free will.

3.1. Computability of Expected Reality

Researchers in the cellular automata approach to physics, e.g., [Fey82, Min82, FT82, Tof84, TM87, Mar84, SW86, Mac86, Tof77b, Tof77a, Vic84, Wol83, Svo86, FHP86, Has87], take seriously the idea that the universe, including space and time, may well be discrete. Here Feynman [Fey82] is crucial, and Minsky [Min82] lays out the ideas of Ed Fredkin on some of the different ways physical space could be discrete.[1]

In a discrete, random universe but with *computable probability distributions* for its expected behaviors (e.g., a discrete, quantum mechanical universe with such distributions — as, I believe, ours is), the *expected* behavior will still be computable. It essentially follows from [dMSS56, San71, Gil72, Gil77] that one can compile any algorithm r having access to a random oracle, *which oracle is subject to a computable distribution*, into a deterministic algorithm d_r computing, in a sense, the expected outputs of r.[2]

Another working hypothesis of mine is, then, that the universe, besides being discrete, is *algorithmic* as to its *expected* behaviors.

N.B. We humans may be too finite ever to figure out *completely* how to *compute* the associated expected behaviors. But that's just about human limitations.

3.2. Computable Expected Behavior of Science

We *humans* are *components* of the universe; hence, communities of scientists over time *must also have computable expected behavior*!

Herein, then, we'll *model scientists* (and communities thereof over time) as *algorithmic*. *Then* we can have theorems about the *boundaries* of the (expected) behavior of science!

---

[1] The author of the present paper with others investigated lattice computers (a kind of cellular automata) in, for example, [CRS01], but he was also motivated therein by potential modeling thereby of spatial cognition [CRS94].

[2] We don't know which, any, of the many provably disparate theoretical models of randomness (see, for example, [LV08, BST10]) corresponds to the randomness of, say, alpha-decay — and we'd like to know.

Just as a conservation assumption from physics provides boundaries on and insight into the physically possible, so too the computable expected behavior assumption on scientific inference provides boundaries on and insight into what's possible with scientific inference.

In [Cas07] I discuss related language learning examples for *cognitive science* (not treated herein).

I invite physicists to explore the consequences *for physics* of our universe having computable expected behaviors. I'd really like to see something come out of that.

3.3. Creativity and Free Will

First we discuss creativity.

In a world with only computable expected behaviors, what about human *creativity*? How does my *somewhat* mechanistic working hypothesis account for the [Cas92] unbidden images which occur to people and which lead to solutions of difficult problems and/or works of great beauty and significance for the human condition?

I argue [Cas92] that humans are mostly not *consciously* aware of the brain processes that invoke such insights; hence, we have the *illusion* they aren't algorithmically produced. Our conscious thoughts are the mere tip of an iceberg.

Post [Pos44] described as *creative* cases where algorithmic processes are *algorithmically* transcended. His examples generalize a bit the algorithmic process of Gödel [Göd86] essentially for transforming an algorithm for deciding a set of "consistent" axioms for an arithmetic into a corresponding Gödel sentence. Adding (trivially algorithmically) that sentence to the axiom set provides the transcendence.[3]

Next we discuss free will.

Libet, et al, [LGWP83] found that particular, experimentally detectable *un*conscious cerebral activity always strictly precedes *conscious* human experiences of *will*ing to do something. This is somewhat suspicious methinks re the existence of human *free* will.

Conway and Kochen interpret their (Strong) Free Will Theorem [CK06, CK09] to mean, if some human has free will (about setting the details of some quantum mechanics experiment), then so do some particles.

[3] [Cas07] briefly refutes the argument that Gödel's process falsifies mechanism.

They want to retain human free will, so they ascribe it to some particles too. Of course, at least in the case of particles, they mean by it only non-determinism.

I'm inclined to see *conscious* free will as another one of many human illusions. We *may* have some non-determinism, but our *expected* behavior does not.[4]

4. Machine Inductive Inference

Next we begin to describe a model of scientific inference.

$$(0, f(0)), \ldots, (t-1, f(t-1)) \quad \xrightarrow{\text{In}} \quad \mathbf{M} \quad \xrightarrow{\text{Out}} \quad p_t$$

Above M is an algorithmic device receiving f's data points $(t, f(t))$, for $t = 0, 1, \ldots$. N.B. For simplicity herein we'll restrict the order of presentation of data from f to be in *this* order (this matters in *some* cases).

M's output above, having seen the data sequence

$$f[t] \stackrel{\text{def}}{=} (0, f(0)), ..., (t-1, f(t-1)),$$

is p_t, *where* p_t is a program in some fixed, general programming system.[5] We write $M(f[t]) = p_t$. N.B. For simplicity herein we'll restrict outselves to the case where M on $f[t]$ does *not* go into an infinite loop never producing p_t (this matters in *some* cases).

Perhaps, if M is "clever" enough and f is associated with a phenomenon F that is not too hard to figure out, eventually, i.e., for suitably large ts, the p_ts may come usefully close to computing f. More on this topic, in Section 4.1 just below where we begin to discuss in more detail what can be meant by *successful* scientific inference.

Then, in Section 4.2, we provide *with interpretations* some sample theorems about scientific inference.[6] Near the end of Section 4.2, we segue into Section 4.3 which discusses machine self-reference techniques which can, many times, be used to provide *very succinct* proofs, relevantly herein, of results regarding scientific inference.

---

[4] We might have (in some sense) less non-determinism than mere particles since, for example, our DNA has some error-correcting capabilities.

[5] When $t = 0$, $f[t]$ is the empty sequence.

[6] [Cas07] provides additional examples.

Lastly, in Section 4.4, is discussed, whether the self-referential examples employed might actually correspond to (non-artifactual) examples in the real world.

4.1. Criteria of Success

Definition 1 (Success Criteria $\mathbf{Ex}^a$)**.** Suppose $a \in (\mathbb{N} \cup \{*\})$. Suppose $\mathcal{S}$ is a class of computable functions f. '**Ex**' stands for 'Explanatory.' a stands for $a$nomaly.

$\mathcal{S} \in \mathbf{Ex}^a$ iff there is a suitably clever M so that, for *every* $f \in \mathcal{S}$, for *some* associated t, $M(f[t]) = M(f[t+1]) = \cdots$ *and* $M(f[t])$ *computes* f — *except* at up to a data points. Here, up to $*$ points means up to finitely many.

Informally, M witnesses that $\mathcal{S} \in \mathbf{Ex}^a$ *means*, on any $f \in \mathcal{S}$, M's output programs on f, *eventually* settle down syntactically to a single program "for" f which program has *at most* a anomalous predictions re values of f.

In science, we don't know when (if ever) we begin to have predictive explanations that are pretty good; we don't know t's value in the above Definition.

Definition 2 (Success Criteria $\mathbf{Bc}^a$)**.** '**Bc**' stands for 'Behaviorally correct.'

$\mathcal{S} \in \mathbf{Bc}^a$ iff, for some M, for *every* $f \in \mathcal{S}$, for *some* associated t, programs $M(f[t]), M(f[t+1]), \cdots$ *each* computes f — each except at up to a data points.

For these $\mathbf{Bc}^a$ criteria, the programs $M(f[t]), M(f[t+1]), \cdots$ *can* be (syntactically) quite different from one another.

For the criteria $\mathbf{Ex}^a$ and $\mathbf{Bc}^a$, my original motivation for the importance of small values of a, i.e., a few anomalies being tolerated in final predictive explanations, came from *anomalous dispersion*: the *classical* explanation for the degree of bending of "light" passing through a prism, fails for the X-ray case, an anomalous case.

4.2. Sample Theorems

Theorem 3 (Gold & Blums [Gol67, BB75])**.** The class of polynomial time computable functions $\in \mathbf{Ex}^0$.

Theorem 4 (See [CS78, CS83])**.** $\mathbf{Ex}^0 \subset \mathbf{Ex}^1 \subset \cdots \subset \mathbf{Ex}^* \subset \mathbf{Bc}^0 \subset \mathbf{Bc}^1 \subset \cdots \subset \mathbf{Bc}^*$, where $\subset$ is *proper* subset.

Hence, tolerating anomalies *strictly* increases inferring power as does relaxing the restriction of (syntactic) convergence to single programs.

Physicists' use of slightly faulty explanations is vindicated!

The anomalies that *must* be exploited to prove the $\mathbf{Ex}^a$-hierarchy above are anomalies of *omission* or *in*completeness: the predictive explanations' errors are where they loop infinitely with *no* prediction [CS78, CS83].

Hence, thanks to the unsolvability of the Halting Problem [Rog67], Popper's Refutability Principle [Pop68] is violated in a way Popper didn't consider [CS78, CS83]!

We next present some very interesting restricted versions of $\mathbf{Ex}^0$.

Definition 5 (Postdictive Completeness [Bār74, BB75, Wie76, Wie78])**.** $\mathcal{S} \in$ **PdCompEx** iff, some M witnesses that $\mathcal{S} \in \mathbf{Ex}^0$ *and*, for every $f \in \mathcal{S}$, for every t, for *each* $s < t$, the I/O behavior of program $M(f[t])$ *on input* s must agree with f *on input* s.

PdCompEx provides a strong common sense constraint on $\mathbf{Ex}^0$: a scientist should always hypothesize a program which at least *postdicts* his known data.

Definition 6 (Postdictive Consistency [Wie76, Wie78, Cas07])**.** $\mathcal{S} \in$ **PdConsEx** iff, some M witnesses that $\mathcal{S} \in \mathbf{Ex}^0$ *and*, for every $f \in \mathcal{S}$, for every t, for *each* $s < t$, *either* the I/O behavior of program $M(f[t])$ *on input* s must agree with f *on input* s *or* program $M(f[t])$ *on input* s loops infinitely.

PdConsEx provides a weaker common sense constraint on $\mathbf{Ex}^0$: a scientist should never conjecture an hypothesis which makes an *explicit* prediction contradicting his known data.

Theorem 7 ([Bār74, BB75, Wie76, Wie78, CJSW04, Cas07])**.**

$$\mathbf{PdCompEx} \subset \mathbf{PdConsEx} \subset \mathbf{Ex}^0!$$

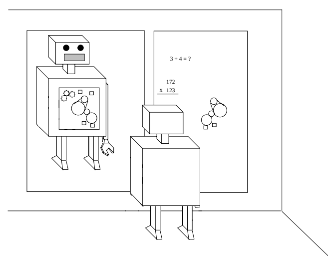
Hence, *surprisingly*, for example, judiciously employing hypotheses *explicitly contradicting known data* can *strictly enhance* inferring power!

For example, it can be shown by a machine self-reference argument [Rog67, Kleene's Recursion Theorem, Page 214] that the *class* of all computable f *with finite range* and *where* max(range(f)) codes a program for f is $\in$ ($\mathbf{Ex}^0$ − **PdConsEx**) [Wie76, Wie78, CJSW04, Cas07].

To show this self-referential class $\in \mathbf{Ex}^0$ is *straightforward*: have M always output the program coded by the largest number it's seen so far in the range of f. This makes the proof of the positive half extremely short.

To show this class $\notin$ **PdConsEx** succinctly employs machine self-reference mixed with so-called diagonalization [Rog67].[7]

4.3. Self-Reference Techniques



The robot above has a transparent front through which its complete program (flowchart, wiring diagram, ...) can be seen. It stands in front of a mirror and a writing board, so it can copy its complete program on the board for use as data in its computations.

A simplest case, then, of *machine self-reference* involves a program (like the robot's above) which makes a copy of itself to use as data. It, then, has usable, perfect *self-knowledge*!

The robot shown uses a mirror to make its self-copy. Simple self-replication works in the general case [Cas74, Cas94].[8]

Machine self-reference can involve many programs, *including infinitely*

[7][Cas07] contains a proof of a related result which proof also essentially works for this result.

[8]It can be *correctly* argued that the No Cloning Theorem of Quantum Mechanics [WZ82, Die82] implies that self-copies perfect down to quantum state can not be made. Fortunately, self-copies at usefully higher levels than the quantum states level *can* be created so as to persist long enough for human endeavors. Crucial example: computer bit settings *can* be copied with the copies' bits stably persisting for very long time intervals and with corresponding negligible probability of degradation during those very long intervals.

many programs each making a self-copy for data-use by all of them [Cas74, Cas94]!

Consider the class $\mathcal{S}_{\mathbf{Bc}}$ of all computable f such that all but finitely many numbers in the sequence of f's successive values, $f(0), f(1), f(2), \ldots$, code programs for f.

It is straightforward to see that $\mathcal{S}_{\mathbf{Bc}} \in \mathbf{Bc}^0$: have M successively output the programs coded by the succession of values of f, $f(0), f(1), f(2), \ldots$.

When I was co-creating [CS78], I had the *intuition* that $\mathcal{S}_{\mathbf{Bc}}$ captured the *essence* of $\mathbf{Bc}^0$.

In particular I thought that if *any* class would be in $(\mathbf{Bc}^0 - \mathbf{Ex}^*)$, $\mathcal{S}_{\mathbf{Bc}}$ would be. I showed with Harrington [CS78, CS83], by an infinitary machine self-reference argument, that, in fact, $\mathcal{S}_{\mathbf{Bc}} \notin \mathbf{Ex}^*$.

We can now formally *define* in strong senses what it means for a class to capture the essence of a success criterion and can *prove* for $\mathcal{S}_{\mathbf{Bc}}$ that, in one sense, it *does* (although, newer, related, so-called *self-learning* classes, capture in a stronger sense). See Case and Kötzing [CK11a] (see also [CK08], and, for related work re language learning, see [CK10b, CK10a, CK11b]).

We can now formally *define*, in a strong sense, what it means for a class to capture the essence of a success criterion and can *prove* for self-referential classes like $\mathcal{S}_{\mathbf{Bc}}$ it *does*. See Case and Kötzing [CK10b] for preliminary work.

4.4. Self-Reference in Reality

Generally, machine self-reference proofs for theorems like the above are more succinct than alternative proof techniques. I like them.

Interesting work exists on whether separation results from Section 4.2 above hold if one "destroys" the self-reference tricks [Zeu86, KS89, Ful90, CJO$^+$00, CJSW04, JSW01, Jai99, OS02] We'll not pursue this further herein.

Instead, our interest herein is whether self-referential examples entail the existence of (non-artifactual) real world witnessing examples.[9]

Case [Cas86] notes that in some views of the world it is a network with parts reflecting on the whole. That resembles multiple machine self-reference. Human social cognition is an imperfect such network.

Case [Cas99] argues that a machine self-reference argument is such a *simple* reason for a truth, the "space" of reasons for its truth may be broad enough to admit natural examples.

[9] One could, in principle, build *artifactual* black box devices which work (and could be inductively inferred from their behavior) like the members of $\mathcal{S}_{\mathbf{Bc}}$ above.

Also noted therein is that, empirically, while Gödel [Göd86] proved his famous first incompleteness theorem by a (linguistic) self-reference argument[10], later researchers [PH77, Sim85, Sim87] found quite natural examples of incompleteness.

I think machine self-reference proofs for the existence of situations are harbingers of natural examples witnessing the same situations.

5. Infeasibly Computable Expected Reality

In modern computer science there is a sensible, *practical* focus on a special case of *computable*, namely, *feasibly* computable.

Originally this meant polynomial-time computable (in the length of input on multi-tape Turing machines) [Cob64], but recently it can mean $\mathcal{BQP}$-computability, a *quantum-parallelism* version of polynomial-time computability [BV97].

Does the universe have *some non*-artifactual expected behaviors which, while computable, are *in*feasibly so? An *artifactual* computer with access to increasing memory resources and which employs any algorithm for deciding Presburger arithmetic would, in principle, have *in*feasible expected behavior [FR74].

Fix a meaning for *feasibly computable*. Humans (including human scientists) don't currently know how to compute *feasibly* the *in*feasibly computable. I conjecture that our massively parallel processing brains are sometimes good for handling largish constants in front of whatever run-time bounds.[11]

[10] It can also be proved by a *machine* self-reference argument.

[11] So-called *hypercomputation* typically involves allowing infinitely many (say, Turing machine) computation steps in finite time (see, for example, [Kre65, Kre74, Rog67, Dav04, Dav06a, Dav06b, Dav06c, Dow10, AMN$^+$10]. A recursive iteration of the idea leads to Kreisel's $\aleph_0$-mind computability (characterizing the Π^1_1-computable partial functions); $\aleph_0$-mind computability permits deciding first order arthmetic, but not second order arithmetic (see Rogers [Rog67] for definitions, results, and very nice discussion).

The big question is whether, in our physical universe (which happens to currently include human brains), such computations are actually executable. I, of course, believe they are not — although I wish they were. Margolus [Mar95, Cas07] pointed out to me that such computations would require infinitely much energy, which is ostensibly not available to apply anywhere locally in our universe. Wonderful tricks possibly to achieve real world hypercomputations which exploit temporal differences inside and outside black holes (see, for example, brief mention and references in [Dow10]) won't work either — since, thanks

Feynman [Fey82] was the first to remark that some of our (humanly produced, predictive) explanations for quantum phenomena and run on (deterministic) parallel processing computers produce answers significantly more slowly than nature itself produces those answers! This led to the beginning of thinking of possible speed-ups with quantum parallelism.[12]

I expect any *humanly-produced and employed* acceptable programming systems[13] are quite unlike the (unknown) acceptable system of non-artifactual physical reality.

References

[AMN+10] H. Andréeka, J. Madarász, I. Németi, P. Németi, and G. Székely. Axiomatization of relativistic physics in a logical framework. In *Proceedings of the 3rd International Workshop on Physics and Computation*, pages 72–74, September 2010.

[Bār74] J. Bārzdiņš. Two theorems on the limiting synthesis of functions. *In Theory of Algorithms and Programs, Latvian State University, Riga*, 210:82–88, 1974.

[BB75] L. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

[BST10] L. Bienvenu, F. Stephan, and J. Teutsch. How powerful are integervalued martingales? In *Sixth Conference of Computability in Europe (CiE 2010), Proceedings*, volume 6158 of *Lecture Notes in Computer Science*, pages 59–68. Springer-Verlag, Berlin, 2010.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM Journal on Computing*, 26:1411–1473, 1997.

to Hawking Radiation [Haw74], a black hole completely evaporates in *finite* time.

[12] Kemeny [Kem59] provides a wonderful, informal (self-reference) argument for cases where one cannot predict reality *before* that reality happens.

[13] The acceptable systems are essentially the natural, general purpose systems for computing all the partial-computable functions Mathematically, they are those in which any other system can be interpreted [Rog58, Rog67, MY78, Ric80, Ric81, Roy87]. An example humanly producible and employable one would be C++ implemented on some deterministic parallel processing computer, which computer permits dynamic memory extensions.

[Cas74] J. Case. Periodicity in generations of automata. *Mathematical Systems Theory*, 8:15–32, 1974.

[Cas86] J. Case. Learning machines. In W. Demopoulos and A. Marras, editors, *Language Learning and Concept Acquisition*. Ablex Publishing Company, 1986.

[Cas92] J. Case. Turing machine. In Stuart Shapiro, editor, *Encyclopedia of Artificial Intelligence*. John Wiley and Sons, New York, NY, second edition, 1992.

[Cas94] J. Case. Infinitary self-reference in learning theory. *Journal of Experimental and Theoretical Artificial Intelligence*, 6:3–16, 1994.

[Cas99] J. Case. The power of vacillation in language learning. *SIAM Journal on Computing*, 28(6):1941–1969, 1999.

[Cas07] J. Case. Directions for computability theory beyond pure mathematical. In D. Gabbay, S. Goncharov, and M. Zakharyaschev, editors, *Mathematical Problems from Applied Logic II. New Logics for the XXIst Century*, International Mathematical Series, Vol. 5, pages 53–98. Springer, 2007. Invited book chapter.

[CJNM94] J. Case, S. Jain, and S. Ngo Manguelle. Refinements of inductive inference by Popperian and reliable machines. *Kybernetika*, 30:23–52, 1994.

[CJO$^+$00] J. Case, S. Jain, M. Ott, A. Sharma, and F. Stephan. Robust learning aided by context. *Journal of Computer and System Sciences*, 60:234–257, 2000. Special Issue for *COLT'98*.

[CJSW04] J. Case, S. Jain, F. Stephan, and R. Wiehagen. Robust learning – rich and poor. *Journal of Computer and System Sciences*, 69:123–165, 2004.

[CK06] J. Conway and S. Kochen. The free will theorem. *Foundations of Physics*, 17:59–89, 2006.

[CK08] J. Case and T. Kötzing. Dynamically delayed postdictive completeness and consistency in learning. In *19th International Conference on Algorithmic Learning Theory (ALT'08)*, volume

5254 of *Lecture Notes in Artificial Intelligence*, pages 389–403. Springer, 2008.

[CK09] J. Conway and S. Kochen. The strong free will theorem. *Notices of the AMS*, 56:226–232, 2009.

[CK10a] J. Case and T. Kötzing. Solutions to open questions for non-U-shaped learning with memory limitations. In M. Hutter et al, editor, *21st International Conference on Algorithmic Learning Theory (ALT'10)*, volume 6331 of *Lecture Notes in Artificial Intelligence*, pages 285–299, 2010. Invited for submission to the associated Special Issue of *TCS*.

[CK10b] J. Case and T. Kötzing. Strongly non U-shaped learning results by general techniques. In *Proceedings of the 23rd Annual Conference on Learning Theory (COLT'10)*. Omnipress, 2010. www.colt2010.org/papers/COLT2010proceedings.pdf is the Proceedings.

[CK11a] J. Case and T. Kötzing. Measuring learning complexity with criteria epitomizers. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS'11)*, 2011. To appear.

[CK11b] J. Case and T. Kötzing. Memory-limited non-U-shaped learning: Solving problems by general techniques, 2011. Invited for submission and submitted to the associated Special Issue of *TCS* for *ALT'10*, but also based on our paper [CK10b] in *COLT'10*.

[Cob64] A. Cobham. The intrinsic computational difficulty of functions. *Proceedings of the 1964 International Congress for Logic, Methodology, and the Philosophy of Science*, pages 24–130, 1964.

[CRS94] J. Case, D. Rajan, and A. Shende. Representing the spatial/kinematic domain and lattice computers. *Journal of Experimental and Theoretical Artificial Intelligence*, 6:17–40, 1994.

[CRS01] J. Case, D. Rajan, and A. Shende. Lattice computers for approximating euclidean space. *Journal of the ACM*, 48(1):110–144, 2001.

[CS78] J. Case and C. Smith. Anomaly hierarchies of mechanized inductive inference. In *Symposium on the Theory of Computation*, pages 314–319, 1978.

[CS83] J. Case and C. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.

[Dav04] M. Davis. The myth of hypercomputation. In C. Teuscher, editor, *Alan Turing: Life and Legacy of a Great Thinker*, pages 195–212. Springer, 2004.

[Dav06a] M. Davis. Computability, computation and the real world. In S. Termini, editor, *Imagination and Rigor : Essays on Eduardo R. Caieniello's Scientific Heritage*, pages 63–70. Springer-Verlag Italia, 2006.

[Dav06b] M. Davis. Why there is no such subject as hypercomputation. *Applied Mathematics and Computation*, 2006. Special Issue on Hypercomputation; to appear.

[Dav06c] M. Davis. The Church-Turing thesis: Consensus and opposition. In *Proceedings CiE 2006*, Springer Notes on Computer Science, Swansee, July 2006.

[Die82] D. Dieks. Communication by EPR devices. *Physics Letters A*, 92(6):271–272, 1982.

[dMSS56] K. deLeeuw, E. Moore, C. Shannon, and N. Shapiro. Computability by probabilistic machines. *Automata Studies, Annals of Math. Studies*, 34:183–212, 1956.

[Dow10] G. Dowek. The physical Church thesis as an explanation of the Galileo thesis. In *Proceedings of the 3rd International Workshop on Physics and Computation*, pages 26–33, September 2010.

[ER06] T. Engel and P. Reid. *Thermodynamics, Statistical Thermodynamics, and Kinetics*. Pearson-Benjamin-Cumming, San Francisco, CS, 2006.

[Fey82] R. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7), 1982.

[FHP86] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier Stokes equation. *Physical Review Letters*, 56(14):1505–1508, April 1986.

[FR74] M. Fischer and M. Rabin. Super-exponential complexity of presburger arithmetic. In *Proceedings of the SIAM-AMS Symposium in Applied Mathematics*, volume 7, pages 27–41, 1974.

[FT82] E. Fredkin and T. Toffoli. Conservative logic. *International Journal of Theoretical Physics*, 21(3/4), 1982.

[Ful85] M. Fulk. *A Study of Inductive Inference Machines*. PhD thesis, SUNY at Buffalo, 1985.

[Ful90] M. Fulk. Robust separations in inductive inference. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, pages 405–410, St. Louis, Missouri 1990.

[Gil72] J. Gill. *Probabilistic Turing Machines and Complexity of Computation*. PhD thesis, University of California, Berkeley, 1972.

[Gil77] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6:675–695, 1977.

[Göd86] K. Gödel. On formally undecidable propositions of Principia Mathematica and related systems I. In S. Feferman, editor, *Kurt Gödel. Collected Works. Vol. I*, pages 145–195. Oxford Univ. Press, 1986.

[Gol67] E. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[Has87] B. Hasslacher. Discrete fluids. *Los Alamos Science*, (15):175–217, 1987. Special Issue.

[Haw74] S. Hawking. Black hole explosions? *Nature*, 238:30, 1974.

[Jai99] S. Jain. Robust behaviorally correct learning. *Information and Computation*, 153(2):238–248, September 1999.

[JSW01] S. Jain, C. Smith, and R. Wiehagen. Robust learning is rich. *Journal of Computer and System Sciences*, 62(1):178–212, 2001.

[Kem59] J. Kemeny. *Philosopher Looks at Science*. Van Nostrand Reinhold/co Wiley, 1959.

[Kre65] G. Kreisel. Mathematical logic. In T.L. Saaty, editor, *Lectures in Modern Mathematics III*, pages 95–195. J. Wiley and Sons, New York, 1965.

[Kre74] G. Kreisel. A notion of mechanistic theory. *International Journal of Theoretical Physics*, 29:11–26, 1974.

[KS89] S. A. Kurtz and C. Smith. On the role of search for learning. In R. Rivest, D. Haussler, and M. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory, Santa Cruz, California*, pages 303–311. Morgan Kaufmann Publishers, Inc., 1989.

[LGWP83] B. Libet, C. Gleason, E. Wright, and D. Pearl. Time of conscious intention to act in relation to onset of cerebral activity (readiness-potential). The unconscious initiation of a freely voluntary act. *Brain*, 106:623–642, 1983.

[LV08] M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, third edition, 2008.

[Mac86] Thinking Machines. Introduction to data level parallelism. Technical Report 86.14, Thinking Machines, April 1986.

[Mad08] P. Maddy. How applied mathematics became pure. *The Review of Symbolic Logic*, 1(1):16–41, 2008.

[Mar84] N. Margolus. Physics–like models of computation. *Physica 10D*, pages 81–95, 1984.

[Mar95] N. Margulus, 1995. Private communication.

[Min82] M. Minsky. Cellular vacuum. *International Journal of Theoretical Physics*, 21(6/7), 1982.

[MY78] M. Machtey and P. Young. *An Introduction to the General Theory of Algorithms*. North Holland, New York, 1978.

[OS02] M. Ott and F. Stephan. Avoiding coding tricks by hyperrobust learning. *Theoretical Computer Science*, 284(1):161–180, 2002.

[PH77] J. Paris and L. Harrington. A mathematical incompleteness in Peano arithmetic. In J. Barwise, editor, *Handbook of Mathematical Logic*. North Holland, 1977.

[Pop68] K. Popper. *The Logic of Scientific Discovery*. Harper Torch Books, New York, second edition, 1968.

[Pos44] E. Post. Recursively enumerable sets of positive integers and their decision problems. *Bulletin of the American Mathematical Society*, 50:284–316, 1944.

[Put63] H. Putnam. Probability and confirmation. *Voice of America, Forum on Philosophy of Science*, 10, 1963. Reprinted as [Put75].

[Put75] H. Putnam. Probability and confirmation. In *Mathematics, Matter, and Method*. Cambridge University Press, 1975.

[Ric80] G. Riccardi. *The Independence of Control Structures in Abstract Programming Systems*. PhD thesis, SUNY Buffalo, 1980.

[Ric81] G. Riccardi. The independence of control structures in abstract programming systems. *Journal of Computer and System Sciences*, 22:107–143, 1981.

[Rog58] H. Rogers. Gödel numberings of partial recursive functions. *Journal of Symbolic Logic*, 23:331–341, 1958.

[Rog67] H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw Hill, New York, 1967. Reprinted, MIT Press, 1987.

[Roy87] J. Royer. *A Connotational Theory of Program Structure*. Lecture Notes in Computer Science 273. Springer-Verlag, 1987.

[San71] E. Santos. Computability by probabilistic Turing machines. *Transactions of the AMS*, 159:165–184, 1971.

[Sim85] S. Simpson. Nonprovability of certain combinatorial properties of finite trees. In L. Harrington, M. Morley, A. Schedrov, and S. Simpson, editors, *Harvey Friedman's Research on the Foundations of Mathematics*, pages 87–117. North Holland, 1985.

[Sim87] S. Simpson. Unprovable theorems and fast–growing functions. In S. Simpson, editor, *Logic and Combinatorics*, AMS Contemporary Mathematics, pages 359–394. American Mathematical Society, 1987.

[Svo86] K. Svozil. Are quantum fields cellular automata? *Physics Letters A*, 119(4), December 1986.

[SW86] J. Salem and S. Wolfram. Thermodynamics and hydrodynamics with cellular automata. In S. Wolfram, editor, *Theory and Applications of Cellular Automata*. World Scientific, 1986.

[TM87] T. Toffoli and N. Margolus. *Cellular Automata Machines*. MIT Press, 1987.

[Tof77a] T. Toffoli. Cellular automata machines. Technical Report 208, Comp. Comm. Sci. Dept., University of Michigan, 1977.

[Tof77b] T. Toffoli. Computation and construction universality of reversible cellular automata. *Journal of Computer and System Sciences*, 15:213–231, 1977.

[Tof84] T. Toffoli. CAM: A high–performance cellular–automaton machine. *Physica 10D*, pages 195–204, 1984.

[Vic84] G. Vichniac. Simulating physics with cellular automata. *Physica 10D*, pages 96–116, 1984.

[Wie76] R. Wiehagen. Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Elektronische Informationverarbeitung und Kybernetik*, 12:93–99, 1976.

[Wie78] R. Wiehagen. *Zur Theorie der Algorithmischen Erkennung*. PhD thesis, Humboldt University of Berlin, 1978.

[Wol83] S. Wolfram. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 55(3):601–644, July 1983.

[WZ82] W. Wootters and W. Zurek. A single quantum cannot be cloned. *Nature*, 299:802–803, 1982.

[Zeu86] T. Zeugmann. On Bārzdiņš' conjecture. In K. P. Jantke, editor, *Analogical and Inductive Inference, Proceedings of the International Workshop*, volume 265 of *Lecture Notes in Computer Science*, pages 220–227. Springer-Verlag, 1986.