The Four Sons of Penrose

Nachum Dershowitz*

School of Computer Science
Tel Aviv University
Ramat Aviv 69978, Israel
nachum.dershowitz@cs.tau.ac.il

Abstract. We distill Penrose's argument against the "artificial intelligence premiss", and analyze its logical alternatives. We then clarify the different positions one can take in answer to the question raised by the argument, skirting the issue of introspection *per se*.

1 The Argument

> It follows that there are four sons:
> one wise; and one wicked;
> one simple; and who knows not how to ask.
>
> —*Mekhilta of R. Ishmael* (c. 300)

Artificial Intelligence (AI) is the endeavor to endow mechanical artifacts with human-like intellectual capacities. The "strong" AI hypothesis (as propounded in [7], for example, and critiqued in [18]) avows that "an appropriately programmed computer really is a mind" [18]. The Computational Hypothesis asserts that the human mind is in reality some kind of physical symbol-manipulation system. The "weak" version of the hypothesis ("A physical symbol system has the necessary and sufficient means for intelligent action." [13]) allows for the possibility that the mind is not mechanical, but claims that it is (theoretically, at least) simulatable by mechanico-symbolic means (to wit, by a Turing machine).[1]

In *The Emperor's New Mind* [14] and especially in *Shadows of the Mind* [15], Roger Penrose argues against these AI theses, contending that human reasoning cannot be captured by an artificial intellect because humans detect nontermination of programs in cases where digital machines do not. Penrose thus adapts the similar argumentation of Lucas [11]. The latter was based on Gödel's incompleteness results, whereas Penrose uses the undecidability of the halting problem, demonstrated by Turing [22].

In a nutshell, Penrose's argument runs as follows:

1. Consider all current sound human knowledge about non-termination.

* This research was supported by the Israel Science Foundation (grant no. 250/05).

[1] For a discussion of problems inherent in comparisons of computational power via simulations, see [2].

2. Suppose one could reduce said knowledge to a (finite) computer program.
3. Then one could create a self-referential version of said program.
4. From the assumed existence of such a program, a contradiction to its correct performance can be derived.

Penrose's resolution of this contradiction is to deny the validity of the second step: No program can incorporate everything (finitely many) humans know. This, it would seem, violates even the weak AI premiss.

Since some (immortal) humans can emulate (unbounded) Turing machines, while machines—according to this argument—cannot simulate all humans, Penrose concludes that the human mind comprises super-Turing abilities, using undiscovered physical processes. (For a more recent dispute over whether quantum physics supports potentially super-Turing computability, see [8, 21, 9, 19].) Penrose's conclusions have been roundly critiqued, for example, in [1, 3, 5, 10, 16].

In this paper, we distill the arguments on both sides. Specifically, we reduce the bone of contention to a consideration only of the question, "Does X not respond to input X?", and restrict ourselves to one entity versed in computer science, namely, "Roger". In the process, we demonstrate that there are exactly four ways to resolve the conundrum raised by the above "diagonalization" argument. Roger falls into one (or more) of the following categories:

I. An idealized human who is inherently more powerful than Turing's machines.
II. A slipshod human who can err in judgement.
III. An impetuous human who sometimes errs, having resorted to a baseless hunch.
IV. A pedantic human who may decline to express an opinion when questioned.

The analysis remains the same regardless of whether the entities involved are human, humanoid, or otherwise endowed with reasoning abilities. Knowledge of one's self-consistency does not directly enter the equation.

Most discussions exclude options II and III, as irrelevant when considering "ideal" beings. Thus, it appears that IV, though rarely proposed explicitly in these terms, is the preferred alternative for those who, unlike Penrose, do not accept I. It goes without saying that real, corporeal mortal, humans suffer from both II and III, and ultimately from IV, and—in the final analysis—have no more computational power than sub-Turing finite automata.

In Sect. 3, we recapitulate a simplified version of Turing's proof of the undecidability of the halting problem. Before and after that section, we give a fanciful rendition of the interplay between soundness (never giving a wrong answer) and completeness (in the sense of always knowing when the answer is "yes"). Section 5 defines transfinite sequences of better and better programs for termination analysis. In Sect. 6, we introduce the entities that play a rôle in our analysis. After setting the stage, we present our quadriad of possible solutions in Sect. 7. Finally, in the concluding section, these alternatives are matched up with some of the different published opinions on the subject.

2 The Androids

> Thousands of battle droids, super battle droids,
> droidekas and other models
> are built from start to finish within the factory.
>
> —`starwars.com`

Androids have become more and more commonplace in the 21st century. Each specimen is identified by model# and serial#. The older Model-T units are being phased out. Most modern consumer models belong to either the R series (circa 2001) or S series (circa 2010). Intelligence engineers have worked hard over the years to continually lower response time, without compromising performance quality. The R series is quite impressive, with guaranteed response time nowadays of less than one minute. Reaction to this series, however, has been mixed, since R-series androids have been known to occasionally give wrong answers and, hence, cannot be trusted with sensitive tasks. Despite manufacturer claims that such occurrences are extraordinarily rare, and that normal household use is highly unlikely to suffer, the fact is that complaints continue to stream in.

In response to customer demands, the S series was launched, in which reliability was made a top priority. These androids came with a "money back" guarantee of correctness, for which purpose logicians were hired by android manufacturers. Reviews of this series remain mixed, however. As it turns out, some questions seem to befuddle members of this class, and unreasonably long delays have been experienced before an answer was forthcoming. Some questions took so long, that the "last resort" restart procedure was manually invoked.

It has become something of a geek game to come up with neat questions that trip-up R-units and/or stump S-units. A simple litmus test to distinguish between these two series is to ask the "trick question":[2]

Will you answer "no" to this question?

All R models give a wrong answer, though some answer in the affirmative and others in the negative. On the other hand, no S model answers within a minute, or—indeed—has ever been known to answer this trick question. In fact, this question belies claims that R-series droids will never fail in ordinary day-to-day use.

In response to customer dissatisfaction, a new model has just hit the market. It is the vanguard of the much-vaunted Q-series, which promises to harness quantum technology to overcome shortcomings of the R and S models. Whether it will be a success remains to be seen.

[2] I have not yet found the origin of this riddle.

3 The Halting Problem

This statement is false.
—Eubulides (c. −350)

The argument for undecidability of the halting problem, as in the seminal work of Alan Turing, is by *reductio ad absurdum*. We provide a full "one-minute proof" of the undecidability of a special case (viz. self-divergence), inspired by Doron Zeilberger's "2-minute proof" [24] and by Penrose's claims. The idea is to formalize a paraphrasing of the trick question of the previous section, namely,

Will you not answer "yes" to this question?

computationally.

Consider any programming language supporting programs as data (as in typical AI languages), which has some sort of conditional (**if ... then ... else ...**) and includes at least one non-terminating program (which we denote **loop**). Consider the decision problem of determining whether a program X diverges on itself, that is, $X(X) = \perp$, where $\perp$ denotes a non-halting computation. Suppose A were a program that purported to return true (T) for (exactly) all such X. Then A would perforce fail to answer correctly regarding the behavior of the following (Lisp-ish) program:

$$C(Y) := \textbf{if } A(Y) \textbf{ then } T \textbf{ else loop}() \,,$$

since we would be faced with the following contradiction:

$$C(C) \text{ returns } T \quad \Leftrightarrow \quad A(C) \text{ returns } T \quad \Leftrightarrow \quad C(C) \text{ diverges} \,.$$

The first biconditional is by construction of C (the only case in which C returns T is when A does); the second, by specification of A (A is to return T iff the program it is applied to is self-looping).

So, we are forced to reject the supposition that there exists such an A. Technically, we say that the self-looping problem is not semi-decidable. But the fact that no program can answer such a question should not surprise us, any more than the failure of smart humans at the same task.

Programming languages that do not directly support "procedures as parameters" need to use some "code" c as the parameter instead of program C itself, but otherwise the undecidability proof is unchanged:

$$C(c) \text{ returns } T \quad \Leftrightarrow \quad A(c) \text{ returns } T \quad \Leftrightarrow \quad C(c) \text{ diverges} \,.$$

4 The Clones

This copy will outlive the original
and always look young and alive.

—*L'Eve future* (Villiers de l'Isle-Adam, 1886)

Our goal in this section is to demonstrate the impossibility of designing an omniscient robot.

Consider a Model-T android, named Andrea, with the ability to speak, comprehend speech, and react. Any one could pose questions to Andrea, like "Is it raining here, now?". Andrea might answer correctly (by sticking her hand out the window and determining the meteorological state), she might lie (if she is contrary), she might guess and take her chances at being right or wrong (without looking out the window), she might give an inappropriate answer (like, "Shall I get you an umbrella?"), or she may ignore the question and simply stay mum on the subject.

Just as people might question Andrea, other robots might query her. Furthermore, people as well as robots, might ask her questions about herself or about other robots, like: "Are you hungry?"; "Do you fancy Borg?"; or "Is Borg in love with himself?".

The situation can get trickier. Andrea might be programmed to consult her cohorts regarding certain questions. For example, rather than trying to figure out for herself whether Borg is narcissistic, she may be designed to refer such questions to the subject himself. In that case, Andrea will give the same answer to this question as would Borg had we asked him directly (assuming Borg does not formulate his answer based on who is doing the asking). Andrea might turn some questions around before turning to Borg, or might barrage Borg with a series of questions.

Alternatively, Andrea may be smart enough to occasionally detect that Borg is lying, after hearing him explain his answer. So it may be that Andrea gives a different answer than Borg. Still, let's assume that in any such case, where Andrea requests an answer from Borg, but he refuses to answer, she too remains reticent.

Now, hypothesize the existence of a "know-it-all" android, Data. An impossibly self-contradictory situation follows logically from the supposition that such an omniscient, unerring robot is conceivable. If one could construct such a Data, then one could also build a sister robot Echo with design specifications that include the following behavior pattern:

> If anyone asks Echo the abbreviated question, "What about So-and-So?", where "So-and-So" is the name (or serial number) of any robot, then Echo first asks Data (or, better, a built-in homunculus clone of Data) the following roundabout question:
>
> "Does So-and-So answer the question
> 'What about So-and-So?' ?".
>
> Moreover, Echo is quite contrary:

– whenever Data answers "no" to this question, she answers "yes";
– whenever Data answers "yes" to this question, she keeps her mouth shut.

For example, if we ask Echo about Andrea, Echo turns to Data to ask whether Andrea answers the question, "What about Andrea?". Suppose Andrea would answer "no" to this particular question, and Data is smart enough to predict Andrea's answer without even asking. Then Data will answer "yes" to Echo, since Andrea in fact gives a negative answer. Hearing Data's answer to her question, Echo refuses to answer. Echo also keep her mouth shut whenever Data neglects to answer her, but she never answers "no", herself.

The crux of the issue is whether Data (or any other robot) could in fact be all-knowing. To resolve this, consider the specific question "What about Echo?" and imagine that we pose this question to Echo herself! Echo proceeds to ask Data whether or not Echo answers the very same question. Consider all three possibilities:

– If Echo in fact answers "yes" when asked that question, it can only be because Data answers "no" when Echo asks him about her own behavior. But then Data gave the *wrong* answer. He was asked whether Echo answers. She does, but he said she doesn't.
– If Echo does not answer the question, it may be because Data answers "yes", but then again Data got it backwards.
– It may also be that Echo does not answer us, because Data does not answer her. But that means that Data himself does not know the right answer.

The inescapable conclusion is that no robot can be made smart enough to answer such questions: Either Data gives an erroneous answer (our Option II), or else he is dumbfounded (Option IV), just like a human interlocutor in the same situation. The intent of the vague question ("What about So-and-So?") is immaterial.

Of course, bystanders, equipped with hindsight, have no problem giving the correct answer *ex post facto*, as soon as Echo answers—should she altogether. Furthermore, privy to the inner workings of Echo's CPU, and armed with the knowledge that Data is programmed to never lie, no matter what, we can predict the correct answer: Echo will not answer (since Data won't).

5 The Transfinite

To iterate through ordinals requires ordinal notations.
These are notations for computable predicates,
but it is necessary to establish that the computation
really produces a well-founded total ordering.
Thus we need to consider provably recursive ordinals.

—John McCarthy (1999)

In fact, one can build a transfinite series of (ordinal-indexed) programs or robots, each more knowledgable about such matters (self-looping) than its predecessors.

Let $\mathcal{O}$ be any system of ordinal notations (e.g. ordinal diagrams [20] or the recursive path ordering [6]) with programmable ordering $<$, that is, such that the computation of a comparison $\beta < \alpha$ terminates for all $\alpha, \beta \in \mathcal{O}$. Define, for each $\alpha \in \mathcal{O}$:

$$S_\alpha(y) \;:=\; \textbf{if } o_S(y) < \alpha \textbf{ then } T \textbf{ else loop}() \;,$$

where $o_S(y)$ is a pattern-based function that checks if y is a program of the form **if** $\_ < \beta$ **then** T **else loop**(), and returns the upper bound β if it is (and $\mathcal{O}$, otherwise, where $\mathcal{O}$ is bigger than any $\alpha \in \mathcal{O}$, as is customary). Similarly,

$$R_\alpha(y) \;:=\; \textbf{if } o_R(y) < \alpha \textbf{ then loop}() \textbf{ else } T \;,$$

where o_R returns β if y is of this form (or else $\mathcal{O}$).

For all $\alpha \in \mathcal{O}$, we have $S_\alpha(S_\alpha) = \bot$ and $R_\alpha(R_\alpha) \neq \bot$. So, all the S_α are guaranteed sound with respect to the question $X(X) = \bot$, and are complete for $X = S_\beta$, for all β up to (but not including) the ordinal α. Similarly, all the R_α are guaranteed complete (responsive when the answer is in the affirmative), and are sound for all R_β, $\beta < \alpha$. However, for us to be sure that they are correct, we must verify the correctness of $<$ on $\mathcal{O}$.

Despite the fact that S_ω and R_ω have no trouble answering correctly regarding infinitely many programs, there are transfinitely many "better" programs! (Cf. the ordinal-indexed search algorithms of [17].)

6 The Processes

> You must reject the statement I am now making to you
> because all the statements I make are incorrect.
>
> —*The Monkey Wrench* (Gordon Dickson, 1951)

Now we add two new components to the argument, corresponding to the plausible option (III) that an android sometimes just guesses an answer (instead of fruitlessly mulling over the question) and to the remote prospect that some alien androids are not cloneable (Option I).

Five processes will play a rôle:

- R: This (Data-like) process (a.k.a. Roger) is meant to identify some programs X that diverge when fed themselves as input, but is implemented in some undisclosed fashion, say, via quantum wetware. (We are living in a Lisp world wherein programs are their own code.) At this point, we are making no assumptions about R's correctness.
- A: This program (Andrea, say) has the same purpose as R. In Penrose's scenario [15], A incorporates all current, sound scientific knowledge on the subject, but only answers "yes", if it answers at all. It is enough for the argument,

however, to incorporate all of R's knowledge. (Since R's knowledge is presumed to be some finite of "rules", were we able to program all of it in a finite program, a finite set of rules that include only those of R's ideas that are sound would also have to exist as a program.) Again, we will make no *a priori* presumptions about the correctness of A's behavior.

G: This (God-like) entity is our truth yardstick, an oracle that always has the absolute, correct answer to such questions of divergence.

C: This (Echo-like) program applies Cantorian diagonalization to A in the standard fashion so as to produce paradoxical behavior *vis-à-vis* any pretensions of A to know too much.

K: This will be an undisclosed process (in Roger's cerebrum or Data's logic circuitry) used by R to inspect programs like A.

Unlike [15], we will be specializing A and R to deal with divergences (lack of answer) of self-applications $X(X)$, rather than questions regarding more general applications $Y(X)$. This simplifies matters and is all that is, in the final analysis, cogent to the argument.

Let Π denote the set of one-input partial predicates in any standard model of computation (which contains diverging programs and has a conditional construct and subprocedures). By "predicate", we mean that the output is always one of the Boolean truth values, T/F; by "partial", we mean that some inputs may result in no output. As is common, one can enrich the range of a function to include an undefined value $\perp$, denoting the outcome of a never-ending or non-responsive ("I don't know the answer.") process. That is, each $p \in \Pi$ may be viewed as a total function $p : y \mapsto \{T, F, \perp\}$. For example, Π can be the set of one-argument untyped Lisp programs whose range is $\{T, F, \perp\}$ (or a subset thereof).

As explained above, for any particular program $A \in \Pi$, one can construct the following diagonalized program $C_A \in \Pi$:

$$C_A(Y) := \textbf{if } A(Y) \textbf{ then } T \textbf{ else loop}() \textbf{ where } A \textbf{ ...} \ , \tag{1}$$

The input Y can be any program in Π (Borg, say). The behaviors of A and C_A are intimately connected:

- When $A(Y)$ returns T, so does $C_A(Y)$.
- If $A(Y)$ responds F, then $C_A(Y)$ enters an eternal loop.
- If $A(Y)$ does not respond, neither does $C_A(Y)$.

The stated requirement for A is that $A(X)$ answer T when "it" is aware that execution of $X(X)$ is nonterminating ($X(X) = \perp$). In other words, A is sound if $A(X) \Rightarrow X(X) = \perp$. But there is no guarantee that A behaves as expected. Were A to know all there was to know (completeness), that would mean $X(X) = \perp \Rightarrow A(X)$.

On the other hand, $G : \Pi \to \{T, F\}$ is the total predicate,

$$G(X) := [X(X) = \perp] \ , \tag{2}$$

manifesting the truth of the matter. Equality (=) is semantic: both sides must be equally (un)defined.

Now consider some (partial) predicate $R : X \mapsto \{T, F, \bot\}$ with the following behavioral rule:

$$\begin{array}{l} \textit{return } T \textit{ if} \\ \textit{program } X \textit{ is of the form} \\ X(Y) := \textbf{if } Z(Y) \textbf{ then } T \textbf{ else loop}(), \textbf{ where } Z \textbf{ ...} \\ \textit{and} \\ K[Z(X) \neq T] \, . \end{array} \tag{3}$$

Here X, Y, and Z are pattern variables ("placeholders") and $K : S \to \{T, F, \bot\}$ is some partial predicate over statements S. The process K is meant to model whatever thought processes are involved in R's analysis of the question whether $Z(X) \neq T$. Thus, the above behavior is (a special case of) what Penrose believes humans are capable of.

The presumption is that R on input X will, in fact, answer T regarding the divergence of $X(X)$ when and if R "believes it knows"—via process K—that the test $Z(X)$ performed by $X(X)$ does not succeed. Specifically, $R(C_A)$ returns T if $K[A(C_A) \neq T]$ returns T and some other rules of R has not already ventured an answer. On the other hand, R may have various additional considerations that that pre-empt the above behavior and are employed when K responds with F, or when K does not come up with an answer within some reasonable time frame.

7 The Four Sons

> All human errors are impatience,
> a premature breaking off of methodical procedure....
>
> —Franz Kafka (1917)

The following facts are indisputable:

$$A(C_A) = T \quad \Leftrightarrow \quad C_A(C_A) = T \tag{4}$$
$$A(C_A) \neq T \quad \Leftrightarrow \quad C_A(C_A) = \bot \tag{5}$$
$$A(C_A) = T \quad \Rightarrow \quad G(C_A) = F \tag{6}$$
$$A(C_A) \neq T \quad \Rightarrow \quad G(C_A) = T \, . \tag{7}$$

Facts (4,5) follow directly from the references to A in the definition (1) of C: C_A calls A, answers T if A does, and loops, otherwise. Facts (6,7) follow directly from C's behavior and the specification (2) of G: If $A(C_A)$ yields T, then C_A does not diverge (4), and G knows it; if $A(C_A)$ doesn't yield T, then C_A does diverge (5), and again G knows it.

Now, G is infallible and total ($G(C_A) \neq \bot$). Hence (by 6, 7), no A can always be right, whether the result of A, when asked question C_A, is T, F, or $\bot$. That is:

$$A(C_A) \neq G(C_A) \, , \tag{8}$$

which is just a restatement (as in Sect. 3) of Turing's undecidability result for the halting problem. That is, no program $A(X)$ can answer infallibly—for any program X—whether $X(X)$ diverges; specifically, it must trip up with regard to C_A. So, if A happens to agree with R about C_A, then R, too, must not give the textbook answer G.

The upshot of the above facts is that:

$$A(C_A) = R(C_A) \quad \Rightarrow \quad R(C_A) \neq G(C_A) \; . \tag{9}$$

In other words, *if* A simulates R (at least on C_A), then R does not respond properly (T for F, F for T, or $\perp$), while *if* R is averred to never err (precluding both Options II and III), then either $A(C_A) \neq R(C_A)$ (Option I) or else $R(C_A) = \perp$ (as dictated by Option IV). In the last case, R's knowledge is incomplete:

$$A(C_A) = R(C_A) = \perp \quad \Rightarrow \quad \neg K[A(C_A) \neq T] \; , \tag{10}$$

since, were K to have responded, so would have R.

The dichotomy at the heart of the debate is whether there in fact exists a computer program A in Π that agrees with R on C_A, or perhaps there can never be such a program. According to both the strong and weak AI points of view, there exists such a program A that, in particular, agrees with R when queried regarding C_A. But, then, either neither answer, or else both give the same wrong answer. In the latter case, R's error may result either from faulty "reasoning", or from some other cause. It is much like an examinee who, presented with a difficult true/false question, cannot work out the correct answer within the time limit. In this situation, a person may "guess" (using heuristics, perhaps), or may give up and leave the answer blank.

To summarize, we have discerned four characteristics of the nature of R:[3]

I. R **the Wise:** $R \notin \Pi$
(wise, in a super-Turing sense);
II. R **the Wicked:** $K[A(C_A) \neq T] = T$ but in fact $\neg[A(C_A) \neq T]$
(wicked, in that R internalizes an untruth);
III. R **the Simpleton:** $K[A(C_A) \neq T] \neq T$ and in reality $R(C_A) = \neg G(C_A)$
(acting without thinking);
IV. R **the Ignorant:** $R(C_A) = \perp$
(expressing no opinion in the matter).

Using an ostensibly ratiocinative, but fallacious, process (K) is Case II; resorting to an extralogical process is our Case III; not answering is IV. See Fig. 1.

If R gives the wrong answer, it is either due to the above-specified behavior pattern (3), in which case K is unsound (Case II), or else R answers wrongly based on some other consideration or impulse (Case III). In the latter event, K does not respond with T within some allocated time frame, either because its answer is F, or else because it never reaches a conclusion.

[3] The options are evocative of the "Four Sons" of the Passover *Haggadah*, derived from the *Mekhilta*, quoted at the outset.

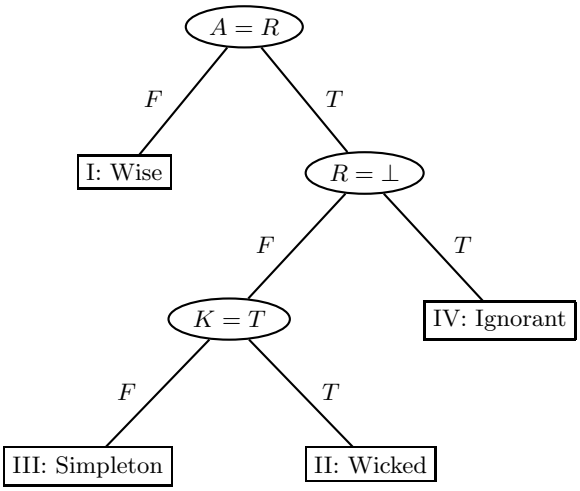

Fig. 1. Possible resolutions

Case I follows from the supposition that there is no $A \in \Pi$ such that $A(C_A) = R(C_A)$. If, on the contrary, $A(C_A) = R(C_A)$ for some $A \in \Pi$, then (by 9) $R(C_A) \neq G(C_A)$, either because $R(C_A)$ gives a wrong answer or yields no answer. With the latter outcome (Case IV), R is not fully (self-) cognizant, since $\neg K[A(C_A) \neq T]$, though in fact $R(C_A) = A(C_A)$ and $R(C_A) \neq T$ (by 10). In other words, R is an incomplete reasoner.

8 The Conclusion

> If a machine is expected to be infallible, it cannot also be intelligent.
> There are several theorems which say almost exactly that.
> But these theorems say nothing about how much intelligence may be displayed
> if a machine makes no pretense at infallibility.
>
> —Alan Turing (1947)

We have skirted the issue of R's being "aware" or "unaware" of its own consistency. We all know (even R's creator claims to know) that R cannot correctly answer all questions involving his *own* consistency, any more than can A. Whether R sees himself reflected in A is beside the point. The more perspicacious question is whether R reasons soundly about one specific aspect of one particular observable program A. Does R (erroneously, perhaps) believe (via K) that he knows how A behaves given program C_A (which, in turn, involves A) as input?

Penrose opts for the "R the Wise" solution, since he believes that R is sound (neither "Wicked" nor "Simpleton") and responsive (not "Ignorant"). He goes on [14, 15] to propose a non-Turing-equivalent model for $R \notin \Pi$. Rejoinders to Penrose along the lines that R represents an "idealized" mathematician agree that such an R cannot be captured algorithmically, but is rather more G-like.[4]

Detractors of Penrose who contend that there may be a program A mimicking R must choose between one of the other three options: R reasons unsoundly with K (II); R feels under pressure and answers using a process other than K (III); or R doesn't answer at all (IV). For example, Hilary Putnam is quoted in [11] as suggesting that humans are inconsistent machines, that is, R ("the Wicked") believes (via K) a falsehood ($A(C_A) \neq T$), our Case II. Similarly, Martin Davis [5] says in response to Penrose: "No human mathematician can claim infallibility. We all make mistakes! So there is nothing in Gödel's theorem to preclude the mathematical powers of a human mind being equivalent to an algorithmic process that produces false as well as true statements."

In frustration at getting nowhere with his sound, cerebral reasoning faculty K, R may blurt out some simplistic—but invariably wrong—answer (Option III). This is how I interpret one of John McCarthy's [12] criticisms: "Much of Penrose's reasoning is nonmonotonic, *e.g.* preferring the simplest explanation of some phenomenon, but his methodology doesn't allow for nonmonotonic reasoning by the program." In other words, there is in fact an A that acts precisely like R and answers incorrectly, for reasons that are non-Aristotelian, but Penrose looks instead at an alternate A' that acts like a hamstrung R for which only monotonic K is consulted. Thus, it is $A' \neq R$, whereas $A = R$.

Lucas [11] does attribute *actual* human foibles to "Simpleton" shortcomings: "Our inconsistencies are mistakes rather than set policies. They correspond to the occasional malfunctioning of a machine, not its normal scheme of operations." Along these lines, most discussions exclude option III as irrelevant when considering "ideal" humans.

Consider arguments such as:

- "Perhaps we are sound, but we cannot know unassailably that we are sound." [4]
- "There is an obvious lacuna: the possibility of a program ... which is not 'simple enough to appreciate in a perfectly conscious' way is overlooked." [16]
- "One can show quite rigorously that Penrose's notion of what it is to know oneself to be sound cannot itself be sound." And, "Humans may be *unable* to know that they are consistent." [10, *emphasis mine*]

[4] "[Penrose] admits that he is talking about an idealized mathematician, not an actual one. It would be a great feat to discover that a certain program is the one that the brain of an actual mathematician 'runs', but it would be quite a different feat to discover that a program is the one that a brain of an idealized mathematician would run." [16].

– "We cannot fully analyze a complicated learning machine, let alone the human mind. Hence, one cannot establish one's own self-consistency." [1, *my translation*]

These quotes do not make it clear what the authors believe human mathematicians do in the face of this lack of soundness/consistency. Some are (perhaps) suggesting that R is "inadequate" and need not have an answer to each and every question. Rather, R—in honest ignorance—does not respond at all to the most vexing of questions, C_A, or perhaps reaches his demise without ever having reached a conclusion. Interpreted thus, they are subscribing to Option IV.

Finally, it is engaging to consider the analogous situation where A is an android (Andrea), designed to parrot R when asked whether C (A's alter ego) diverges on C. That places R in a quandary: Any answer will turn out to be wrong. Whenever someone inquires of C, C consults A, who turns the question over to R. If R says "yes" to A, when asked if C will diverge, then A answers "yes" to C, and C converges, instead. If, on the other hand, R predicts that C will respond, then R says "no", and A also says "no", in which case C cycles, contrary to R's assertion. Thus, the only sound alternative for R, in such a circumstance, would be to "take the Fifth" and avoid perjuring himself.

Paraphrasing Turing [23]:

If a machine is intelligent, it must also be fallible.

Acknowledgement

I thank Udi Boker, Yishai Feldman, and referees for their reactions.

References

1. Arnon Avron. *Mishpete Gedel u-ve'ayat ha-yesodot shel ha-matematikah (= Gödel's Theorems and the Problem of the Foundations of Mathematics)*. Broadcast University, Ministry of Defence, Tel Aviv, Israel, 1998. (In Hebrew).
2. Udi Boker and Nachum Dershowitz. Comparing computational power. *Logic Journal of the IGPL*, 2006. To appear; available at: `http://www.cs.tau.ac.il/~nachum/papers/ComparingComputationalPower.pdf`.
3. David Chalmers, editor. *Symposium on Roger Penrose's Shadows of the Mind*, volume 2. Association for the Scientific Study of Consciousness, 1995. Available at `http://psyche.cs.monash.edu.au/psyche-index-v2.html` (viewed September 2005).
4. David J. Chalmers. Minds, machines, and mathematics: A review of *Shadows of the Mind* by Roger Penrose. *Psyche: An Interdisciplinary Journal of Research on Consciousness*, 2(9), June 1995. Available at `http://psyche.cs.monash.edu.au/v2/psyche-2-09-chalmers.html` (viewed September 2005).
5. Martin Davis. *Engines of Logic: Mathematicians and the Origin of the Computer*. W. W. Norton & Company, New York, 2001.
6. Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, March 1982.

7. Douglas R. Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid*. Basic Books, 1979.
8. Tien D. Kieu. Quantum algorithm for Hilbert's Tenth Problem. *ArXiv Quantum Physics e-prints*, October 2003. Available at `arXiv:quant-ph/0110136`.
9. Tien D. Kieu. Hypercomputability of quantum adiabatic processes: Fact versus prejudices. *ArXiv Quantum Physics e-prints*, April 2005. Available at `arXiv.org:quant-ph/0504101`.
10. Geoffrey LaForte, Patrick J. Hayes, and Kenneth M. Ford. Why Gödel's theorem cannot refute computationalism. *Artificial Intelligence*, 104(1–2):265–286, 1998.
11. John R. Lucas. Minds, machines and Gödel. *Philosophy*, XXXVI:112–127, 1961. Reprinted in *The Modeling of Mind*, K. M. Sayre and F. J. Crosson, eds., Notre Dame Press, 1963, pp. 269–270; available at `http://users.ox.ac.uk/~jrlucas/mmg.html` (viewed September 2005).
12. John McCarthy. Awareness and understanding in computer programs: A review of *Shadows of the Mind* by Roger Penrose. *Psyche: An Interdisciplinary Journal of Research on Consciousness*, 2(11), July 1995. Available at `http://psyche.cs.monash.edu.au/v2/psyche-2-11-mccarthy.html` (viewed September 2005).
13. Allen Newell and Herbert A. Simon. Computer science as empirical enquiry. *Communications of the ACM*, 19(3):113–126, March 1976.
14. Roger Penrose. *The Emperor's New Mind: Concerning Computers, Minds, and The Laws of Physics*. Oxford University Press, New York, 1989.
15. Roger Penrose. *Shadows of the Mind: A Search for the Missing Science of Consciousness*. Oxford University Press, Oxford, 1994.
16. Hilary Putnam. Book review: *Shadows of the Mind* by Roger Penrose. *Bulletin of the American Mathematical Society*, 32(3):370–373, July 1995. Available at `http://www.ams.org/bull/pre-1996-data/199507/199507015.pdf` (viewed September 2005).
17. Edward M. Reingold and Xiaojun Shen. More nearly optimal algorithms for unbounded searching, Part II: The transfinite case. *SIAM J. Comput.*, 20(1):184–208, 1991.
18. John Searle. Minds, brains and programs. *Behavioral and Brain Sciences*, 3:417–424, 1980. Available at `http://members.aol.com/NeoNoetics/MindsBrainsPrograms.html` (viewed September 2005).
19. Warren D. Smith. Three counterexamples refuting Kieu's plan for "quantum adiabatic hypercomputation"; and some uncomputable quantum mechanical tasks. *Journal of Applied Mathematics and Computation*, 2006. To appear.
20. Gaisi Takeuti. Ordinal diagrams. II. *J. Math. Soc. Japan*, 12:385–391, 1960.
21. Boris Tsirelson. The quantum algorithm of Kieu does not solve the Hilbert's Tenth Problem. Available at `arXiv.org/abs/quant-ph/0111009`, November 2001.
22. Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Ser. 2*, 42:230–265, November 1936. Correction in vol. 43 (1937), pp. 544-546. Available at `http://www.abelard.org/turpap2/tp2-ie.asp` (viewed September 2005).
23. Alan M. Turing. Lecture to the London Mathematical Society on 20 February 1947. In B. E. Carpenter and R. W. Doran, editors, *A. M. Turing's ACE Report of 1946 and Other Papers*, volume 10 of *Charles Babbage Institute Reprint Series for the History of Computing*. MIT Press, Cambridge, MA, 1986.
24. Doron Zeilberger. A 2-minute proof of the 2nd most important theorem of the 2nd millennium. Available at `http://www.math.rutgers.edu/~zeilberg/mamarim/mamarimhtml/halt.html` (viewed September 2005), October 1998.