

Can machines make us think?

In memory of Alan Turing (1912–1954)

Daniel Berrar^{*1} Akihiko Konagaya ^{*1} Alfons Schuster^{*2}

^{*1}Tokyo Institute of Technology, Yokohama, Japan

^{*2}School of International Liberal Studies, Waseda University, Tokyo, Japan

Alan Turing’s question “*Can machines think?*” motivated his famous imitation game, now commonly referred to as the “Turing test”. It was widely assumed that a machine that could pass this test must have intelligence. The construction of such a machine, therefore, was seen by many as the “holy grail” of artificial intelligence (AI). For several decades, the Turing test had a tremendous impact on computer science and has stirred a wide range of philosophical debates. Today, however, the Turing test has nearly vanished from AI research agendas. Here, we argue that the Turing test is still a timely and inspirational source for many researchers. Modern computing machinery is now an integral part in myriads of problem-solving processes, and it is believed that this powerful machinery has revolutionized the way how science is done. Computing machinery now even extends beyond the traditional silicon-based environment to encompass carbon-based, living organisms. This backdrop encourages us to suggest that there is a creative, bidirectional interplay between human intelligence and the increasing sophistication demonstrated by many computer-based applications and systems. We suggest that even though today’s machines may not be able to think, they *can make us think* and encourage *us* to strive for new insights and knowledge. Whenever this new knowledge is fed back into the various types of machines or unconventional computing environments, man and machines will become mutual beneficiaries rewarded with increasing levels of sophistication and intelligence.

1. Introduction

It is difficult to find any other example in computer science that has stirred as many heated debates as the Turing test did since its conception more than six decades ago [Turing, 1950]. For many years, the test has been extolled and deprecated, attacked and defended, over and over again [Saygin et al., 2000]. Although the Turing test is commonly seen as *the* ultimate benchmark test for demonstrating that a machine “has intelligence”, it seems that it is not immune to aging.

Whereas the test was initially a driving force for AI, it has lost its impetus over the last two decades. Since the 1990’s, the test has nearly vanished from the research agendas and is now merely confined to the history books.

Recently, however, Turing-like tests seem to enjoy a renaissance for evaluating living artifacts in systems biology (e.g., artificial worms [Harel, 2005]). In the spirit of the common interpretation of the Turing test, a synthetic organism that is indistinguishable from its natural analog has past such a Turing-like test.

But what exactly had Turing in mind when he asked the now mythical question “*Can machines think?*” The wealth of papers discussing the test shows that the answer is far from being unambiguous. It is clear, however, that Turing did arguably *not* intend the imitation game as an operational definition of intelligence [Whitby, 1996]. In his seminal paper, Turing clearly writes that

“The original question, “*Can machines think?*” I believe to be too meaningless to deserve discussion. Nevertheless I believe that at the end

of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted.” [Turing, 1950, p. 440]

Turing instead proposed an imitation game. Although the original imitation game is more complex, in simple terms, it refers to a blinded conversation where a human judge interacts with another human being and a machine. If the machine can fool the judge most of the time into believing that it is human – hence, that the machine and the human are mostly indistinguishable – then the machine has won the game.

“Turing test” is nowadays commonly interpreted as an intelligence test, and there exist various flavors of the test (e.g., see [Saygin et al., 2000] for an excellent overview). Despite this common interpretation it is often overlooked or unknown that the test (arguably) was not conceived as a test for intelligence. Turing’s imitation game rather provides a framework for a discussion of fundamental concepts in philosophy, cognitive science, and computer science.

Here, we do not intend to contribute a new interpretation to this controversial discussion. Instead, we argue that machines can *make us think* – in the sense that machines somehow amplify human intelligence in a creative interplay that can be beneficial for refining existing theories or even for finding new theories about the natural world. To some degree, our argument rests on the observation that powerful or smart or omnipresent (to avoid the term intelligent) computing machinery has revolutionized how science is done across practically all domains, thereby giving rise to *computational science*. It is a simple fact that modern

computing machinery is now an integral part in a myriad of problem-solving tasks that require intelligent approaches. These intelligent approaches may guide us to new insights, and these new insights can then feed back into existing approaches in order to fine-tune and improve them. Even more, this new knowledge may provide a springboard towards new computing-based technology or theory.

As an example, consider the efforts in various human brain projects^{*1}. The ultimate goal in these projects is to develop a computer-based simulation of a biological human brain. Of course, the process of producing such a system will greatly enhance our understanding about biological nervous systems and the human condition at large. By the same token, however, the product system would be an intelligent supercomputer with capacities that allow the system to perform higher cognitive tasks such as problem-solving, planning, decision-making, or even “thinking”.

In order to further support and more clearly express our argument this paper is organized as follows. Section 2 directs the reader towards some underrated problems of the common interpretation of the Turing test. Then, we take a closer look at some facets of intelligence that may not have received due attention yet. Finally, we discuss an example illustrating how machines “can make us think”.

2. Problems with the Turing test

There certainly is no shortage of criticisms of the Turing test. The most well-known ones include the following:

Criticism 1. The test focuses on only a tiny aspect of intelligence, namely human conversational skills and notably the skills at deceiving. It is obvious that many humans would fail the test, so what is the point of subjecting a machine to it? In section 3.4.2, we discuss an example illustrating that “intelligent communication” does not necessarily need to be defined only from an anthropocentric point of view.

Criticism 2. The test fails to disentangle the observer (i.e., the judge) from the phenomenon (i.e., the machine trying to convince the judge that it is a human being). Consequently, the test has embedded an intrinsic confounder.

Criticism 3. If the test is understood as a test of intelligence, then it is circular: it defines the very notion (“intelligence”) that it claims to be a test for [Hayes and Ford, 1995].

Criticism 4. Lady Lovelace argued that a machine could never surprise us, only execute programs, and therefore could never have intelligence.

Criticism 5. Hayes and Ford [1995] mention a perhaps less well-known problem of the Turing test. The Turing test (and the imitation game) are designed to detect “no difference” between man and machine. Searle does “[...] not see

how one could object to such a test” [Searle, 2009, p. 140]. However, from a practical point of view, how do we look for something that is not there? When we translate the question into the language of significance testing, then the problem is tantamount to a null hypothesis (H0: there is no difference) that we seek to confirm. This, however, is nearly intractable. In significance testing, the null hypothesis would be stated as H0: there is a difference, which we might either reject or fail to reject. More importantly, however, would be the following question: how big is the difference between man and machine? Effect size estimation with confidence intervals are always more meaningful than binary “significant/non-significant” verdicts [Berran and Lozano, 2012].

Criticism 6. A further problem pertains to the definition of terms. Turing was quite clear that terms need to be defined and consequently began his paper as follows:

“I propose to consider the question, “Can machines think?” This should begin with definitions of the meaning of the terms “machine” and “think.” [Turing, 1950, p. 433]

All too often, however, no operational definitions for key terms are given in papers about the Turing test, and the terms therefore often remain elusive. We could live with this state of affairs, if these terms were then not used interchangeably, implying that they refer to the same thing. All too often, no distinction seems to be made between “*thinking*”, “*ability to think*”, “*reasoning*”, “*reason*”; “*intelligence*”, “*consciousness*”, “*mindfulness*”, and “*having a mind*”; “*having a brain*” and “*humanness*”, as if all these words meant the same thing. In a similar vein, “*being mindless*” and “*lacking intelligence*” is sometimes understood to mean the same thing. The juxtaposition of words as in “[...] genuinely intelligent machines and mindless machines.” [Saygin et al., 2000, p. 483] is another confusing example – is “*intelligent*” really the opposite of “*mindless*”? And in what way is a “*genuinely intelligent*” machine different from a merely “*intelligent*” one?

We find it even expedient to consider the difference between the noun “*intelligence*” and the adjective “*intelligent*”. An agent may have “*intelligence*” (whatever that is) yet perform an act that is not “*intelligent*”, and vice versa. So what do we wish to measure – is it an intrinsic ability of the agent or is it something that is linked to an act?

3. Facets of “intelligence”

We do not aim at providing any operational definition of the term “intelligence”. In the remainder of this article, we focus on “intelligent” as an adjective that

1. is time-dependent;
2. involves creativity;
3. and refers to a problem-solving process.

*1 Two of the most prominent projects are the “*Blue Brain Project*” at EPFL (Ecole Polytechnique Fédérale de Lausanne) at <http://bluebrain.epfl.ch/>, and “*SyNAPSE*”, a cognitive computing project from IBM Research at http://www.ibm.com/smarterplanet/us/en/business_analytics/article/cognitive_computing.html.

3.1 Time dependence

Turing believed that the use of words would change over time so that we might indeed speak of machines thinking, at the end of the century [Turing, 1950]. However, the very notion of “intelligence” is also evolving. What we consider “intelligent” today we may not consider “intelligent” tomorrow. Would a machine that had passed the Turing test be labeled “intelligent once and for all”, as Harel [2005] (p.496) argues? That can be questioned. Today, there are countless examples ranging from automated speech recognition to chess computers at grandmaster level that would certainly have surprised Lady Lovelace, to say the least.

In the early 1990’s, chatterboxes have successfully fooled human beings, so technically, we might argue that they have passed the Turing test [Humphrys, 2009]. Still, in 2012, where computer systems such as IBM’s Watson^{*2}, for instance, are supreme champions in popular quiz shows where contestants engage in an answer-and-question style competition, and that are watched live by millions of viewers, we would arguably not consider these earlier programs intelligent. Hence, in our attempt to subject a machine to a test of intelligence, we are moving the goalpost because our own perception of “intelligence” changes over time. Thus, if we describe something as “intelligent”, we imply that it is intelligent only at a given moment in time.

3.2 Creativity

Intelligence is linked to creativity [Berrar et al., 2010]. There exists a panopticum of sometimes substantially diverse definitions of creativity [Scott, 1999, Boden, 1990].

For example, Kryssanov et al. [2001] consider creativity as a “cognitive process that generates solutions to a task, which are novel or unconventional and satisfy certain requirements” (p.332). They note two essential cognitive mechanisms of creativity: (*i*) divergent thinking, which generates original, new ideas, and (*ii*) convergent thinking, which logically evaluates a variety of possible solutions to find the optimal one.

Bisociation is a term that Koestler coined to denote a creative process involving “the sudden interlocking of two previously unrelated skills, or matrices of thought” [Koestler, 1964, p. 121].

Boden [1990] distinguishes between two main types of creativity: (*i*) improbableist creativity (i.e., constructing new concepts by combining existing ones); and (*ii*) impossibilist creativity (i.e., mapping a concept into a new space; for example, Matzingers revolutionary insights into the controlling role of T-cells in immune system responses, which were allegedly triggered by her dog chasing after sheep). Impossibilist creativity is a deeper type of creativity as it requires the mapping, exploration, and transformation of conceptual spaces [Boden, 1990].

According to the theory by Mednick [1962], creativity can operate through three components: (*i*) serendipity (i.e., relationships are found by chance), (*ii*) similarity (i.e., remote associations, for instance through a metaphor model),

or (*iii*) mediation (i.e., cognitive problem-solving) [Scott, 1999]. According to Mednick, the more remote associated concepts are, the more creative is the resulting idea. However, the creative idea must not be only original but also practically useful.

Chess is a very good example illustrating that intelligence and creativity often go hand in glove. In 1946, Alan Turing noted:

“This... raises the question ‘Can a machine play chess?’ It could fairly easily be made to play a rather bad game. It would be bad because chess requires intelligence. We stated...that the machine should be treated as entirely without intelligence. There are indications however that it is possible to make the machine display intelligence at the risk of its making occasional serious mistakes. By following up this aspect the machine could probably be made to play very good chess.” [Hodges, 2008, p. 17].

In 1997, Deep Blue, a computer specifically developed to play chess, won a game against the then reigning world chess champion Garry Kasparov who allegedly said that “he sometimes saw deep intelligence and creativity in the machine’s moves” [Martin, 2008, p.191]. “The decisive game of the match was Game 2...we saw something that went beyond our wildest expectations...The machine refused to move to a position that had a decisive short-term advantage – showing a very human sense of danger.” [Martin, 2008, p. 191].

Again, Turing’s longsightedness amazes us. However, what we consider “intelligent” behavior may sometimes be simply the responses of complex systems, as we will see in section 3.4.1.

3.3 Problem-solving as a collective effort

Conceptually, any engineered and natural system consists of a genotype, i.e., the information that it inherently contains, and a phenotype, i.e., the observable qualities such as the morphology of the system. Naturally, a system operates in an environment, which, together with the genotype and random variations, determines the phenotype.

One of the fundamental characteristics of complex and evolvable systems is robustness [Schuster, 2008, Berrar et al., 2010], which is believed to be characterized by: (*i*) modularity – the system’s components “work together” synergistically; (*ii*) redundancy – some of the components share an identical function; (*iii*) feedback control – the system can detect and react to changes in the environment; (*iv*) spatial compartmentalization – the system has an embodiment with compartments that exchange information with each other; (*v*) distributed processing – collectively, the components give rise to a higher, system-level *gestalt* (e.g., a swarm); and furthermore, for biological systems, (*vi*) extended phenotype – a biosystem could affect its environment to increase its chances of survival. For example, termite mounds might be regarded as the extended phenotype of a termite’s genome. Similarly, albeit contentious,

^{*2} IBM Watson Research Center; <http://www.watson.ibm.com/index.shtml>.

human societies and cultures might also be regarded as the extended phenotypes of the human genome.

Two aspects are particularly interesting here: spatial compartmentalization and distributed processing. An example of a complex system is a biological neural system that has learned an association between patterns and can therefore abstract to similar but new patterns. If we now consider a problem-solving process that involves a machine, then we could argue that that machine is an integral part of a complex system. Thus, if we consider such a process “intelligent”, could we then argue that the machine deserves at least some credit for the intelligent process?

3.4 Examples of unconventional computing

Problem-solving is an important ability of biological systems and essential for survival. Natural problem-solving strategies have inspired the design of highly efficient machine learning algorithms, i.e., algorithms that improve their performance with experience. Many of these algorithms fall into the remit of optimization and were inspired by the intelligent problem-solving strategies observed among insect societies such as, for example, ants and bees. The relatively young field of unconventional computing provides several examples of intelligent problem-solving. The computing machines, however, are not necessarily based on silicon only.

3.4.1 Physarum machines

Physarum polycephalum is a true slime mold that has demonstrated astonishing problem-solving capabilities such as maze solving [Nakagaki et al., 2000, Tero et al., 2007], remembering and anticipating periodic events [Saigusa and Kuramoto, 2008], as well as primitive decision-making [Latty and Beekman, 2010]. This amoeboid organism is a large aggregate of protoplasm whose network of tubular veins enables it to explore its environment [Nakagaki et al., 2004, Aono et al., 2010]. In its vegetative state, this organism slowly crawls along surfaces, mainly searching for food and avoiding sunlight. The two main stimuli (food and light) can be used to train the organism to solve computational tasks. Despite lacking a central nervous system, *P. polycephalum* can make decisions bearing resemblance to a primitive form of intelligent behavior [Latty and Beekman, 2010]. *P. polycephalum* might therefore be considered a programmable, massively parallel, and living computer.

These physarum machines only follow physical and biological laws, though, and what we see as “intelligent behavior” is merely an expression of their responses to external stimuli.

We queried one of the largest scientific databases, ScienceDirect, for those papers that report on *P. polycephalum*. Our analysis revealed the trends illustrated in Figure 1.

Between 1992 and 2007, very few papers reported on this organism in computer science, decision science, and engineering-related fields. Only since 2007, we can observe an increasing interest. In all other fields of science, notably in the biological sciences, we observe an overall decreasing trend in interest since the early 1990’s. In 2001, shortly after the publication of the seminal paper by Nakagaki et al. [2000], *P. polycephalum* regained in popularity again, but

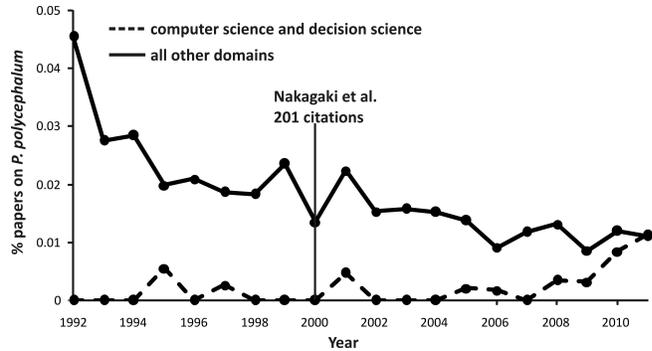


Figure 1: Percentage of papers published that report on *P. polycephalum* in the period from 1992 to 2011.

only for a short time. In 2011, the percentage of papers on *P. polycephalum* was slightly higher in computer science and engineering-related fields than in all other fields combined. Nonetheless, apart from applications in robotics and unconventional computing paradigms, it has received only scant attention in other computer science-related domains. One reason for the still limited interest is certainly that, as a living computer, it is extremely slow, compared with conventional computers. However, computational time is not always a critical factor (e.g., in tasks such as optimal route planning, time sometimes plays a secondary role).

Physarum machines are therefore an interesting novel approach to tackle computational tasks that require slow but “intelligent” decision-making. Indeed, physarum machines were recently used to solve the multi-armed bandit problem [Kim et al., 2010], which is a classical machine learning task.

In his Lecture Notes on Computation, Richard Feynman noted on Turing machines: “We will see that these machines are horribly inefficient and slow – so much so that no one would ever waste their time building one except for amusement – but that, if we are patient with them, they can do wonderful things.” Perhaps a similar description would fit physarum machines.

3.4.2 Collectives of organisms

Naturally occurring, self-organizing collectives of simple organisms, such as bacteria, often exhibit amazing problem-solving capabilities despite their lack of a nervous system [Berrar et al., 2010]. A key element of a bacterial system is *quorum sensing*, the regulation of gene expression in response to fluctuations in cell-population density [Miller and Bassler, 2005]. Bacteria can use quorum sensing as a means of *communication* with each other, for example, to orchestrate attacks by synchronously releasing toxins. Natural quorum sensing was relatively recently discovered, and artificial quorum sensing is a very young field of research in artificial life [Beckmann and McKimley, 2009].

Figure 2a illustrates an abstraction of a digital creature. The expression (or activation) of different genes leads to different phenotypical responses. The creature is equipped with receptors to detect signaling molecules, called auto-inducers, which are produced and released. Under low con-

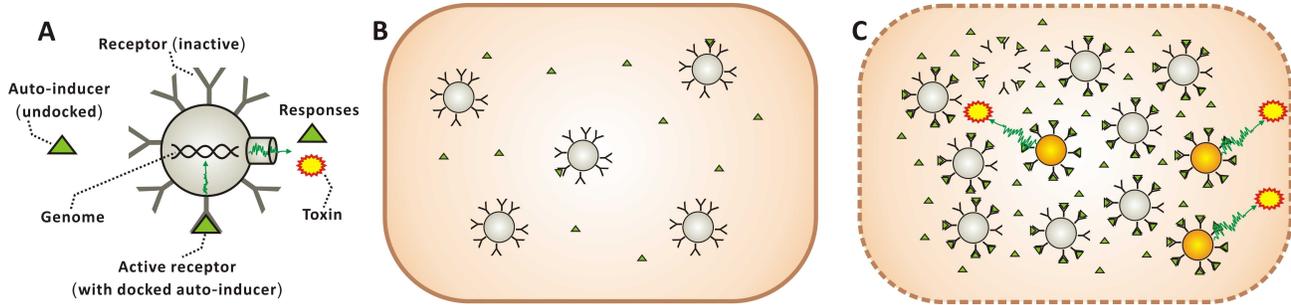


Figure 2: (a) Schematic illustration of a digital creature (modeled on a bacterium) equipped with receptors to detect auto-inducers. Depending on the concentration of activated receptors, different genes are expressed (i.e., activated), which trigger different responses. (b) Under low-density conditions, the auto-inducers remain undetected. (c) Once a quorum has been reached, the creatures begin expressing the same genes and act as a collective, for example, by producing toxins to escape their natural confinement.

centration conditions, they simply diffuse and remain undetected (Figure 2b). However, as the density of creatures increases, the concentration of auto-inducers increases, too, which leads to their detection and subsequent production of more auto-inducers (Figure 2c). This positive feedback loop massively increases the levels of auto-inducers, and once all receptors of a creature are activated, they trigger the activation of other genes. In this example, the activation of an aggressive gene produces toxins that target the cell membrane (i.e., the natural confinement of the bacteria) and allow the invasion of neighboring cells (Figure 2c).

What is important here is the fact that seemingly primitive organisms (such as real bacteria or digital creatures) are capable of evolving intricate means of communication that allow them to act in unison and give rise to a higher, system-level phenotype.

This example illustrates that “intelligent communication” does not necessarily need to be defined from a strictly anthropocentric point of view, as in the imitation game. When Turing wrote

“There would be no question of the machines dying, and they would be able to converse with each other to sharpen their wits” [Turing, 1951, p. 475],

he was referring to digital machines, not living computers. For the imitation game, Turing only permitted digital computers, but his writing makes it clear that his idea of “machines” must have been much broader, and that he even did not make a clear demarcation between man and machine:

“It is natural that we should wish to permit every kind of engineering technique to be used in our machines ... Finally, we wish to exclude from the machines men born in the usual manner.” [Turing, 1950, p. 435]

Today, however, it has become possible to program living organisms with specific functions [Berrar and Schuster, 2011]. For example, Tamsir et al. [2010] implemented all possible two-input gates including the XOR and equal function in a colony of *Escherichia coli* bacteria. Indeed, such machines can die and converse with each other.

4. Can machines “make us think”?

For his imitation game, Turing allowed only digital computers, not engineered carbon-based artifacts. If we consider conventional computers only, can they amplify human intelligence, help us refine or even discover new theories about the natural world, hence “make us think”?

We propose the *dungeon dilemma*, which is mathematically identical to the *Monty Hall problem*, a variant of the *three prisoners problem*. However, the *dungeon dilemma* frames the problem in a different context involving a computer as critical component.

Imagine that Bob is the prisoner in a dungeon. There are three locked doors. The jailor has the keys to these doors.

Jailor: “Only one of these doors leads to freedom. I know which one. If you can find that door, then you may leave the dungeon; otherwise, you will stay. Which door do you choose?”

As Bob has no further information, he thinks that his choice really does not matter – any door can be the good one.

Bob: “I’ll take the door in the middle, then.”

With a big smile, the jailor does not open the middle door but the left door.

Jailor: “Good that you haven’t chosen the left door. See? There is a brick wall behind it! So are you sure that you want me to unlock the middle door? Or perhaps would you like to switch to the right door? Remember, only one door leads to freedom; the other one has a brick door behind it, like the door that I have just opened.”

Bob hesitates a moment.

Bob (thinking): “Why should I change my mind now? After all, there are two doors left. So my chances are now 50:50, aren’t they? But why does he make this proposal, actually? Perhaps I am right with my choice, and he just wants to trick me? Or maybe I am wrong, and he wants to give me another chance? Or perhaps he would have opened one of the two doors with a brick wall behind, anyway, regardless of whether I was right or wrong with my first hit. I really can’t tell.”

Bob’s head starts spinning. But his thoughts about the jailor’s psychology do not get him anywhere. Then Bob

has the following idea.

Bob (thinking): “*I wonder how many prisoners in the same situation could get out by sticking to their initial choice, and how many prisoners could get out by changing their mind. If the numbers are about the same, then it really does not matter whether I stick to the middle door or switch to the right door. But if the numbers are different, then I know what to do!*”

Luckily, Bob had his laptop with him. Hastily, he hacks in a few lines of code in R [R Development Core Team, 2009] that simulate his situation. Bob assumes that the jailor would open one of the doors with a brick wall behind, regardless of Bob’s initial choice.

To his great surprise, Bob obtains the following result: of 100 prisoners in the same situation, 32 would stay in prison by sticking to their initial choice, whereas 68 would go free by changing their mind. This means that his chances of leaving the dungeon are about twice as high if he changes his mind, that is, if he chooses the right door! Not without hesitation, Bob tells the jailor that he decided to switch to the right door.

Jailor: “*Congratulations! You are free to go!*”

Then the jailor unlocks the right door, and Bob happily leaves the dungeon.

Bob’s hastily hacked R code

```
n <- 100; k <- NULL; s <- NULL
for (i in 1:n){
  x <- sample(c(1,0,0)); y <- sample(c(1,2,3))[1]
  keep <- x[y]; p <- which(x==0); o <- setdiff(p,y)
  if(keep==1){o <- sample(o)[1]}; swap <- x[c(-y,-o)]
  k <- c(k,keep); s <- c(s,swap) }
stay <- sum(k)/n*100; free <- sum(s)/n*100
```

Once escaped from the dungeon, Bob started to study his program. He observed that the estimates for **stay** and **free** become more stable with increasing values of n ; and for very large values, he observed that **stay** $\approx \frac{1}{3}$ and **free** $\approx \frac{2}{3}$.

Bob also found an analytical solution to the problem. Before the jailor had opened the left door, his chances^{*3} of being right were $\frac{1}{3}$. Let $P(\text{door})$ denote the chance that a door leads to freedom. Then $P(\text{middle door}) = \frac{1}{3}$. One of the doors must lead to freedom, so $P(\text{left door}) + P(\text{middle door}) + P(\text{right door}) = 1$. This means that $P(\text{left door}) + P(\text{right door}) = 1 - P(\text{middle door}) = 1 - \frac{1}{3} = \frac{2}{3}$. So, the chance that either the left or the right door leads to freedom is $\frac{2}{3}$. This is what he could have inferred *before* the jailor opened the left door. *After* the jailor had opened the left door, he knows that a brick wall is behind it; thus, $P(\text{left door}) = 0$. It therefore follows that $P(\text{right door}) = \frac{2}{3} - 0 = \frac{2}{3}$. Hence, the chances of leaving the dungeon are exactly twice as high if he decides to switch to the right door.

*3 We intentionally choose the term “chance” in lieu of “probability” to avoid digressing on frequentist and Bayesian interpretations of “probability” in this context.

The dungeon dilemma is a variation of the Monty Hall problem, which gained popularity through the TV show “Let’s make a deal”. Here, the quiz master, Monty Hall, offered the candidates to choose among three doors. Behind only one of the doors, there is a prize. After the candidate had announced his choice, Monty Hall opened a non-winning door and offered the candidate to switch. This game sparked heated debates, even among statisticians. Most people found the correct solution counter-intuitive and wrongly believed that the chances are 50:50 after the non-winning door had been opened.

The dungeon dilemma frames the decision-making problem in a deliberately extreme (and actually trivially simple) scenario. But it nicely illustrates a synergistic problem-solving process by a man and a machine. Neither entity alone could have solved the problem. Bob knew how to specify and code the problem, while the machine “knew” how to execute it. Neither Bob nor the machine could have known or even guessed the experimental results (i.e., the numeric values of **stay** and **free**).

5. Conclusions

The Turing test has been a driving force for AI for several decades. More recently, however, the Turing test has nearly vanished from AI research agendas. Indeed, today, it is almost considered irrelevant as a subject of research in AI. Against this backdrop, we argued that Turing’s landmark paper [Turing, 1950] is still an inspirational source that serves well as a framework for the discussion of key issues in philosophy and AI.

Specifically, the Turing test (or, more precisely, the imitation game) is relevant when we study it with an emphasis on the human psychology: how well can we be fooled by sophisticated computer programs on the internet and tricked into disclosing sensitive data?

A particular focus of this paper emerged from Turing’s famous question: “*Can machines think?*” The paper took a step ahead of this question by suggesting a creativity-encouraging feed-back mechanism between human intelligence and the increasing smartness and problem-solving skill demonstrated by many computer-based applications and systems in a variety of conventional as well as unconventional computing environments. The observation that “AI is the business of using computation to make machines act more intelligently, or to somehow amplify human intelligence.”, as noted by Hayes and Ford [1995] (p.977), summarizes this point rather well. And perhaps, Bob in the dungeon dilemma, where a simple computer program could indeed somehow amplify his intelligence, couldn’t agree more.

References

- M. Aono, M. Hara, K. Aihara, and T. Munakata. Amoeba-based emergent computing: Combinatorial optimization and autonomous meta-problem solving. *International Journal of Unconventional Computing*, 6:89–108, 2010.
- B.E. Beckmann and P.K. McKinley. Evolving quorum sensing in digital organisms. *Proceedings of the ACM Genetic and Evolutionary Computation Conference*, 2009.
- D. Berrar and J.A. Lozano. Significance tests or confidence intervals: which are preferable for the comparison of classifiers? *Journal of Experimental and Theoretical Artificial Intelligence*, to appear May, 2012.
- D. Berrar and A. Schuster. The omnipresent computing menace to information society. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 15(7):786–792, 2011.
- D. Berrar, N. Sato, and A. Schuster. Quo vadis, artificial intelligence? *Advances in Artificial Intelligence*, Article ID 629869:3–12, 2010.
- M. Boden. *The creative mind: myths and mechanisms*. Weidenfeld and Nicholson, London, 1990.
- D. Harel. A Turing-like test for biological modeling. *Nature Biotechnology*, 23(4):495–496, 2005.
- P. Hayes and K. Ford. Turing test considered harmful. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, 1:972–977, 1995.
- A. Hodges. Alan Turing and the Turing test. In R. Epstein, G. Roberts, and G. Beber, editors, *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, pages 13–22. Springer, 2008.
- M. Humphrys. How my program passed the Turing test. In R. Epstein, G. Roberts, and G. Beber, editors, *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, pages 237–260. Springer, 2009.
- S.J. Kim, M. Aono, and M. Hara. Tug-of-war model for multi-armed bandit problem. *Proceedings of Unconventional Computing, Springer Lecture Notes in Computer Science*, 6079:69–80, 2010.
- A. Koestler. *The Act of Creation*. London, Hutchinson, 1964.
- V.V. Kryssanov, Tamaki H., , and Kitamura S. Understanding design fundamentals: how synthesis and analysis drive creativity, resulting in emergence. *Artificial Intelligence in Engineering*, 15:153–169, 2001.
- T. Latty and M. Beekman. Speed-accuracy trade-offs during foraging decisions in the acellular slime mould *Physarum polycephalum*. *Proceedings of the Royal Society B: Biological Sciences*, 278(1705):539–545, 2010.
- R.M. Martin. *Philosophical Conversations*. Broadview Press Ltd., 2008.
- S.A. Mednick. The associative basis of the creative process. *Psychological Review*, 69(3):220–232, 1962.
- M.B. Miller and B.L. Bassler. Quorum sensing in bacteria. *Annual Reviews in Microbiology*, 55:165–199, 2005.
- T. Nakagaki, H. Yamada, and A. Tóth. Maze-solving by an amoeboid organism. *Nature*, 407(6803):470, 2000.
- T. Nakagaki, R. Kobayashi, Y. Nishiura, and T. Ueda. Obtaining multiple separate food sources: behavioural intelligence in the *Physarum plasmodium*. *Proceedings of the Royal Society London, Series B*, 271:2305–2310, 2004.
- R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. URL <http://www.R-project.org>. ISBN 3-900051-07-0.
- T. Saigusa and Y. Kuramoto. Amoeba anticipate periodic events. *Physical Review Letters*, 100(1):018101, 2008.
- A.P. Saygin, I. Cicekli, and V. Akman. Turing test: 50 years later. *Minds and Machines*, 10:463–518, 2000.
- A. Schuster. *Robust Intelligent Systems*. Springer Verlag, London, 2008.
- T.E. Scott. Knowledge. In S.R. Pritzker and M.A. Runco, editors, *Encyclopedia of Creativity, Vol.1*, pages 119–129. Academic Press, 1999.
- J.R. Searle. The Turing test: 55 years later. In R. Epstein, G. Roberts, and G. Beber, editors, *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, pages 139–150. Springer, 2009.
- A. Tamsir, J.J. Tabor, and C.A. Voigt. Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature*, 469:212–216, 2010.
- A. Tero, R. Kobayashi, and T. Nakagaki. A mathematical model for adaptive transport network in path finding by true slime mold. *Journal of Theoretical Biology*, 244(4):553–564, 2007.
- A.M. Turing. Computing machinery and intelligence. *Mind*, 59:433–460, 1950.
- A.M. Turing. Intelligent machinery, a heretical theory. *Philosophia Mathematica (reprinted 1996)*, 4(3):256–260, 1951.
- B. Whitby. The Turing test: AI's biggest blind alley? In P. Millican and A. Clark, editors, *Machines and Thought: The Legacy of Alan Turing*, pages 53–62. Oxford University Press, 1996.