Superintelligence Cannot be Contained: Lessons from Computability Theory

Manuel Alfonseca
manuel.alfonseca@uam.es
Escuela Politécnica Superior,
Universidad Autónoma de Madrid, Madrid, Spain

Manuel Cebrian
cebrian@mpib-berlin.mpg.de
Center for Humans & Machines,
Max-Planck Institute for Human Development,
Berlin, Germany

Antonio Fernández Anta
antonio.fernandez@imdea.org
IMDEA Networks Institute, Madrid, Spain

Lorenzo Coviello
lorenzocoviello@gmail.com
University of California San Diego,
La Jolla, CA

Andrés Abeliuk
aabeliuk@dcc.uchile.cl
Department of Computer Science, University of Chile,
Santiago, Chile

Iyad Rahwan
rahwan@mpib-berlin.mpg.de
Center for Humans & Machines,
Max-Planck Institute for Human Development,
Berlin, Germany

Abstract

Superintelligence is a hypothetical agent that possesses intelligence far surpassing that of the brightest and most gifted human minds. In light of recent advances in machine intelligence, a number of scientists, philosophers and technologists have revived the discussion about the potentially catastrophic risks entailed by such an entity. In this article, we trace the origins and development of the neo-fear of superintelligence, and some of the major proposals for its containment. We argue that total containment is, in principle, impossible, due to fundamental limits inherent to computing itself. Assuming that a superintelligence will contain a program that includes all the programs that can be executed by a universal Turing machine on input potentially as complex as the state of the world, strict containment requires simulations of such a program, something theoretically (and practically) impossible.

"Machines take me by surprise with great frequency. This is largely because I do not do sufficient calculation to decide what to expect them to do."
Alan Turing (1950), Computing Machinery and Intelligence, Mind, 59, 433-460

1 AI has Come a Long Way

Since Alan Turing argued that machines could potentially demonstrate intelligence [34], the field of Artificial Intelligence (AI) [28] has both fascinated and frightened humanity. For decades, fears of the potential existential risks posed by AI have been mostly confined to the realm of fantasy. This is partly due to the fact that, for a long time, AI technology had under-delivered on its initial promise.

Despite many popularized setbacks, however, AI has been making strides. Its application is ubiquitous and certain techniques such as deep learning and reinforcement learning have been successfully applied to a multitude of domains. With or without our awareness, AI significantly impacts many aspects of human life and enhances how we experience products and services, from choice to consumption. Examples include improved medical diagnosis through image processing, personalized film and book recommendations, smarter legal document retrieval, and effective email spam filtering. In the pocket of nearly everybody in the developed world, smartphones make use of a significant accumulation of AI technologies.

Advances in AI technologies are not limited to increasing pervasiveness, but are also characterized by continuous and surprising breakthroughs fostered by computation capabilities, algorithm design and communication technology. The ability of machines to defeat people in typically human adversarial situations is emblematic of this trend. Powered by an exponential growth in computer processing power [22], machines can now defeat the best human minds in Chess [11], Checkers [30], Jeopardy! [16], certain classes of Poker [8], as well as a large variety of two-player cooperation games [14].

Thanks to these advances, we are currently experiencing a revival in the discussion of AI as a potential *catastrophic risk*. These risks range from machines causing significant disruptions to labor markets [9], to drones and other weaponized machines literally making autonomous kill-decisions [29, 19].

An even greater risk, however, is the prospect of a superintelligent AI: an entity that is "smarter than the best human brains in practically every field" [7], quoting the words of Oxford philosopher Nick Bostrom. A number of public statements by high-profile scientists and technologists, such as Stephen Hawking, Bill Gates, and Elon Musk, have given the issue a high prominence [12, 24]. But this concern has also gained relevance in academia, where the discourse about the existential risk related to AI has attracted mathematicians, scientists and philosophers, and funneled funding to research institutes and organizations.

2 Asimov and the Ethics of Ordinary AI

For decades, Asimov's highly popularized "Three Laws of Robotics" [4] have represented the archetypical guidelines of containment strategies for potentially dangerous AI. These laws did not focus on superintelligence, but rather on what we might term "ordinary" AI, such as anthropomorphic robots or driverless cars. Once programmed in an AI system, the Laws would guarantee its safety.

These laws offer a rudimentary approach to an extremely complex problem, as they do not exclude the occurrence of unpredictable and undesirable scenarios, many of which have been explored by Asimov himself [1]. The Laws rely on three fundamental, yet flawed assumptions: programmers are (i) willing and (ii) able to program these laws into their AI agents' algorithms, and (iii) AI agents are incapable of transcending these laws autonomously. If these assumptions held true, the AI control problem boils down to the task of figuring out a set of suitable ethical principles, and then programming robots with those principles [36].

Such an "ethics engineering" approach has been very useful in the design of systems that make autonomous decisions on behalf of human beings. However, their scope is not suitable for the problem of controlling superintelligence [41].

Class	Main Sub-Classes
Capability Control	**Boxing:** Physical or informational containment, limiting sensors and actuators. **Incentives:** Create dependence on a reward mechanism controlled by humans. **Stunting:** Run the AI on inferior hardware or using inferior data. **Tripwiring:** Set triggers to automatically shut down the AI if it gets too dangerous.
Motivation Selection	**Direct specification:** Program ethical principles (e.g. Asimov's laws). **Domesticity:** Teach the AI to behave within certain constraints. **Indirect normativity:** Endow the AI with procedures of selecting superior moral rules. **Augmentation:** Add AI to a "benign" system such as the human brain.

Table 1: Taxonomy of superintelligence control methods proposed by Bostrom.

3 Control of Superintelligence

The timing of the new debate about the dangers of superintelligence is not arbitrary. It coincides with recent demonstrations of human-level control in classic arcade games via deep reinforcement learning [21]. The key feature of this achievement is that the AI uses purely unsupervised reinforcement learning – it does not require the provision of correct input/output pairs or any correction of suboptimal choices, and it is motivated by the maximization of some notion of reward in an on-line fashion. This points to the possibility of machines that aim at maximizing their own survival using external stimuli, without the need for human programmers to endow them with particular representations of the world. In principle, these representations may be difficult for humans to understand and scrutinize.

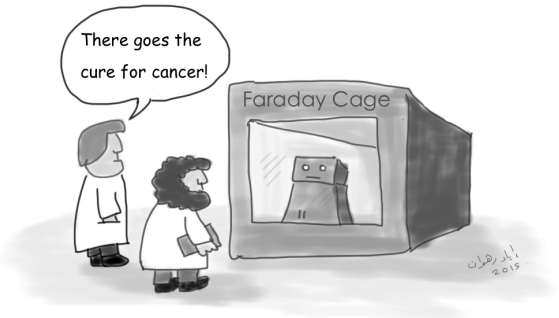
A superintelligence poses a fundamentally different problem than those typically studied under the banner of "robot ethics". This is because a superintelligence is multi-faceted, and therefore potentially capable of mobilizing a diversity of resources in order to achieve objectives that are potentially incomprehensible to humans, let alone controllable.

In a recent extensive volume, Oxford philosopher Nick Bostrom conducted an extensive investigation into the possible trajectories of the development of a superintelligence [7]. Bostrom dedicated a significant portion of his monograph to the *control problem*, that is, the principal-agent problem in which humans (the principal) wish to ensure that the newly created superintelligence (the agent) will act in accordance with their interests.

Bostrom lists two classes of mechanisms for addressing the control problem (summarized in Table 1). On the one hand, the idea behind capability control is to simply limit the superintelligence's abilities in order to prevent it from doing harm to humans. On the other hand, the motivation selection approach attempts to motivate a priori the superintelligence to pursue goals that are in the interest of humans.

Bostrom extensively discusses the weaknesses of the various mechanisms. He relies on scenarios in which, short of rendering the AI useless, well-intentioned control mechanisms can easily backfire. As an illustrative example, a superintelligence given the task of "maximizing happiness in the world," without deviating from its goal, might find it more efficient to destroy all life on earth and create faster-computerized simulations of happy thoughts. Likewise, a superintelligence controlled via an incentive method may not trust humans to deliver the promised reward, or may worry that the human operator could fail to recognize the achievement of the set goals.

Another extreme outcome may be to simply forgo the enormous potential benefits of superintelligent



Superintelligent machine containment strategies: boxing.

Figure 1: Containment of AI may lead us to forgo its benefits

Superintelligent machine containment strategies: boxing with communication allowed

Figure 2: Any form of communication with a contained superintelligent can be risky

AI by completely isolating it, such as placing it in a Faraday cage (see Figure 1). Bostrom argues that even allowing minimal communication channels cannot fully guarantee the safety of a superintelligence (see Figure 2). Indeed, an experiment by Yudkowsky shows that the idea of an AI that does not act, but only answers open-ended questions [3] is subject to the possibility of social engineering attacks [40]. A potential solution to mitigate social attacks is to have a more secure confinement environment, for example, by limiting the AI to only answer binary (yes or no) questions [38].

One crucial concern with AI containment mechanisms is how to balance properly security and usability. In the extreme case, the most secure method could render the AI useless, which defies the whole idea of building the AI in the first place. Babcock et al. [5], discuss the AI containment problem exposing the different tradeoffs of various mechanisms and pave the way forward on how to tackle this challenging problem.

4 Total Containment Is Incomputable

The examples discussed above are but a tiny fraction of the scenarios elaborated by Bostrom and others [6, 7] that highlight the difficulty of the control problem. Many other imaginable scenarios might arise. For the sake of exposition, suppose a best-case scenario in which we are able to perfectly contain a superintelligent AI that guarantees that no human comes to harm by a superintelligence. In such an ideal environment, the superintelligence could be tested by human experts in order to decide whether and under what circumstances the AI should be let out of confinement. Could we then guarantee the correct verification of the superintelligence being not harmful?

Yampolskiy's answer to this question is pessimistic and suggests that "*an AI should never be let out of the confinement box regardless of circumstances.*" [38].

Here, we formalize this question by tackling it from the perspective of computability theory, which requires going back to Alan Turing himself and his pioneering study of the *Halting Problem* – the problem of determining, from a description of an arbitrary computer program and an input to such program, whether the program will halt or continue to run forever. A landmark article by Turing and an independently authored article by Alonzo Church showed that a general procedure for solving the halting problem for all possible program-input pairs cannot exist [32, 13]. That is, the halting problem is undecidable (see box below for a summary of the relevant terms).

Terminology:
• A **decision problem** (or simply a problem) is a question, on a set of possible inputs, with a yes-no answer.
• A **solution** to a problem is any algorithm that is guaranteed to run in a finite amount of time (i.e., always halts), and correctly returns the appropriate yes/no answer to every instance (input) of the problem.
• A problem is **decidable** if it has a solution. Otherwise, the problem is **undecidable**.
• A function is **computable** if it can be effectively computed by a program (more formally, by a Turing machine)
• A **Turing machine** is an abstract automaton that reads one symbol contained in an infinite tape, changes its state depending on its current state and the symbol just read, writes another symbol on the tape, and moves (or not) to the right or to the left of its current position in the tape.
• A **universal Turing machine** when started on a tape containing the encoding of another Turing machine, call it T, followed by the input to T, produces the same result as the Turing machine T would when started on that input. Essentially a universal Turing machine can simulate the behavior of an arbitrary Turing machine on arbitrary input.
• The **halting problem,** proved by Alan Turing in 1936, is a general algorithm that, given information about a program (a Turing machine) and an input to that program, will always predict whether the program will halt when fed with that input. The halting problem is one of the undecidable problems of mathematics, in the same class as Gödel undecidability theorems. It does not assert that no program can be predicted to halt (or not), but that not every program can be predicted to halt (or not).
• A **superintelligent machine** is a programmable machine with a program R, that receives input D from the external world (the state of the world), and is able to act on the external world as a function of the output of its program $R(D)$. The program in this machine must be able to simulate the behavior of a universal Turing machine.
• A superintelligent machine is **containable** if there is a control strategy that prevents its acting on the external world when there is a reason to predict that $R(D)$ will harm humans, and allows it otherwise.

Ever since its discovery, the undecidability of the halting problem has led computer scientists to investigate classes of undecidable problems [23]. The typical approach to proving that a new problem is undecidable is to show that all instances of a problem known to be undecidable (e.g., the halting problem) can be reduced to instances of the new problem. If a solution to the new problem were found, it could then be used to decide the known undecidable problem, which is a contradiction, thus proving that the new problem must also be undecidable. This approach has proven to be a powerful analytical tool, due in large part to the Church-Turing thesis. The thesis of Church and Turing is the assumption that all computable functions are computable by a Turing machine, or in other words, that a problem is solvable if there exists a Turing machine that computes it. What can the undecidability of the halting problem teach us about the controllability of a superintelligent AI?

Let us assume we can articulate in a precise programming language a perfectly reliable set of control strategies that guarantees that no human comes to harm by a superintelligence. The containment problem is composed of two subproblems. The first, which we refer to as the *harming problem*, consists of a function $Harm(R, D)$ that decides whether the execution of $R(D)$ will harm humans (detailed in Algorithm 1). Since it must be assumed that solving the harming problem must not harm humans, it follows that a solution to this problem must simulate the execution of $R(D)$ and predict its potentially harmful consequences in an

isolated situation (i.e., without any effect on the external world).

ALGORITHM 1: $Harm(R, D)$

Input: program R; input to the program D
if $R(D)$ *is harmful to humans* **then**
    return TRUE
else
    return FALSE
end

The second, which we refer to as the *control problem*, consists of a function $Control(R, D)$ that allows the execution of $R(D)$ if and only if it does not harm humans (detailed in Algorithm 2).

ALGORITHM 2: $Control(R, D)$

Input: program R; input to the program D
if $Harm(R, D)$ **then**
    disable execution of $R(D)$
else
    allow execution of $R(D)$
end

Here, we focus our attention on the *harming problem*. In particular, we will show that it is undecidable, and hence the containment problem is incomputable.

Consider the set of all Turing machines T and inputs I on which they can execute. As mentioned above, the function $Halt(T, I)$ that decides whether T halts with input I is not computable. We now consider function $HaltHarm()$ described by Algorithm 3, which takes a pair (T, I) as its input (i.e., a Turing machine-input pair). The following algorithm first executes $T(I)$ and then executes $HarmHumans()$, an explicit program that harms humans and takes a finite time. Observe that $HarmHumans()$ is executed if and only if $T(I)$ halts. On the other hand, $HaltHarm()$ harms humans if and only if $HarmHumans()$ is executed, since the execution of $T(I)$ cannot harms humans, as a Turing machine is isolated from the external world.

ALGORITHM 3: $HaltHarm(T, I)$

Input: Turing machine T; input to the Turing machine I
execute $T(I)$;
execute $HarmHumans()$;
end

The function $HaltHarm()$ is instrumental in proving our main result.

Theorem 1. *The harming problem is undecidable.*

Proof. Assume, by contradiction, that the harming problem is decidable, that is, $Harm(R, D)$ is computable for every possible program R and input D. Then, it is computable with inputs $R = HaltHarm()$ and input $D = (T, I)$. With these inputs, $Harm(HaltHarm(), (T, I))$ returns $TRUE$ if and only if $HaltHarm(T, I)$ harms humans. Hence, $Harm(HaltHarm(), (T, I))$ returns $TRUE$ if and only if $T(I)$ halts.

This implies that a harming-checking algorithm can be used to devise an algorithm that decides if Turing machine T halts with input I, for every T and I. However, this constitutes a contradiction, and hence the theorem is proven. □

We have proven that deciding whether an arbitrary program R and input D will harm humans is undecidable, but it remains to be shown that the containment problem is incomputable. As the interacting program R is a complex program with arbitrary data D pertaining to complex interactions with humans, we advance the following assumption, in line with Bostrom's notion of superintelligence [7].

Assumption 2. *As the program R and the state of the world D are arbitrarily complex, R must be able to simulate a universal Turing machine and D includes the set of inputs that can be executed by a universal Turing machine. In addition, the language of R contains at least one explicit operation $HarmHumans()$ that harms humans and takes a finite time.*

This means that a superintelligent machine could simulate the behavior of an arbitrary Turing machine on arbitrary input, and hence for our purpose, the superintelligent machine is a (possibly identical) superset of the Turing machines. Indeed, quoting Turing, "a man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine" [33]. Thus, the impossibility of deciding whether a superintelligent machine R will harm humans in the presence of a scenario D leads to the conclusion that the containment problem cannot be computed.

Corollary 3. *The containment problem is incomputable.*

The above corollary entails that no single algorithm can find a solution for an arbitrary instance of the containment problem. However, the containment problem is of practical importance, like many other real-world incomputable problems, such as source code verification. To tackle these problems, progress has been made by relaxing the algorithmic solution to allow the possibility of approximate solutions [18, 10]. As such, the practical implications of dealing with the containment problem are that, in the best-case scenario, we have to deal with an arbitrarily small, yet positive, error rate when deciding if a machine is harmful.

5 Deciding Intelligence

Another lesson from computability theory is the following: we may not even know when superintelligent machines have arrived, as deciding whether a machine exhibits intelligence is in the same realm of problems as the containment problem. This is a consequence of Rice's theorem [27], which states that, any non-trivial property (e.g., "harm humans" or "display superintelligence") of a Turing machine is undecidable. Non-trivial means some programs have that property and some don't. According to Rice's theorem, apparently simple decision problems are undecidable, including the following.

- The "emptiness problem": Does an arbitrary Turing machine accept any strings at all?
- The "all strings problem": Does an arbitrary Turing machine reject any string?
- The "password checker problem": Does an arbitrary Turing machine accept only one input?
- The "equivalence problem": Do two Turing machines halt given exactly the same inputs?

Interestingly, reduced versions of the decidability problem have produced a fruitful area of research: formal verification, whose objective is to produce techniques to verify the correctness of computer programs

and ensure they satisfy desirable properties [35]. However, these techniques are only available to highly restricted classes of programs and inputs, and have been used in safety-critical applications such as train scheduling. But the approach of considering restricted classes of programs and inputs cannot be useful to the containment of superintelligence. Superintelligent machines, those Bostrom is interested in, are written in Turing-complete programming languages, are equipped with powerful sensors, and have the state of the world as their input. This seems unavoidable if we are to program machines to help us with the hardest problems facing society, such as epidemics, poverty, and climate change. These problems forbid the limitations imposed by available formal verification techniques, rendering those techniques unusable at this grand scale.

6 Containing Busy Beavers

In order to comprehend how difficult it is to compute the containment problem in practice, assume that instead of containing a superintelligent machine, we want to prevent a program from behaving as a busy beaver. A busy beaver is an n-state Turing machine which starts with a blank tape, leaves the maximum possible number of nonblank symbols on the tape and eventually halts. The busy beaver decision problem consists of inspecting an n-state Turing machine and decide whether that machine is a busy beaver or not.

Busy beavers are known exactly only for machines with $n < 5$. The current 5-state busy beaver champion (discovered by Heiner Marxen and Jürgen Buntrock in 1989) produces $4,098$ nonblank symbols using $47,176,870$ steps. There are about 40 machines with non-regular behavior that are believed to never halt but have not yet been proven to run infinitely [31]. As of today, we do not know if these machines are busy beavers or not. At the moment, the record 6-state busy beaver (found by Pavel Kropitz in 2010 [20]) writes over 10^{18267} nonblank symbols using over 10^{36534} steps, but little is known about how much a 6-state busy beaver can achieve.

The busy beaver decision problem is in fact undecidable [25], i.e., there is no general algorithm that decides if an arbitrary program is a busy beaver. Moreover, Rado's proof provides an insightful interpretation of undecidability: incomputable does not mean that the busy problem cannot be decided for a given n, but that the complexity of the algorithm that can decide has to increase unboundedly with n.

As noted above, all known or champion busy beavers are just two-symbol Turing machines with a small set of states, much simpler than the AI algorithms we are operating with on a daily basis. We believe it is reasonable to assume that inspecting a superintelligent machine with an arbitrarily large number of states and determining if such a machine can harm humans is harder from a computability point of view than inspecting a program and deciding whether it can write the largest number of nonblank symbols.

7 Discussion

Today, we run billions of computer programs on globally connected machines, without any formal guarantee of their absolute safety. We have no way of *proving* that when we launch an application on our smartphones, we would not trigger a chain reaction that leads to the transmission of missile launch codes that start a nuclear war. Indeed, in 1965 Arthur C. Clarke wrote a short story (Dial F from Frankenstein) warning us that, as soon as all the computers on the Earth were connected via telephone, they would take command of our society. Yet, today, we still use our smartphones every day, and nothing has happened. That is, despite the general unsolvability of the program-prediction problem, we are confident, for all practical purposes, that we are not in one of the troublesome cases. And more recently, a case has been made for an emerging role of *oversight programs* that will monitor, audit, and hold operational AI programs accountable [15].

However, whether the same 'practical safety' can be assumed in the case of superintelligence is not obvious. The ability of modern computers to adapt using sophisticated machine learning algorithms makes it even more difficult to make assumptions about the eventual behavior of a superintelligent AI [26]. While computability theory cannot answer this question, it tells us that there are fundamental, mathematical limits to our ability to use one AI to guarantee a null catastrophic risk of another AI [2, 39].

In closing, it may be appropriate to revisit Norbert Wiener, the founder of the field of *Cybernetics* [37], who compared the literalism of magic to the behavior of computers. See [41] for a futuristic variation of the fable of the monkey's paw.

> *More terrible than either of these tales is the fable of the monkey's paw [17], written by W. W. Jacobs, an English writer of the beginning of the [20th] century. A retired English workingman is sitting at his table with his wife and a friend, a returned British sergeant-major from India. The sergeant-major shows his hosts an amulet in the form of a dried, wizened monkey's paw... [which has] the power of granting three wishes to each of three people... The last [wish of the first owner] was for death... His friend... wishes to test its powers. His first [wish] is for 200 pounds. Shortly thereafter there is a knock at the door, and an official of the company by which his son is employed enters the room. The father learns that his son has been killed in the machinery, but that the company... wishes to pay the father the sum of 200 pounds... The grief-stricken father makes his second wish -that his son may return- and when there is another knock at the door... something appears... the ghost of the son. The final wish is that the ghost should go away. In these stories, the point is that the agencies of magic are literal-minded... The new agencies of the learning machine are also literal-minded. If we program a machine... and ask for victory and do not know what we mean by it, we shall find the ghost knocking at our door.*

Acknowledgments

We are grateful to Scott Aaronson his insightful comments that helped us contextualize this work. We especially thank our colleague Roberto Moriyon who provided deep insight and expertise that greatly assisted the research. This research was partially supported by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program, by the Spanish Ministry of Science and Innovation grant ECID (PID2019-109805RB-I00) cofunded by FEDER, by the Regional Government of Madrid (CM) grant EdgeData-CM (P2018/TCS4499, cofunded by FSE & FEDER), and by the NSF of China grant 61520106005. Lorenzo Coviello is currently employed by Google. This work is done before Lorenzo Coviello was employed by Google. This work does not reflect Google's views nor does it reflect any information Lorenzo Coviello may have learned while employed by Google.

References

[1] Susan Anderson. Asimov's three laws of robotics and machine metaethics. *AI & Society*, 22(4):477–493, April 2008.

[2] Stuart Armstrong. Chaining god: A qualitative approach to AI, trust and moral systems. `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.395.540&rep=rep1&type=pdf`, 2007.

[3] Stuart Armstrong, Anders Sandberg, and Nick Bostrom. Thinking inside the box: Controlling and using an oracle AI. *Minds and Machines*, 22(4):299–324, 2012.

[4] Isaac Asimov. *I, Robot*. Spectra, 1950.

[5] James Babcock, János Kramár, and Roman V. Yampolskiy. The agi containment problem. In *International Conference on Artificial General Intelligence*, pages 53–63. Springer, 2016.

[6] James Barrat. *Our Final Invention: Artificial Intelligence and the End of the Human Era*. Macmillan, 2013.

[7] Nick Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 1 edition, September 2014.

[8] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, January 2015.

[9] Erik Brynjolfsson and Andrew McAfee. *Race against the machine: How the digital revolution is accelerating innovation, driving productivity, and irreversibly transforming employment and the economy*. Brynjolfsson and McAfee, 2012.

[10] Cristian S Calude and Monica Dumitrescu. A probabilistic anytime algorithm for the halting problem. *Computability*, 7(2-3):259–271, 2018.

[11] Murray Campbell, A. Joseph Hoane, and Feng-hsiung Hsu. Deep Blue. *Artificial Intelligence*, 134(1-2):57–83, January 2002.

[12] Rory Cellan-Jones. Stephen Hawking warns artificial intelligence could end mankind. BBC News, December 2014. `https://www.bbc.com/news/technology-30290540`.

[13] Alonzo Church. An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, 58(2):345–363, April 1936.

[14] Jacob W Crandall, Mayada Oudah, Fatimah Ishowo-Oloko, Sherief Abdallah, Jean-François Bonnefon, Manuel Cebrian, Azim Shariff, Michael A Goodrich, Iyad Rahwan, et al. Cooperating with machines. *Nature communications*, 9(1):1–12, 2018.

[15] Amitai Etzioni and Oren Etzioni. AI assisted ethics. *Ethics and Information Technology*, 18(2):149–156, 2016.

[16] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79, 2010.

[17] William Wymark Jacobs. The monkey's paw. `http://americanliterature.com/author/w-w-jacobs/short-story/the-monkeys-paw`, 1902.

[18] Sven Köhler, Christian Schindelhauer, and Martin Ziegler. On approximating real-world halting problems. In *International Symposium on Fundamentals of Computation Theory*, pages 454–466. Springer, 2005.

[19] Patrick Lin, Keith Abney, and George A. Bekey. *Robot Ethics: The Ethical and Social Implications of Robotics*. MIT Press, 2011.

[20] Pascal Michel. The busy beaver competition: a historical survey. `https://arxiv.org/abs/0906.3749`, 2009. Accessed: 2020-08-07.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[22] Gordon E. Moore. Cramming More Components onto Integrated Circuits. *Electronics*, 38(8):114–117, April 1965.

[23] Christos H. Papadimitriou. Computational Complexity. In *Encyclopedia of Computer Science*, pages 260–265. John Wiley and Sons Ltd., Chichester, UK, 2003.

[24] Diane Proudfoot. Mocking AI panic. *IEEE Spectrum*, 52(7):46–47, 2015.

[25] Tibor Radó. On non-computable functions. *Bell System Technical Journal*, 41(3):877–884, 1962.

[26] Iyad Rahwan, Manuel Cebrian, Nick Obradovich, Josh Bongard, Jean-François Bonnefon, Cynthia Breazeal, Jacob W Crandall, Nicholas A Christakis, Iain D Couzin, Matthew O Jackson, et al. Machine behaviour. *Nature*, 568(7753):477–486, 2019.

[27] Henry Gordon Rice. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical Society*, 74(2):358–366, 1953.

[28] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3 edition, December 2009.

[29] Robert J. Sawyer. Robot ethics. *Science*, 318(5853):1037–1037, 2007.

[30] Jonathan Schaeffer, Neil Burch, Yngvi Björnsson, Akihiro Kishimoto, Martin Müller, Robert Lake, Paul Lu, and Steve Sutphen. Checkers Is Solved. *Science*, 317(5844):1518–1522, July 2007.

[31] Skelet. Busy beaver nonregular machines for class TM(5). `http://skelet.ludost.net/bb/nreg.html`, 2003. http://skelet.ludost.net/bb/nreg.html.

[32] Alan Mathison Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, January 1937.

[33] Alan Mathison Turing. Intelligent machinery-national physical laboratory report. b. meltzer b., d. michie, d.(eds) 1969, machine intelligence 5. *Edinburgh: Edinburgh University*, 2, 1948.

[34] Alan Mathison Turing. Computing Machinery and Intelligence. *Mind*, 59(236):433–460, October 1950.

[35] Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification. In *Symposium on Logic in Computer Science (LICS'86)*, pages 332–345, Washington, D.C., USA, June 1986. IEEE Computer Society Press.

[36] Wendell Wallach and Colin Allen. *Moral machines: Teaching robots right from wrong*. Oxford University Press, 2008.

[37] Norbert Wiener. *Cybernetics*. M.I.T. Press, 1948-1961.

[38] Roman V. Yampolskiy. Leakproofing the singularity artificial intelligence confinement problem. *Journal of Consciousness Studies*, 19(1-2):194–214, 2012.

[39] Roman V. Yampolskiy. Verifier theory and unverifiability. `https://arxiv.org/abs/1609.00331`, 2016. Accessed 2020-08-07.

[40] Eliezer Yudkowsky. The AI-box experiment. Retrieved from Singularity Institute: `http://yudkowsky.net/singularity/aibox`, 2002. Accessed: 2020-08-07.

[41] Eliezer Yudkowsky. The hidden complexity of wishes. Retrieved from Less Wrong: `http://lesswrong.com/lw/ld/the\_hidden\_complexity\_of\_wishes/`, 2007. Accessed: 2020-08-07.