Practical applications of Philosophy in Artificial Intelligence

Karim Oussayef

Among the sciences, Artificial Intelligence holds a special attraction for philosophers. A.I. involves using computers to solve problems that seem to require human reasoning. This includes computer programs that can beat human opponents at games, automatically find and proof theorems and understand natural language. Some people in the AI field contend that programs that solve these types of problems have the possibility of not only thinking like humans, but also understanding concepts and becoming conscious. This viewpoint is called strong AI[1]. Many philosophers are concerned with this bold statement and there is no shortage of arguments against the metaphysical possibility of strong AI. If these philosophical arguments against strong AI are true then there are limits to machine intelligence that cannot be surpassed by better algorithms, faster computers or more clever ideas.

Hilary Putnam in his paper *Much Ado About Not Very Much* asks "AI may someday teach us something about how we think, but why are we so exercised about it now? Perhaps it is the prospect that exercises us, but why do we think now is the time to think decide what might in principle be possible?" The reason we are so exercised about A.I. is because knowing whether true intelligence is a possibility will change the goals of researchers in the field. If strong AI is not possible then the best we can hope for is a program that acts humanly but doesn't think humanly. Even this goal is a very difficult and many programs seek to achieve it. Cycorp[2] is a company whose software attempts to

[1] Coined by John Searl in *Minds, Brains and Programs*.
[2] Information from Cycorp's website.

mimic human intelligence by creating a huge database of common sense facts. Their website gives some examples: "Cyc knows that trees are usually outdoors, that once people die they stop buying things, and that glasses of liquid should be carried right side up."

To illustrate how a fact-based program such as Cycorp's would try to solve a simple problem let us turn to the Turing test[3]. Turing reasoned that a computer could prove that it was artificially intelligent by fooling a person into thinking it was another human being. His test was modeled from this reasoning: A human would type questions to either another human or a computer (he or she wouldn't know which) for a certain amount of time. If that person couldn't tell at the end of the time which of the two he or she was talking to, the computer would pass the test (and therefore Turing reasoned, be artificially intelligent). Let me stress that I am not arguing that the Turing test is a good one for determining if a computer can think; I am simply using it to demonstrate how a program might go about solving a problem. The fact-based program mentioned above might try to answer the simple question "What is a car?" by supplying the information that was in its code: "A car is a small vehicle with 4 wheels". A harder question might have to do with a description a car object followed by "What am I describing?" This could be answering by going down a tree of facts as follows: *The description is of a vehicle, search for all the objects under the vehicle topic. It has four wheels; discard the possibility of the motorcycle. It is light; discard the possibility of the truck. Conclusion: It must be a car.*

A program like this could pass the Turing test if it was given enough data. However it has many disadvantages. First it requires someone to input a vast amount of

[3] Introduced by Alan Turing's article Computing Machinery and Intelligence in 1950.

information manually. Although the program is capable of making some extensions of the given information, it still needs millions of hard facts. Cycorp's database has been painstakingly entered using over 600 person-hours of effort since 1984. The list of facts now stands at 3 million (Anthes). Second the machine doesn't seem to work like a human, it looks up rules and then gives an answer instead of figuring out what the question means.

Searle's Chinese room analogy shows why this program isn't an example of strong AI. Imagine an English speaking person inside of a small room. This person has access to a large rulebook, which is written in English. Other people outside the room can pass notes written in Chinese to him through a small hole in the wall. Although the person inside the small room cannot speak Chinese, he uses the complex rulebook to give back an appropriate response to the Chinese writing in Chinese. Also imagine that this rulebook is so well written that the answers the person inside the room gives back are indistinguishable from the answers that a native Chinese speaker might give back. This "man in a room" system would be able to carry on a written conversation with a native Chinese speaker on the other side of the wall. In fact the Chinese person might assume he was speaking to another person who understands Chinese. We can plainly see however, that the person does not.

This analogy is disastrous for fact-based AI. In the same way that the computer passes the Turing test by fooling humans into thinking it is another human, the English speaker can fool native Chinese speakers into thinking that he understands Chinese. To further explain, the person inside the room is analogous to the computer CPU; they both know how to interpret instructions. The rulebook is analogous to the program; they

supply the instructions to obtain the intended result. The computer programmed with this fact-based knowledge does not understand English any more than the English speaker understands Chinese. Both of them are following rules instead of understanding what is being asked and responding based their interpretation.

The defeat of the fact-based program poses problems for strong A.I. supporters. It shows that any program that relies on pre-made a set of rules (no matter how complex) cannot understand in the same way that a human mind does. In fact Searle argues: "… in the literal sense the programmed computer understands what the car and the adding machine understand, namely, exactly nothing" (Searl 511). However Searle's argument doesn't rule out all programs. A program that learns from scratch, without the use of a rulebook or a prefabricated fact database, can understand in the same way that a human can. I will now go about describing such a program.

To construct the fact-based program we attempted to record facts about the world. The learning program takes an orthogonal approach. It attempts to program the computer to learn these facts for itself. To see how to go about this let us examine how a small child learns. A child comes into the world knowing very little. She does not know how to talk, walk or understand English. She goes about learning these abilities with three tools. First she has basic goals or needs. Some of a child's needs are food, water and shelter. Second she can observe the world. A child can tell that when she is eating, she is getting less hungry. Finally she can remember what has happened to her. Let me demonstrate how these three tools allow her to learn something. Imagine that this child is hungry. She *observes* that when she cries her mother brings her food. She *remembers*

what has happened to her and finally her *need* for food causes her to cry again the next time she's hungry. Her tools have allowed her to learn that crying results in getting food.

These three tools are the core of the learning program. However, the goals of a computer will differ from the goals of a human. A computer has no need for food or water so they are not appropriate goals. Instead these goals can be anything that A.I. programmers think are important. Isaac Asimov proposed three such goals (or laws) in his fictional stories[4]:

> 1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
>
> 2. A robot must obey the orders given it by human beings, except where such orders would conflict with the First Law.
>
> 3. A robot must protect its own existence, as long as such protection does not conflict with the First and Second Laws.

In short a robot's goals are human well-being, human will and its own well-being. These goals can be implemented in the form of variables linked to actions that the computer might perform. Whenever the computer does something that accomplishes one of its goals it might raise the value of the variables connected with its current state or action. Similarly it would lower the values of these action-variables when it did something against its goals. These variables also represent the computer's memory. This is where the computer remembers what to do the next time it is in a similar situation. Finally the computer needs a console, sensors or some other form of input so it can observe what is happening around it. Let me demonstrate how it works with a simple example.

Imagine a robot equipped with a camera, a flashlight and wheels. The robot is put in an environment and given the extra goal of reaching a certain spot. If the robot had

[4] First published in Runaround in 1940.

never been in this situation before it might have no idea of how to reach the goal in much the same way that the child does not know how to get food. So it might begin by doing any number of things. Perhaps it would turn on its flashlight. This would not help it reach it's goal so would try something different. Maybe it starts driving towards the goal. The robot would *observe* that it is accomplishing a goal so the "going forward" action might get a "+ 1 points" in the "trying to reach an object" context. Perhaps there is a wall in front of it halfway to the flag. It runs into the wall and damages itself. This is bad for the "well-being of self" goal so the "driving forward" action might get "–1 points" in the "wall in front of me" context. These point value will help it *remember* what to do next time it is trying to get from one point to another. When it sees a wall infront of it in the future, the robot will see that "driving forward" has less points than, say, "driving sideways" and might pick that option. The fact that it *wants* to reach its goals will teach the robot through trial and error. Eventually it will learn how do drive around objects (instead of into them).

I argue that a robot constructed in this fashion would actually understand how to accomplish goals. To support this belief, let's see if it does any better with the Chinese room example. Remember that for the fact-based program the person inside the room is analogous to the computer CPU and the rulebook is analogous to the program. However, for the learning program there is no rulebook. The person inside the room is analogous to both the CPU and the program. Instead of people asking questions and having him answer back, imagine that the input through the slot in his room is the information he receives from the outside world. At first he has no idea what this input means. He sends random symbols back but after a while he notices a correlation between what he sends

out and what he gets back. He starts to write his own rulebook in his head from this information that allows him to translate Chinese input into English. When he writes back he translates the answers that he thought of in English back to Chinese.

The way the "learning-program person" can communicate in Chinese is drastically different than the way the "fact-based person" does. The "learning-program person" learns what the Chinese means by association. From his knowledge he knows the sense of the words. Some people may point out that he does not actually think in Chinese so he must not understand the language. However, there are many people who converse in a non-native tongue. We cannot claim that these people's understanding of the world is different than our own.

Searl might respond to this learning-program by saying that the person inside the Chinese room would simulate the entire learning process and that the learning is not internal but external. This means that the person inside of the room is following directions that correspond to learning but he himself is not learning. But if such a program falls victim to the Chinese room, wouldn't a human brain fall victim as well? Let us imagine a modified Chinese room for the human brain. Instead of the man inside of the Chinese room simulating a computer program, he simulates the neurons in someone's brain. When he receives input, he would keep track of what neurons get excited and calculate whether or not they fire. He would know from his rulebook (a compendium of the laws of physics, chemistry and biology that would allow him to completely simulate the inner workings of the brain) that when certain neurons fired that he should output an answer. The person simulating the brain doesn't understand Chinese any better than the one simulating a computer program. Why would one be different than

the other? Searl's opinion is that "actual human mental phenomena might be dependant on actual physical-chemical properties of actual human brains" (Searl 519). Penrose's "The emperor's new mind" provides insight as to why this may be the case.

Penrose mentions many physical processes that are not computable. He first examines the Mandelbrot set. The Mandelbrot set is created by mapping a formula using the combination of real and complex numbers. The result is an Argand Plane. Here is where Penrose brings up an important comment: "We might think of using some algorithm for generating the successive digits of an infinite decimal expansion, but it turns out that only a tiny fraction of the possible decimal expansions are obtainable in this way: the *computable numbers*" (Penrose 648). In other words, the exact notion of the Mandelbrot set cannot be computed with a computer. Penrose also mentions quantum mechanical principles. Tiny sub-atomic particles do not follow the same laws of physics that larger objects do. The superposition principle states that a particle can be in many different states at the same time. These states are defined by factors of complex numbers and thus are another example of a physical law that cannot be simulated in a computer.

These two examples may show why the Chinese room cannot simulate the human brain. When the person inside of the room was following the directions for simulating a computer the steps he took were explained by a well-defined algorithm. This is because computers are Turing machines, a concept that was formalized elegantly by Alan Turing. All Turning machines can be thought of as a device that reads and writes from an infinitely long tape. On the tape is a sequence of partitions that are either blank or marked. The device operates by moving either left or right on the tape. It can change the current section to either "marked" or "blank" and read its current state. It does this by

following a finite set of instructions. This simple abstraction is enough to run any computer program no matter how complex. It is easy to think of the human inside of the Chinese room controlling a Turing machine.

The brain may, however, rely on non-algorithmic processes than the person inside the Chinese room will not be able to follow. If, for example, neuron X would fire only because of a certain arrangement of subatomic particles, there would be no hard set directions for what the Chinese-room-person should do. Perhaps the next instruction has a random chance of occurring, if so the person will be confused and unable to complete the instruction. It is important to find out whether the brain makes use of these processes because if it does, it would explain why the Chinese room works for computers but not for the human brain.

In the chapter “Where lies the physics of the mind,” Penrose argues that the brain does indeed make use of non-computable phenomenon. He contends that expressions that deal with consciousness such as “understanding” and “judgment” and those that do not such as “mindlessly” and “automatically”, suggest a distinction between two parts of the brain: algorithmic and non-algorithmic (Penrose 653). Penrose brings up Godel’s incompleteness theorem as an example of how the brain makes use of non-algorithmic part of the brain. Godel encoded first order predicate calculus into normal arithmetic using prime numbers. By breaking down F.O.P.C. in this way, he could write out arithmetic formulas that would equate to either true or false. He used this trick to demonstrate that there are some statements that cannot be proven or disproved. One such sentence would be: "A computer which knows the answer to all questions will never

prove that this sentence is true."[5] Human beings know that this sentence is true *without actually going through the process of proving it.* If, however, a computer attempts to assess the validity of the state through a formal proof it will be confused because the statement remains true until the proof is complete.

Penrose argues that these types of sentences, which humans can reason about, would be impossible for a computer to understand. What Penrose doesn't notice is that even if some statements could not be proved or disproved using FOPC logic, there are other ways for computers to approach these problems. There is no reason that computers couldn't use higher logic to solve puzzles just like a human does. Penrose's goal of proving strong A.I. impossible fails because he doesn't make the link between the non-algorithmic/non-computable physical phenomenon and the human brain. If in the future neuroscientists discovered that the brain relies on such processes then his argument would hold more weight. Still, it would be possible for a program to simulate the workings of the brain without simulating the actual physical processes.

In fact, computers and human brains excel at different tasks, a fact which makes literal simulations wasteful. A computer can remember things for an infinite amount of time (assuming the file isn't deleted). It can also compute complicated mathematical expressions in milliseconds. Even a human with the best eidetic memory or an extraordinary mathematical talent couldn't rival a computer in these tasks. On the other hand, computers have a very hard time recognizing objects such as human faces. In dark or light, different clothes or dyed hair, we can still recognize our best friend. Similarly the human ability to understand language is amazing. We can utter sentences that we have never said or heard before and understand a variety of accents and slang. These

[5] Adapted from Denton

"human algorithms" which require almost no effort for us are very difficult for a computer. To throw away a computer's advantages in mathematics, memory and many other tasks seem a waste. Yet attempting to create a model of human neurons seems to do exactly that. Instead, it would be better to attempt to simulate the way a human brain solves problems instead the actual physical processes behind human thinking.

In this paper I have shown how various arguments against strong A.I. interact. These arguments do not show that it is impossible but do restrict what kind of programs can be thought of as "truly intelligent". Searl's Chinese room argument shows that fact-based programs are incapable of understanding things in the same way as humans do. It also excludes programs that have all their information hard coded in. Learning is essential to programs that wish to support strong A.I. because information has to come from the program, not from the programmer. Penrose has suggested that the brain is unable to be simulated by a computer. If this is true than computers must be a simulation of how the brain thinks not how the brain works. Finally Godel's incompleteness theorem shows that programs must use higher reasoning to achieve its goals. Philosophy is often criticized for being un concerned with real world implications but in this case it has shown the best direction for A.I. researchers to explore.

References

Books

Clancey, William J. 1997. Situated Cognition. Cambridge, UK: Cambridge University Press.

Dreyfus, Hubert. 1992. What Computers Still Can't Do: A Critique of Artificial Reason. Cambridge, MA: MIT Press.

Kim, Jaegwon. 1998. Philosophy of Mind. Boulder Colorado: Westview Press Inc.

Penrose, Roger. 1989. The Emperor's New Mind: Concerning Computers, Minds and the Laws of Physics. Oxford: Oxford University Press.

Russell, Smart and Norvig, Peter. 1995, Artificial Intelligence: A Modern Approach

Smith, Brian Cantwell. 1996. On the Origin of Objects. Cambridge, MA: MIT Press/Bradford Books.

Papers

Dennett, Daniel C. 1988. When Philosophers Encounter Artificial Intelligence. The Artificial Intelligence Debate: False Starts, Real Foundations: 283-296.

Fodor, J.A. 1980. Searl on What Only Brain Can Do. The Nature of Mind: 520.

Fodor, J.A. 1998. After-thoughts: Yin and Yang in the Chinese Room. The Nature of Mind: 524.

LaForte, Geoffrey, Patrick J. Hayes, and Kenneth M. Ford. 1998. Why Godel's Theorem Cannot Refute Computationalism. Artificial Intelligence: 211-264.

McCarthy, Daniel C. 1988. Mathematical Logic in Artificial Intelligence. The Artificial Intelligence Debate: False Starts, Real Foundations: 297-311

Putnam, Hillary. 1988. Much Ado About Not Very Much. The Artificial Intelligence Debate: False Starts, Real Foundations: 269-282.

Sokolowski, Robert. 1988. Natural and Artificial Intelligence. The Artificial Intelligence Debate: False Starts, Real Foundations: 45-64.

Searl, John R. 1980. Minds, Brains and Programs. The Nature of Mind: 509-519.

Searl, John R. 1980. Author's response. The Nature of Mind: 521-523.

Searl, John R. 1998. Ying and Yang Strike Out. The Nature of Mind: 525.

Turing, A.M. (1950). Computing machinery and intelligence. Mind, 59, 433-460.

Journals

Gary H. Anthes, *Computerizing Common Sense*. Computerworld. 4/8/02.

Electronic

Cycorp: Company Overview. http://www.cyc.com/overview.html

Denton, Willaim. 2000. Godel's Incompleteness Theorem http://www.miskatonic.org/godel.html