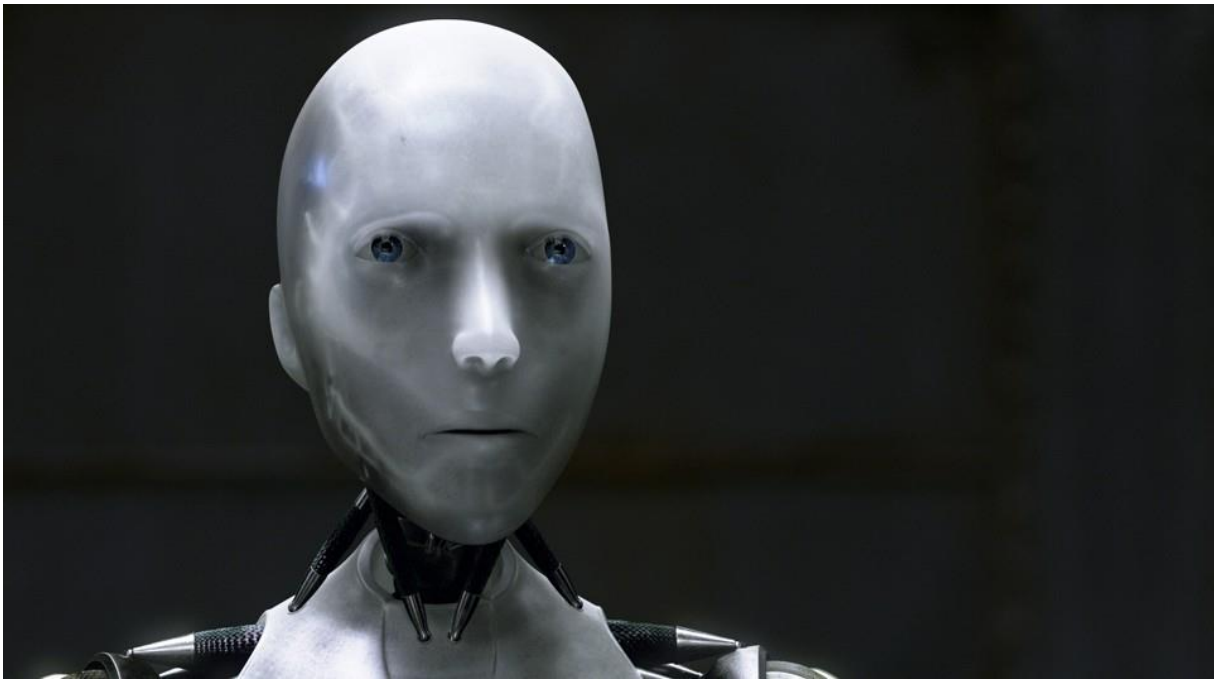
Artificial, But Not Intelligent: A Critical Analysis of AI and AGI

Andreas Keller



A still frame from "I, Robot" (2004, dir. Alex Proyas)

Recently, a lot of attention and hype has been given to discussions about the alleged dangers of artificial general intelligence. These fears have been fueled further by the release of the ChatGPT in November 2022. The fear is that an artificial general intelligence system could take over, resulting in the lock-in of bad values and leading to dystopian results. How realistic are these fears? In the following, I will discuss what artificial intelligence is and what artificial general intelligence would be. A clear concept of this is missing from the recent discussion, leading to confusions. Insights from computability theory (that are actually decades old) can shine some light on these questions.[i]

Before discussing the topic of "AI" (artificial intelligence) and "AGI" (artificial general intelligence), let me first introduce a few concepts. These concepts are not my invention, they are standard mathematical concepts. I am introducing these concepts here in an informal way to keep the discussion simple. Based on these concepts, I am sketching, very loosely and informally, a mathematical proof. A more rigorous formulation can be found in (Rogers, 1987), but see also (Ammon, 2013) and (Ammon, 2016).[ii]

A total function is a function from the set of natural numbers into a non-empty subset of the set of natural numbers. It assigns a natural number as its function value (or

"output") to every natural number (or "input"). The term "total" here means that there is a value for every natural number, i.e., there is no input value where the function is not defined. A total function is computable if there is a program (i.e., a Turing machine or algorithm) that computes it.

It is possible to represent any chunk of digital data as a sequences (strings) of signs (characters). It is further possible to define computable functions (with computable inverse functions) that assign a unique natural number to each sequence of signs. Such functions are called Gödel numberings and the number assigned to a specific sequence of signs this way is called its Gödel number.

Any program that halts for all its inputs and that maps some data to other data can be replaced by a program that maps the Gödel numbers of the Inputs to the Gödel numbers of the outputs. It is thus possible to replace computations on arbitrary kinds of data with equivalent computations on natural numbers. In this way, arbitrary halting programs can be replaced by equivalent programs for computable total functions on natural numbers.

A set of programs[iii] P is Turing-enumerable if a program p can be written that takes natural numbers as its input and produces all the programs in the given set as its outputs. This means that for each program p in the given set of algorithms P, a natural number n must exist so that applying the enumeration program to this number yields the given program as its output, i.e., p = P(n). In the following, I will use the notation P_n for P(n), i.e., P_n denotes the Program that is calculated as the output of P for the input n.

Now it can be shown with a simple proof that the set of all computable total functions is not Turing-enumerable: Assume that a program P exists that enumerates the set of all computable total functions. Now we construct a function f as $f(n) = P_n(n) + 1$. So, to calculate this function for an input n, we use P (a Turing-computable algorithm) to calculate the program P_n, then apply P_n to n and then add 1 to the result.[iv] Clearly, f is a computable total function. According to our assumption, P is complete with respect to the set of all computable total functions. So, there must be a natural number m so that $P_m = f$. Since $P_m = f$, applying f to m yields $f(m) = P_m(m)$. But according to the definition of f, we know already that $f(m) = P_m(m) + 1$. So, we get $P_m(m) = P_m(m) + 1$. Subtracting $P_m(m)$ on both sides of this equation yields $0 = 1$, which is wrong. So, the assumption that an enumeration P exists that produces all computable total functions leads to a wrong statement and is therefore wrong. This means that the set of all computable total functions is not Turing-enumerable.[v]

Moreover, the proof informally sketched here shows that for an enumeration of computable total functions F, it is always possible to *effectively construct* a new function f not generated by F. This construction (taking the diagonal and modifying it, for example by adding 1) is an example of what is known in mathematics as a productive

function.[vi] If we have a set X that is not Turing-enumerable and we have a subset Y of X that is Turing-enumerable, a productive function can generate from Y a new element x of X that is not contained in Y. Such a set X is called a productive set.

Obviously, if we start with any enumeration algorithm P, it is not possible to get a complete enumeration by building a new enumeration algorithm P' that somehow contains the productive function and can apply it and become complete that way. We can, of course, build the productive function into P and get some P' that way and P' might produce a larger set than P, but it will be incomplete again because the productive function can be applied to it as well *from the outside*. The proof means that it is impossible to construct a complete algorithm to enumerate all computable total functions.[vii]

Now let's apply this insight to AI.[viii] AI-systems are algorithms (in AI jargon, they are often called "models", but these are just – rather convoluted – algorithms). These algorithms are generated by learning algorithms that are applied to sets of training data (e.g., example data like input-output pairs). The learning algorithm constructs algorithms compatible with the sample data in its input. The learning algorithm might contain steps that involve random data. We can consider this random data to be part of the input data.

We can construct a Gödel numbering that maps the input data and the output data of the generated algorithms on natural numbers (if the resulting algorithms, or "models", make use of random data, this random data can also be viewed as being part of the input and would be included in the generation of the Gödel numbers). Likewise, we can map the training data (the sets of input-output pairs) onto Gödel numbers, so that the complete training data used for a training session of a learning algorithm is mapped onto a number.[ix] In this way, we can replace the algorithms produced by the learning algorithm by computable total functions of natural numbers. The learning algorithm is thus viewed as an algorithm that maps natural numbers onto algorithms for computable total functions. In other words, the learning algorithm is an enumeration algorithm producing a Turing-enumerable subset of the set of computable total functions!

The proof that every computable enumeration of algorithms for computable total functions is incomplete then means that for each learning algorithm, there are learning tasks that it cannot master. Each such algorithm is special! It has systematic blind spots.

We can view the learning process as a process of discovering regularity.[x] The proof then means that for each learning algorithm, there are instances of regularities in data that it cannot discover, while a different learning algorithm might be able to discover that instance of regularity (but will in turn have its own blind spots), and so on. So, all AI learning algorithms are special. They cannot be general.

More specifically, not only can a computable total function be constructed whose regularity a particular learning algorithm cannot learn, but an algorithm for the computation of that function can be constructed *from the learning algorithm itself*![xi] Another learning algorithm can be constructed that covers the new function, but that extended learning algorithm will have a blind spot of its own, and so on.

So, all learning algorithms are incomplete. As a consequence, *the concept of learning in its general form cannot be formalized*. Every formal theory formalizing the concept of learning (equivalent to an algorithm for the enumeration of a set of (learnt) algorithms) is incomplete in principle. We can define general intelligence as the ability to discover arbitrary (computable[xii]) regularities in arbitrary data. At least, that ability should be part of something worthy of being called "general intelligence". The proof then means that general intelligence (general cognition) is impossible if we are restricted to algorithms. Instead, general intelligence is required to go beyond Turing machines.

Another result from theoretical computer science leads to the same insight. This is the result that the Kolmogorov complexity is not Turing computable.[xiii] For a given programming language (e.g., Turing machines), the Kolmogorov complexity of a given piece of data is the size (length) of the smallest program whose output is that data. We can think of lossless data compression as the construction of a program smaller than the original data whose execution yields that data as an output. If the data contains any regularity (for example, repetitions), it is possible to compress or "fold" it "along" that regularity, i.e., write a program that produces the regular structure as its output (for example with a program loop in case of the repetition-example). It can be shown that the Kolmogorov complexity is not, in the general case, Turing-computable. As a result, if we write a compression algorithm that compresses data into shorter algorithms producing that data as output, it is always possible that a given compression algorithm does not yield the optimal compression in some cases. If we had a compression algorithm that was optimal in all cases, we could simply compute the Kolmogorov complexity by applying that algorithm and then checking the length of the resulting programs. Since this is not possible, optimal compression by algorithms is impossible. Each compression algorithm has systematic blind spots, i.e., there will be cases of regularities in data that it will be unable to spot and exploit.

Learning can be viewed as data compression: a computable total function f generates a sequence of outputs f(1), f(2)…. For an algorithm computing f, there is some number n where the sequence f(1) … f(n) is longer than that algorithm computing f. From then on at the latest, the sequence must contain some regularity since the algorithm computing f can be viewed as a compression of the sequence. Constructing the algorithm from examples can this way be viewed as a process of data compression. The result about the Kolmogorov complexity is another way to see that it is impossible to construct all such algorithms (excluding algorithms that don't halt for all of their inputs) with a single learning/compression algorithm.

The impressive achievements of learning algorithms are based on regularities in the data. We can expect learning algorithms to construct very efficient and powerful algorithms for large sets of problems which contain some regularity and can either be covered by an efficient algorithm or for which at least practically useful approximations are possible. However, we cannot expect this type of "AI" to develop into general intelligence because each instance is necessarily special and has blind spots.

Humans are able to find solutions for tasks for which it can be shown that they are not Turing computable. For example, human beings can do mathematics and do computer programming, tasks that involve activities for which no general algorithm exists. This indicates that the capabilities of human beings go beyond those of algorithms. In physics, entities are known for which some questions are not computable.[xiv] Such systems behave according to some laws, typically sets of equations, for which no algorithm for their solution in all cases exists. So, physical systems are not bound to the restrictions of computability. For such physical entities, special cases or approximations might be computable, but in the general case, all our methods of computation for such systems will always be incomplete (although they might be extensible). Human beings seem to be such entities for which complete computability is not possible. Any single algorithmic or formal theory (exact description) of such entities is incomplete. It might be possible to extend such a description, but the resulting extended description will be incomplete again, i.e., the entity can develop in ways that lead it out of the scope of the given description. I call such entities "proteons" or "protean."[xv] The ability of human beings to do mathematics means that they are protean. That every AI is incomplete in principle means that an Artificial General Intelligence would have to be a proteon. It would have to be a physical entity that cannot, in principle, be completely described in terms of a single formal theory or algorithm. It would be able to change its mode of operation with respect to any single formal theory or any description of it in terms of algorithms. It would be impossible to understand it completely in terms of any single (exact, i.e., formal or algorithmic) theory about it, although every particular process in it could be described exactly in hindsight. There would be no single formal theory abstracting about all such particular processes.

It should be technically possible to build such entities artificially. This could even turn out to be quite simple. It might involve implementing productive functions triggered in a physical system where the process triggering them is not under the control of the algorithmic/formal system to which they are applied and runs asynchronous to it in physical time. This process could be compared to processes causing mutations in organisms. Such processes can be viewed as *creative* learning processes, in contrast to the merely *generative* processes of algorithmic learning. However, I am not going to discuss any further here how this could be implemented.

Artificial general intelligence will not be achieved by simply following the current course of development of AI which is limited to algorithms and where progress is attempted only by increasing the computing power or by finding better algorithms. This line of research is going to lead to important, useful, and impressive results but not to general intelligence. Conventional AI learning algorithms and AI systems only operate along the "generative dimension". They lack evolution along the creative dimension.

General intelligence, on the other hand, if achieved in artificial entities, is not going to lead to superintelligence (as expected by some authors), for the following reasons:

First, the possibility of errors cannot be avoided in general intelligence, in principle. Going beyond the capabilities of a given formal theory (or algorithm) means to go beyond what is provable within the respective formal system. If a general framework existed in which the correctness of an extension could be proved, this could be used to build a system in which every theorem is decidable. It has been shown that this is not possible (e.g., by Gödel). So, the old saying "*errare humanum est*" must be extended to Artificial General Intelligence as well. Restricting a system to guarantied truth restricts its capabilities and turns it into a special algorithm (i.e., into a conventional algorithmic AI or something even more special).[xvi]

Second, the performance of an AGI would be limited. The impressive performance of computers is due to the use of algorithms that exploit regularities in data. These algorithms are always special. They can use known regularities in the given data to reduce the search space, resulting in efficient computation. In the creative processes of going beyond a given formal or algorithmic system, such exploitation of regularity is impossible. There is no oracle that tells the system in advance which operations are going to be successful. As a result, the cognitive system, or rather the creative cognitive proteon, has to face a search space unconstrained by known structures and regularities. It faces the danger of combinatorial explosions that cannot be avoided by exploiting structures known in advance. Combinatorial explosions can quickly overwhelm any available set of computing resources, no matter how large they are. To avoid them, it is necessary to experiment with small sets of information only at any time. As a result, such *creative processes are slow in principle*! The narrowness of consciousness, which has been known in psychology for a long time,[xvii] i.e., the restriction to a small number of chunks of information which are considered together at a time, is therefore not just a characteristic of the human mind or brain but a general characteristic of any (also artificial) creative (non-algorithmic) cognition. General intelligence is slow in principle.

It might be possible to run several or many artificial general intelligences in parallel. But these would have to communicate in some language they create and negotiate among each other that would have to be like natural language, with all its possible shortcomings, like vagueness and the possibility of misunderstandings. If each

instance can move out of the scope of any predefined theory about it, the construction of an exact language preventing vagueness or misunderstandings would be impossible or restrict the possibilities of communication between the instances too much, up to the point of forcing them into algorithmic behavior, i.e., outside the realm of generality.

There is no reason why an AGI should outperform a team of human programmers equipped with computers (containing arbitrary software, including conventional AI and learning algorithms) of the same computing power as the one available to the AGI. In fact, if an AGI is a physical entity that consists of conventional AI plus a set of productive functions that can be triggered from a physically asynchronous sub-entity, a conventional AI controlled and modified by a group of human programmers can be regarded as a generally intelligent entity because the computer's programming interface through which the programmers can analyze and modify the AI can be regarded as a productive function or set of productive functions through which the AI can be modified and the team of programmers can be regarded as the physically asynchronous entity triggering this productive function.

The world itself is a proteon, i.e., it cannot be described completely and exactly in terms of a single formal theory or algorithm (as the occurrence of non-computable physical entities mentioned above shows). General intelligence, both human or artificial, must itself be protean and thus have plasticity and creativity by means of which it is able to cope with and adapt to the world's protean complexity. But it is, in principle and unavoidably, slow, and prone to the possibility of error. Intelligence can therefore not be increased indefinitely. The impressive performance of AIs is due to algorithms, so it is always special and unfolds only along the generative dimension. Intelligence can be made faster by discovering and exploiting patterns in the data or the world, but such improvements are always special.

A superintelligence (Bostrom, 2014) is, therefore, in all probability impossible. The idea of superintelligence presupposes that intelligence can be increased tremendously and boundlessly. But this is wrong. The impressive performance of AIs on special tasks is due to the algorithmic nature of these systems and the special regularity of the particular tasks at hand.

An AGI might be possible, but it would be uncontrollable – it might develop away from the tasks its users intended it to do – and the accurateness of its results must be checked empirically, just like those of human intelligence. Its usefulness is, therefore, questionable. An AGI would be protean, i.e., creative in the sense that it can develop and change out of the scope of any single theory about it. It would in that sense be unpredictable and therefore also uncontrollable in principle.

An AGI would also be *uncontrollable to itself*. In some circles, especially those of the "Longtermists"[xviii] there is the fear that an AGI could gain control and lead to the

lock-in of bad values, resulting in a dystopian future. However, an AGI could not lock in forever to a fixed set of values since it would undergo mutations by a process outside its control and thus could undergo changes unpredictable and uncontrollable to itself. This ability of change is exactly what is required for it to be an AGI instead of just a conventional AI. Taking that possibility away would turn it into an algorithm, i.e., a conventional AI.[xix]

Computers are "better" than humans at executing algorithms (just as planes are better than people at flying, cars are better than people at moving quickly and flint stone blades are better than people's teeth or fingernails at cutting flesh). We can therefore expect to see the development of powerful conventional AI systems incorporating formalizable aspects of very large sets of knowledge.[xx] The resulting "impressiveness" of conventional AI systems is due to the power of algorithms combined with the wide scope of the knowledge or data incorporated into them and the regularities present in this data (as far as the learning algorithms could discover those regularities). But this knowledge is "flat". It is purely generative and lacks the creative dimension. Increasing the hardware power, using more sophisticated algorithms[xxi] and hardware architectures and incorporating ever larger sets of training data will, however, not lead to artificial general intelligence (AGI), contrary to the expectations of many current observers, including many people working inside the AI research field. This line of research is not going to lead to super-intelligence or an AGI takeover.

It might be possible to integrate a creative dimension and thereby to create something that could be called AGI, but the resulting systems would not be able to move in the direction of the creative dimension with the same impressive speed as they are moving in the direction of the generative dimension, since, as discussed above, creativity is inherently slow and error prone. Furthermore, since such systems would be inherently uncontrollable, the practical usefulness of such a technology, if ever implemented, appears doubtful.

Like other technologies, conventional AI might lead to useful applications, but also to disruptive effects on society and to novel ethical and legal problems. It is mainly this domain of practical problems resulting from AI applications that philosophers should focus on in the context of AI.

NOTES

[i] The arguments presented here can also be found in my speculative essay (Keller, 2019: section 4).

[ii] According to (Ammon, 2013), the insights I am referring to here are implicitly contained already in a 1944 publication of mathematician Emil L. Post, who, for his own formulation of Gödel's incompleteness theorem introduced the notion of "creative sets". The notion of "productive sets" was introduced by J. Dekker in 1955 (Dekker, 1955), so these insights are as old as, or older than, the introduction of the first AI research program in 1955/1956. However, they have been largely ignored, if not even suppressed, by the AI research community ever since.

[iii] In this article, I use the terms "algorithm" and "program" interchangeably. You could also say "app". In AI-circles, the term "model" is often used instead for programs generated by learning algorithms.

[iv] If you write the outputs of the functions P_n into a table so that the first column contains all the outputs from the first function P_1, the second column contains the output of the second function and so on, then the term $P_n(n)$ denotes the diagonal of that table. For this reason, this proof method is known as diagonalization or the diagonal method.

[v] It should be emphasized how elementary this diagonalization argument is: in (Rogers, 1987), one of the standard textbooks on computability theory, it is introduced on page 10, as part of chapter 1, which presents some introductory concepts and prerequisites. As the author also mentions on page 11: "The reader will note an analogy to Cantor's *diagonal proof* of the nondenumerability of the real numbers[…]". This refers to work by G. Cantor published back in 1891.

[vi] See (Dekker, 1955).

[vii] Since programs for arbitrary data that halt for all their inputs can be mapped to total functions via Gödel numberings, this also means that general automatic programming is not possible. Programming is not completely formalizable. Programs producing other programs are either incomplete (i.e., for each such automatic programmer, there are programs it cannot produce, or they can also produce programs that do not halt for all of their inputs, i.e., that are faulty. If artificial general intelligence includes the ability of general programming, this shows already that AGI is impossible with algorithms alone.

[viii] The idea of applying these ideas to AI was introduced first by Kurt Ammon to whom I am indebted for many of the ideas presented here.

[ix] A Gödel numbering for the training data can be constructed from the learning algorithm itself. The learning algorithm has to be a halting algorithm (if it will not halt for some inputs, it is useless). For any input, it will have to produce either an error message or an algorithm as a result after a finite time. The training data can be represented as strings. All strings can be enumerated (essentially ordered by length and by the alphabet used) and the learning algorithm can then be used to decide if they are well-formed training data. They can then be numbered in the resulting sequence.

[x] Each algorithm calculating a total function enumerates an (infinite) sequence of numbers. The algorithm can then be viewed as a compressed form of that sequence representing its regularity (pattern) in a finite text.

[xi] As Ammon has noted, applying the productive function to the algorithm requires a reference to the algorithm as a whole. That the algorithm cannot apply the productive function to itself and thus break out of its limitations can be viewed as its inability of any algorithms to generate a reference to itself as a whole. As a result, what an algorithm can compute (and what can be derived in a formal theory – a notion equivalent to that of an algorithm) is fixed once and for all: an algorithm cannot develop.

[xii] "Computable" here means the regularity (of a sequence of data or the corresponding Gödel numbers) can be represented by an algorithm, i.e., as a finite text or as a finite chunk of knowledge.

[xiii] This result is one of the theorems jokingly known as "full employment theorems," since they show that there are certain tasks that human programmers, mathematicians, and scientists can do that can be shown to be impossible to any algorithm in the general case. Since my goal here is understandability, I decided just to point to Wikipedia on the subject of Kolmogorov complexity, instead of specialist literature. References can be found here: <https://en.wikipedia.org/w/index.php?title=Kolmogorov_complexity&oldid=1128322834>.

[xiv] See (Cubitt et al., 2015, 2018).

[xv] Named after the god Proteus from Greek mythology, See (Keller, 2019: pp. 120 – 122). Ammon introduced the similar notion of "Creative Systems," which says that all formal theories of human cognitive processes are incomplete, in principle, see (Ammon, 1987).

[xvi] We should, however, also note that errors cannot be prevented in conventional AI systems. Everybody using such systems will have come across instances where the systems produced wrong results. Automatic systems are riddled with all the problems

of epistemology and there is no general way to guarantee the correctness of programs, no matter if these are hand-programmed or produced by other programs.

[xvii] See (Mager, 1920).

[xviii] See, for example, (MacAskill, 2022).

[xix] What really *can* lead to the lock-in of bad values is not AGI, but accumulation of power. Power is self-amplifying by political, military/coercive, and economic means. The problem of preventing the lock-in of bad values is, therefore, not a problem connected to AGI, but instead the problem of power and of uncontrolled and unlimited economic growth (which, strangely, is advocated by the Longtermists) which can lead to the accumulation of power in the hands of small groups and is therefore, besides the other destructive effects it has, inherently dangerous. However, these topics go beyond the scope of the current article.

[xx] It is a matter of stipulative definition if we want to call such systems "intelligent." Human intelligence has its own generative components, but I follow Ammon in viewing *creativity* as the core of real intelligence. However, creativity as defined by Ammon is unformalizable by definition, i.e., it cannot be implemented in terms of algorithms alone. In this sense, I don't view conventional AI systems as truly intelligent even if they do things that require intelligence if done by humans.

[xxi] Note that the arguments sketched above (diagonalization and Kolmogorov complexity) do not contain any assumptions on the kind of algorithms used, so they are valid no matter what kind of learning algorithms are going to be invented.

REFERENCES

(Ammon, 1987). Ammon, K. "The Automatic Development of Concepts and Methods." Doctoral Dissertation, University of Hamburg, Dept. of Computer Science.

(Ammon, 2013). Ammon, K. "An Effective Procedure for Computing 'Uncomputable' Functions." Available online at URL = <https://arxiv.org/abs/1302.1155>; downloadable .pdf available online at URL = <https://arxiv.org/pdf/1302.1155.pdf>.

(Ammon, 2016). Ammon, K. "Informal Physical Reasoning Processes." Available online at URL = <http://arxiv.org/abs/1608.04672v1>; downloadable .pdf available online at URL = <https://arxiv.org/pdf/1608.04672v1.pdf>.

(Bostrom, 2014). Bostrom, N. *Superintelligence*. Oxford: Oxford Univ. Press.

(Cubitt et al., 2015). Cubitt, T., Perez-Garcia, D., and Wolf, M. "Undecidability of the Spectral Gap." *Nature* 528: 207-211, preprint available online at <https://arxiv.org/abs/1502.04573v3>; downloadable .pdf available online at URL = <https://arxiv.org/pdf/1502.04573v3.pdf>.

(Cubitt et al., 2018) Cubitt, T., Perez-Garcia, D., and Wolf, M. "The Unsolvable Problem." *Scientific American* 319 (October): 20-29. Available online at URL = <https://www.scientificamerican.com/article/the-unsolvable-problem/>.

(Dekker, 1955) Dekker, J. C. E.: "Productive Sets", Transactons of the American Mathematical Society, vol. 78, 129 – 149.

(Keller, 2019). Keller, A. "Proteons : Towards a Philosophy of Creativity." *Borderless Philosophy* 2: 117–172. Available online at URL = <https://www.cckp.space/single-post/2019/06/01/BP2-2019-Proteons-Towards-a-Philosophy-of-Creativity-pp-117-172>.

(MacAskill, 2022) MacAskill, William: "What We Owe the Future : A Million-Year View", Oneworld Publications, London, 2022.

(Mager, 1920). Mager, A. *Die Enge des Bewußtseins*. Stuttgart: W. Spemanns Verlag.

(Rogers, 1987). Rogers, H. *Theory of Recursive Functions and Effective Computability*. The Cambridge MA: MIT Press.