What is this?

This info-mural is one of seven "argumentation maps" in a series that explores Turing's question: "Can computers think and/or will they ever be able to?" Argumentation mapping is a method that provides:
- a method for portraying major philosophical, political, and pragmatic debates
- a summary of an ongoing, major philosophical debate of the 20th century
- a new way of doing intellectual history.

What does it contain?

Altogether the seven maps:
- summarize over 800 major moves in the debates threaded into claims, rebuttals, and counterrebuttals
- 97-130 arguments and rebuttals per map
- 70 issue areas in the 7 maps
- 32 sidebars history and further background

The argumentation maps:
- arrange debate so that the current stopping point of each debate thread is easily seen
- identify original arguments by over 380 protagonists worldwide over 40 years
- make the current frontier of debate easily identifiable
- provide summaries of eleven major philosophical camps of the protagonists (or schools of thought).

How do I get a printed copy?

You can order artist/researcher signed copies of all seven maps from www.macrovu.com for $500.00 plus shipping and handling.

7 Are Thinking Computers Mathematically Possible?

The History and Status of the Debate — Map 7 of 7

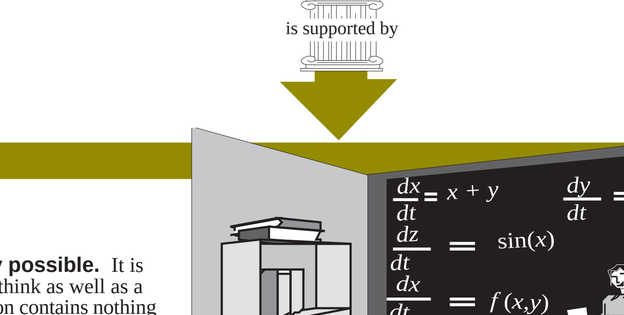
An Issue Map™ Publication

Start Here

1 Alan Turing, 1950
Yes, machines can (or will be able to) think. A computational system can possess all important elements of human thinking or understanding.

is supported by

2
Computer thought is mathematically possible. It is mathematically possible for a computer to think as well as a human can. The mathematics of computation contains nothing to prohibit machines from thinking.



The Background of Gödel's Proof

Context
By the turn of the 20th century a crisis had developed in the foundations of mathematics. The discovery of fundamental paradoxes led to concern about the basic concepts of math and logic. In response to those concerns, mathematicians tried to develop more secure foundational systems (see sidebar, "Formal Systems: An Overview," on this map).

One such system, the *Principia Mathematica* of Bertrand Russell and Alfred North Whitehead, was widely received and provided a framework for subsequent work on the foundations of arithmetic, geometry, analysis, and algebra. Concurrently with Russell and Whitehead, David Hilbert worked on foundational systems in Germany. Hilbert's central idea was that the consistency of mathematics could be shown by a system of *metamathematics*—a system of mathematics *about* mathematics.

The work of Kurt Gödel was situated in this context. He set out to apply the recently developed methods to his own areas of interest, and shortly thereafter discovered the proof for his famous incompleteness theorems.

Method
Gödel developed a numbering system that allowed him to encode formal metamathematical proofs into numeric expressions. The numeric expressions, or Gödel numerals, could be recognized and manipulated *within* the *Principia* system in the same way that any numerals could be (see sidebar, "The Steps of Gödel's Proof," on this map). Gödel numerals allowed the *Principia*—and formal systems generally—to "look at themselves" and say things about themselves. Through this method, Gödel wanted to find out whether the formal system of *Principia Mathematica* could prove itself consistent.

He discovered that no such proof exists. There is no way for the *Principia* to prove itself consistent. Gödel went further and used his method to prove that no complete formalization of arithmetic exists at all (see "Gödel's First Theorem," Box 4). As a corollary, he returned to the consistency question and showed that no consistent formal system of arithmetic could be proved consistent using only its own methods of proof (see "Gödel's Second Theorem," Box 5).

Earlier Precedents
Gödel wasn't the first to suspect that his completeness and consistency results held. Earlier, Finsler (1926) presented an idea similar to Gödel's but without showing how to formalize the argument. Because he did not deal with a specific formal system, Finsler could not present any actual proof of his claim. The American mathematician Emil Post had also proved a result equivalent to Gödel's, but his work wasn't published.

Suggested Reading
Readable discussions of Gödel's theorem include Hofstadter (1978) and Nagel and Newman (1958). Smullyan (1987) teaches Gödel's theorem through a series of puzzles. An introduction with applications to computers is Harel (1987). For historical context, see Rucker (1987), Davis (1965), Dawson (1984a, 1984b), and van Heijenoort (1967). This last volume reprints Gödel's original paper.

The Steps of Gödel's Proof

1 The *Principia Mathematica* gave Gödel a way to translate natural language statements into a formal system of proof. With this formal system in hand, Gödel was able prove that all such systems strong enough to produce arithmetic are inherently limited. Illustration of the proof begins with an English version of what will become the notorious "Gödel sentence."
— "*G* says it is not provable that *G*." ← Sentence in natural language

2 In step 2, the sentence is roughly translated into the formal system of *Principia Mathematica*. The formalization shown here is not well-formed (i.e., is not "grammatically correct"), and consequently the system of proof used in *Principia Mathematica* cannot be applied to it. To make the sentence truly self-referential and well-formed we must first develop a way to make well-formed statements that refer to themselves.
— G <—> ~ prov (G) ← Formalization of Gödel sentence in the formal system (e.g., *Principia Mathematica*)

3 To that end, each symbol in the formal system is replaced with a code number.
— 11 3 1 999 8 11 9 ← Code numbers

4 Those numbers are then used as exponents in a series of prime numbers that will be multiplied together into one large *Gödel number*. This step ensures that every Gödel number corresponds to one and only one formula in the formal system of *Principia Mathematica*.
— $2^{11} \times 3^{3} \times 5^{1} \times 7^{999} \times 11^{8} \times 13^{11} \times 17^{9} = \text{approx. } 2.255092414 \times 10^{881}$ ← Gödel number

5 This number is then represented symbolically, in this case, by a long string of S's followed by a 0 (where the number of S's is equal to the Gödel number). The symbolic "numeral" representation is necessary so that the Gödel number can be dealt with formally in *Principia Mathematica*.
— SSSSSSSSSSSSSSSSSSSSSSSSSS ... SSSSS0 ← Symbolic representation of Gödel numeral

6 This numeral representation for Gödel numbers allows us to use *Principia Mathematica* to "talk about itself." In particular, we can plug the Gödel numeral for *G* back into the formula *G* itself. This generates a well-formed Gödel sentence that makes reference to itself, and that can be assessed using the proof system of *Principia Mathematica*.
— G <—> ~ PROV (SSSSSSSSSSSSSSSSSSSSSSSSSS ... SSSSS0) ← Gödel sentence

7 Now we are in a position to show that the Gödel sentence is neither provable nor disprovable using the system of *Principia Mathematica*. To see how the paradox works, see sidebar, "Self-Referential Paradoxes," on this map.

8 However, the Gödel sentence can be recognized to be true from outside of the system. This is the aspect of the proof that later interpreters would focus on in the context of machine intelligence.

It's so clear from the outside. I can see that *G* is not provable, so the sentence is true.
G <—> ~ PROV (*SSS ... S0*)
I can't decide whether this Gödel sentence is true or not. If it's provable, then it's not provable. But if it's not provable, then it is provable. I'm stuck!
Formal system: *Principia Mathematica*

9 Various authors, notably John Lucas and Roger Penrose, extended the Gödelian insight to computers. They pointed out that because computers are a kind of formal system, the same limitations discovered by Gödel in *Principia Mathematica* and other formal systems might apply to computers as well.

It's so clear from the outside. I can see that *G* is not provable, so the sentence is true.
G <—> ~ PROV (*SSS ... S0*)
I can't decide whether this Gödel sentence is true or not. If it's provable, then it's not provable. But if it's not provable, then it is provable. I'm stuck!
Formal system: computer

Note: A variety of simplifications have been used in this sidebar in order to illustrate in a readable way the general idea behind Gödel's extremely complex proof:
• Gödel does not begin with a non-well-formed formula and then fix it as we have done—this is just a rhetorically useful way to present the proof.
• The Gödel number used for "prov" in step 3 was chosen arbitrarily. Otherwise the number would be extremely long, because "prov" in *Principia Mathematica* builds on various other concepts and definitions with their own code numbers.
• There is a difference between "prov" and "PROV" that is not discussed: "prov" is a property of sentences in the formal system; "PROV" is a property of numerals that represent formulas in the formal system.
• The Gödel sentence does not make reference to itself in the simple way suggested by its informal versions. The numeral for *G*—and the sentence *G* itself—correspond precisely in that there is a one-to-one coding between them, but they do not have the same "meaning." They are "extensionally equivalent" but not "intensionally equivalent."

Self-Referential Paradoxes

The crux of Gödel's proof is a paradox, similar in form to other historical paradoxes about self-referentiality, including the liar paradox, Russell's paradox, and Richard's antinomy (not shown; see Richard, 1905). In each case, a dilemma arises from a self-referential claim.

The Liar Paradox
The liar paradox dates back to the New Testament.

It was a Cretan prophet, one of their own countrymen, who said, "Cretans are always liars, vicious brutes, lazy gluttons"—and he told the truth! (Titus 1:12–13.)
Saint Paul

Sample Liar Sentences
1. This very sentence is false.
2. Sentence 2 is not true.
3. It is true that this very sentence is false.

Either	Or
Assume that the liar sentence is true, *in which case* it is true that "this very sentence is false." So, on this assumption, the sentence is both true and false.	Assume that the liar sentence is false, *in which case* it is false that "this very sentence is false," and so it is also true. So, on this assumption, the sentence is both true and false.
In Either Case	
The liar sentence cannot be pinned down to one truth value. It cannot be decided whether it is true or false.	

Russell's Paradox
In the late 1800s Gottlob Frege introduced a system of logic that seemed capable of providing a consistent basis for mathematics. In a letter to Frege, Bertrand Russell (1902) introduced a paradox that undermined Frege's system and inspired a restructuring in the foundations of mathematics.

To understand Russell's paradox, note that there are some classes that contain themselves and some that do not.
The class *N* of non-red things is itself non-red. This class contains itself as a member. (N) (M) The class *M* of all men is not itself a man. It does not contain itself as a member.

Either	Or
Assume that the class of all classes that do not contain themselves *does* contain itself, *in which case* that class both does and doesn't contain itself.	Assume that the class of all classes that do not contain themselves *doesn't* contain itself, *in which case* the class must contain itself because it is now a member of the class of all classes that do not contain themselves. So again, the class both does and doesn't contain itself.
In Either Case	
The assumption that there is a class of all classes that do not contain themselves leads to contradiction.	

The analogy of this argument with the Richard antinomy leaps to the eye. It is closely related to the "Liar" too. ... We therefore have before us a proposition that says about itself that it is not provable (in PM) (1931, pp. 89–90).

Gödel's Theorem
Gödel sentences generate a paradox similar to the liar's paradox and Russell's paradox, except from the point of view of a formal system such as *Principia Mathematica*.
G <—> ~ prov (G)

Either	Or
It is provable that "*G* is not provable," *in which case* *G* is both provable and not provable.	It is disprovable that "*G* is not provable," *in which case* we can show that it is not the case that *G* is not provable, so *G* is provable after all. So, *G* is both provable and not provable.
In Either Case	
The system can neither prove nor disprove the Gödel sentence *G*. It cannot be decided whether *G* is true or false from the point of view of the system in which *G* is represented.	

Alternative Versions of Gödel's Theorem

Many versions of Gödel's proof have been advanced in the decades since his original treatment. Here are some relatively clear ones, made by writers represented on this map.

There is an algorithm that, given any consistent set of axioms, will output a polynomial equation P = 0, which in fact has no integer solutions, although this fact cannot be deduced from the given axioms (1990, pp. 659–60; also, see "Gödel's Theorem Is Not Decisive," Box 30).
Martin Davis

There exists an algorithm that for any recursively enumerable (r.e.) set of sentences true in the natural numbers produces a true sentence of arithmetic (a "Gödel sentence") not in that set (1990, p. 666; also, see "Penrose Can't Argue For His Hypothesis," Box 28).
Clark Glymour and Kevin Kelly

Unmapped Territory

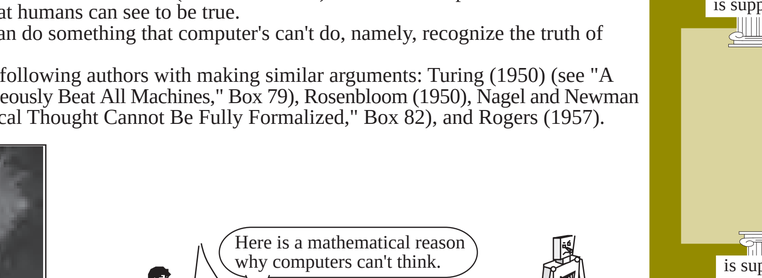
Additional arguments
Because of the technical nature of the arguments on this map, the mapmakers have had to omit many claims and simplify many arguments. Many lines of debate in this field involve sophisticated mathematical symbolism that are difficult to represent in this medium.

Does Gödel's theorem show that machines can't think?

is disputed by

3 John Lucas, 1961
Computers as formal systems are limited by Gödel's incompleteness theorems. Gödel's theorem is the Achilles heel of mechanism. Gödel's theorem proves that a computer cannot, in principle, operate with human understanding. The argument goes as follows:
1. Computing machines are essentially formal systems.
2. Gödel has shown that there are sentences (Gödel sentences) that cannot be proven within a formal system but that humans can see to be true.
3. Therefore, humans can do something that computer's can't do, namely, recognize the truth of Gödel sentences.

Note: Lucas credits the following authors with making similar arguments: Turing (1950) (see "A Human Cannot Simultaneously Beat All Machines," Box 79), Rosenbloom (1950), Nagel and Newman (1958) (see "Mathematical Thought Cannot Be Fully Formalized," Box 82), and Rogers (1957).



is supported by

4 Kurt Gödel, 1931
Gödel's first theorem. Gödel's incompleteness theorem shows that any consistent formal system of axioms and rules of inference, provided it is strong enough to produce arithmetic, will contain true statements that cannot be proven by the procedures provided in the system.
Note: For a brief account of the proof of this theorem, see sidebar, "The Steps of Gödel's Proof," on this map.

is supported by

5 Kurt Gödel, 1931
Gödel's second theorem. As a corollary to Gödel's first theorem it follows that any consistent formal system strong enough to produce arithmetic cannot prove itself consistent.

Mathematical Claims Related to Gödel's Theorem

Church's thesis: This widely accepted (though unproven) claim states that Turing computability, general recursiveness, lambda computability, and any other formulation of effective computability are all equivalent and all adequately describe the intuitive concept of computability.

Church's theorem: First-order predicate logic is undecidable. That is, there is no test to determine the truth or falsity of any arbitrary statement of predicate logic.

Completeness theorem: First-order predicate logic is complete in the restricted sense that all of its (true) statements are provable (but all false statements are not necessarily disprovable). This theorem was first proven by Gödel in 1930.

Tarski's theorem: The concept of truth for arithmetic languages is undefinable. Contemporary proofs of Gödel's theorem often begin with a proof of Tarski's theorem and then derive Gödel's theorem as a corollary.

Turing's theorem: The halting problem for Turing machines is undecidable. That is, there is no single algorithm for Turing machines that allows a machine to halt on an answer to every problem.

Cantor's diagonal theorem: There is no one-to-one correspondence between the real numbers and the rational numbers. This theorem shows that there are distinctly different orders of infinity in mathematics. The diagonalization procedure used by Cantor was later modified by Gödel and used for the proof of his incompleteness theorem.

Note: For further account of these theorems, see Hunter (1973) or Enderton (1972).

Is the Lucas argument dialectical?

is supported by

6
The argument from Gödel's theorem is dialectical. The Lucas argument involves a hypothetical game played between Lucas and a mechanist. A mechanist presents Lucas with a machine model of Lucas's mind. Lucas counters by showing that he can recognize the truth of that machine's Gödel sentence, whereas the machine can't. In this way, Lucas can defeat any of the mechanist's attempts to reduce him to a machine.

Mechanist: You are a machine *M*.
Lucas: No, I'm not, because I can see that the Gödel sentence for *M* is true.
Mechanist: OK, you are a different machine, *M2*.
Lucas: In that case I can produce a new Gödel sentence for *M2*.

Lucas Arithmetic

David Lewis (1969) proposed that the real issue in the Lucas argument is the form of arithmetic that Lucas uses in discussing his own mathematical activity. Lewis dubbed this form of reasoning the Lucas arithmetic.

Lewis defines this arithmetic as the ordinary Peano arithmetic with the addition of an infinitary rule of inference. Peano arithmetic is a formalization of arithmetic developed by Guisseppe Peano in the late 19th century. The system uses a set of 5 axioms to deduce all the truths of ordinary arithmetic.

In ordinary Peano arithmetic, the consistency of the system cannot be proven within the system; this would violate Gödel's second theorem. The addition of an infinitary rule, however, allows us to infer the consistency of a formal system from within that system without violating Gödel's second theorem.

When fully worked out, the Lucas arithmetic includes all of the Gödel sentences of all systems powerful enough to produce arithmetic. As such, it includes an entire hierarchy of systems of arithmetic, each of which contains the Gödel sentences for all of the lower-level systems in the hierarchy.

Arithmetic C / Arithmetic B / Arithmetic A / Gödel sentence for Arithmetic A / Gödel sentence for Arithmetic B

Note: The method of adding an infinitary inference rule to Peano arithmetic was developed by Gerhard Gentzen in 1936.

7 Irving J. Good, 1967
A machine can play Lucas's game. A machine programmed to do transfinite counting could play Lucas's game as well as Lucas can. Lucas implicitly relies on the fact that transfinite counting hasn't been formalized. But the fact that transfinite counting hasn't been formalized doesn't show that humans are any better at it than machines are.
Note: Also, see the "Can improved machines beat the Lucas argument?" arguments on this map, and "The Gödelization Procedure Can Be Algorithmically Specified," Box 52.

trans • fi • nite num • bers: Numbers that go beyond the magnitude of any finite set.

trans • fi • nite count • ing: A form of arithmetic that works with transfinite numbers instead of just with the finite numbers.

is supported by

8 Paul Benacerraf, 1967
The super-mechanist. Even if Lucas can Gödelize any machine that a mechanist can develop, that ability doesn't entail that he can Gödelize any machine whatsoever. A suprahuman mechanist might be able to design a machine that exceeds Lucas's Gödelizing ability.

9 John Lucas, 1967
Good misunderstands the game. Good misunderstands the nature of the game between the mechanist and the mentalist.
• The game is not played with a machine but with the machine's designer. The game is about what the mechanist can do, not about what the machine can do.
• The game is not concerned with showing the superiority of humans over all machines. All the game shows is that for any particular machine the mechanist presents, a mentalist (who knows Gödel's theorem) can show that he or she is not that machine.

10 Irving J. Good, 1967
Gödel's theorem is a red herring. No matter what game we take Lucas to be playing, Gödel's theorem is not the real issue. In fact, Gödel's theorem is a red herring that distracts us from the issue of transfinite counting.
1. We cannot actually know the truth of the Gödel formula; we can only believe it based on our belief that the formal system is consistent (a fact that itself cannot be proved without contradicting Gödel's second theorem).
2. Recognizing the first point, we can see that it was not the Gödel formula that was at the core of Lucas's argument; the real issue was transfinite counting. But Lucas cannot prove that he is better at transfinite counting than a machine is.

Gödel's theorem

red her • ring: An argument that distracts attention from the issue in question. Derives from the use of red herrings (a kind of smoked fish) to distract tracking dogs from the trails of escaped prisoners.

11 Daniel Dennett, 1972
There are no mechanists for Lucas to play with. The dialectical game never gets off the ground because there are no mechanists for Lucas to play with. According to Lucas, mechanists believe that producing true sentences is an activity that can be reduced to finite features and behaviors. But no real mechanist holds this finitistic view of intentional action. All that real mechanists demand is that the mind be deterministic, and for that we do not need to assume that the mind is finite.

12 David Lewis, 1969
Lucas must be able to produce the entire Lucas arithmetic. Lucas's argument requires that a person be able to produce the whole of a "Lucas arithmetic," which includes all of the Gödel sentences of all formal systems powerful enough to produce arithmetic. But Lucas has not shown that it is possible for humans to have this ability.

13 John Lucas, 1970
We don't need the entire Lucas arithmetic. A mentalist doesn't have to produce all of the Lucas arithmetic. It is sufficient that he or she produce enough of the Lucas arithmetic to answer the mechanist at a given step of the game. The success of the Lucas argument must be evaluated in the context of a particular machine being challenged by a particular mentalist.

14 David Lewis, 1969
Machines can't produce the entire Lucas arithmetic. Even though Lucas can't produce the entire Lucas arithmetic, we can still salvage a weaker conclusion from his argument, namely, that machines can't produce the entire Lucas arithmetic either.

15 Hao Wang, 1974
A dilemma about consistency. Lucas's dialectical argument against mechanism runs into a problem about how the mechanist knows whether or not his models are consistent.
Note: Also, see the "Is the use of consistency in the Lucas argument problematic?" arguments on this map.

Either	Or
The mechanist knows all consistent machines, *in which case* the mechanist has a decision procedure for logic. But according to Church's theorem this is impossible. So, the mentalist has no opponent at all.	The mechanist doesn't know all consistent machines, *in which case* the mentalist's ability to refute the mechanist doesn't imply that no consistent machine can prove as much as the mentalist can.
In Either Case	
The mentalist cannot defeat his mechanist opponent.	

16 John Lucas, 1997
The Lucas argument claims less than Wang's dilemma suggests. We can often tell whether or not a machine is consistent, and only those that we know to be consistent are adequate candidates for models of the mind. The mechanist need not know the consistency of all machines in order to know that the ones he presents to the mentalist are consistent.

Does Gödel's theorem show that machines can't be conscious?

17 John Lucas, 1961
Gödel's theorem shows that machines can't be fully conscious. For a machine to understand its own Gödel sentence it would have to be supplied with a consciousness-producing part (e.g., a "Gödelizing operator"). This requirement means that machines can have partial consciousness at best. Humans, by contrast, have full reflective consciousness; they can reflect on themselves as a whole without relying on a special part.

Consciousness-producing part

18 Judson Webb, 1968
Gödel sentences are not self-referential. Gödel sentences don't have anything to do with consciousness, because they are not self-referential (as Lucas claims they are). All that Gödel sentences refer to is their own Gödel numbers.

19 John Lucas, 1971
Gödel sentences are self-referential enough for us to see their truth. Gödel sentences by themselves are not self-referential, but recognizing their truth requires us to see them as self-referential. It is this ability to see Gödel sentences as self-referential that machines lack, and so machines can't recognize the truth of Gödel sentences.

20 Judson Webb, 1968
Machines don't need new parts to incorporate Gödelization. Universal Turing machines don't need new parts to incorporate a Gödelizing operator. All they require is to be refocused on different sets of theorems.
Note: Webb discusses the inclusion of new Gödel sentences rather than "Gödelizing operators," but the point is essentially the same.

21 Judson Webb, 1968
Lucas's interpretation allows for conscious machines. If Lucas's interpretation of the relationship between consciousness and Gödel's theorem is correct, then machines can be conscious after all. All that is required is that:
• They are given the ability to generate their own Gödel sentences.
• They can answer questions about those Gödel sentences.
• Gödel sentences are really self-referential (which they aren't, but which is granted for the sake of argument).

22 John Lucas, 1971
Machines aren't self-critical. Webb focuses on Gödel sentences as a criterion of consciousness. But consciousness should be construed in terms of the ability for self-critical thinking. Self-critical thinking requires a concept of truth. Because machines lack an adequate concept of truth, as Webb admits, they can't think critically in the way that humans can. A machine's inability to recognize the truth of Gödel sentences is just a symptom of its inability to think self-critically.

Does Gödel's theorem show that mathematical insight is non-algorithmic?

23 Roger Penrose, 1990
Mathematical insight is non-algorithmic. Many problems of mathematics (e.g., Gödel's incompleteness problem, the halting problem, etc.) can be understood by conscious humans but cannot be solved algorithmically. This shows that mathematical insight is based on conscious non-algorithmic processes.

24 Daniel Dennett, 1990
Mathematical insight is not the important issue. Even if there is no algorithm for mathematical insight, the lack of one is not crucial because insight is not important to mathematics. Problems like Gödel's theorem and the halting problem can be solved reliably by probabilistic algorithms; whether they are solved by insight or not doesn't matter. Mathematics is grounded in its reliability, not in any particular kind of insight.

25 Jon Doyle, 1990
Mathematical truth may evolve. Penrose assumes that all mathematicians agree on a shared and immutable notion of mathematical truth. But a major school of mathematics, intuitionism, holds that mathematical truth evolves instead. If mathematical truth evolves, then there is no reason to believe that the Gödel sentence generated by a system will still be that system's *own* Gödel sentence at the (later) time when the sentence is evaluated.

Naive Set Theory 1902 — evolved into — Zermelo-Fraenkel Set Theory 1997

26 Adina Roskies, 1990
Penrose's intuitions are irrelevant. The fact that we can't see how algorithms could cause subjective states is irrelevant. We can't see how (following Penrose) quantum wave packets could cause subjective states either. The same problem arises with any physical explanation of subjective states. Not being able to see an explanation doesn't mean that the explanation is incorrect.

27 Roger Penrose, 1990
The absurdity of algorithmic insight. The claim that mathematical insight is algorithmic can be reduced to absurdity.
Assume
There *is* a knowable algorithm (AI procedure) that generates mathematical insight.
It follows that
1. If we were aware of this algorithm, or of how to generate it, then we would have to believe in the soundness of this procedure.
2. Gödel shows how to construct, for any algorithm for mathematical insight, a sentence whose truth follows from the soundness of the algorithm, yet which is inaccessible to that same algorithm.
3. We can understand and believe in the Gödel procedure.
Therefore
There is *not* a knowable algorithm that generates mathematical insight.

28 Clark Glymour and Kevin Kelly, 1990
Penrose can't argue for his hypothesis. No scientist can either establish or refute the hypothesis that mathematical insight is non-algorithmic/noncomputable.
• An empirically decidable property must be "a Δ2 property in the Borel hierarchy." Computability and noncomputability lack that property. Therefore, the hypothesis that mathematical intuition is noncomputable is empirically undecidable.
• Penrose's ability to recognize mathematical truths by insight is consistent both with the hypothesis that insight is algorithmic and with the hypothesis that it isn't. So, Penrose's intuitions fail to establish his hypothesis one way or the other.

29 Roger Penrose, 1990
Glymour and Kelly are too strict. If Glymour and Kelly are right, then it is impossible to verify or refute any scientific theory. But in practice scientific theories can be tested. So it may be possible to test for algorithmicity even if absolute certainty can't be obtained.

30 Martin Davis, 1990
Gödel's theorem is not decisive. The question of whether thinking is algorithmic cannot be decided on the basis of Gödel's theorem.
1. No mathematical insight is necessary to *construct* Gödel sentences.
2. Mathematical insight is involved only in seeing that the system that produces the Gödel sentences is consistent.
3. But, insight into consistency is not reliable (as is shown by numerous historical examples).
4. Because insight into consistency is unreliable, we cannot know whether Gödel's theorem applies to a given system, and so we don't know that mathematical insight is non-algorithmic.
Note: Also, see the "Is the use of consistency in the Lucas argument problematic?" arguments on this map.

31 Roger Penrose, 1990
Insight is essential, even if fallible. Just because insight is sometimes unreliable, we should not conclude that it plays no essential role in mathematics.
• Doubts about consistency in mathematics only arise when mathematicians use systems that go beyond ordinary mathematics.
• When we do become assured of the consistency of a mathematical system, it is always because of insight.

32 Peter Denning, 1990
Algorithms are fixed interpretations. Humans can use conscious observation to step outside of an interpretation and think of alternatives and extensions. Algorithms, however, are inherently fixed interpretations. Thus, no algorithmic account of the mind could ever be sufficient; it could never account for the interpretive flexibility of conscious observation.

Interpretation
I can always step outside an interpretation and think of extensions and alternatives.

Formal Systems: An Overview

David Hilbert laid many of the foundations for what are now known as formal systems. His formalist program aimed to develop axioms that would yield the desired truths of math and geometry, without reliance on intuition. These axioms would be true regardless of whether they were interpreted in terms of points, lines, and planes, or in terms of "law, love, and chimney sweeps."

Formal systems allow mathematical results to be described and assessed with increased precision, and are particularly useful with theories that deal with more than 3 dimensions.

In a formal system, the whole process of proof is reduced to the manipulation of symbols according to rules that completely determine the conclusions that can be drawn from a set of axioms. Intuitions about the "meaning" of the symbols play no role in the workings of the formal system. Such meanings are only introduced with the interpretation of the system.

Below is an example of a simple formal system designed for application to the logic of truth. Note that, in addition to the standard interpretation in terms of truth, there are other interpretations that satisfy the formal system (for more explanation, see "The Lowenheim-Skolem Theorem," Map 3, Box 107).

	Standard Interpretation (TRUTH / FALSITY)	Alternative Interpretation
Axioms		
a —> (b —> a)	If sentence *a* is true, then if sentence *b* is also true, then sentence *a* is still true.	If appleness, then orangeness yields appleness.
(~b —> ~a) —> (a —> b)	If the falsity of *b* entails the falsity of *a*, then the truth of *a* entails the truth of *b* (contraposition).	If non-orangeness yields non-appleness, then appleness yields orangeness.
Rules		
A —> B, A / B	From the truth of "if *A* then *B*," and from the truth of *A*, it follows that *B* is also true.	From appleness yielding orangeness, and from appleness being present, we infer that orangeness is also present.

Notes:
•This formal system is not complete with respect to the logic of truth; that is, it does not yield all valid formulas of the logic of truth. But recall that it does not even make sense to ask whether the system *in itself* is complete or correct; the system in itself is just an orderly set of patterns of symbols with rules to govern their recombination.
•For more on formal systems, see Map 4 and the "Can functional states generate consciousness?" arguments on Map 6.

Can improved machines beat the Lucas argument?

is disputed by

33
Improved machines. A beefed-up machine can recognize the truth of the Gödel sentence. Such a machine defeats Lucas's argument, because it shows that a formal system can evade Lucas's Gödelizing ability.

35 *Anticipated by* C. T. K. Chari, 1963
Self-programming inductive machines beat the Lucas argument. Self-programming inductive machines have enough creativity to recognize the truth of Gödel sentences.
We self-programming inductive machines think.

36 C. T. K. Chari, 1963
Self-programming machines are too rigid to replicate human induction. Self-programming inductive machines only approximate inductive methods. Humans use inductive methods that cannot be formalized. We see this in 2 ways.
1. Human induction is always a creative process that starts with an unspecified set of alternatives. Computers work only with specified alternatives.
2. The concept of probability (which is essential to induction) is flexible and is determined by the use it is put to. Machines are constrained to use probability only in ways that humans determine for them. Machines cannot fit the concept of probability to new situations.
Oh, no you don't.

34 *Anticipated by* John Lucas, 1961
Inductive machines may be immune to Gödelization. Inductive thinking, which humans can do, would allow computers to understand their own Gödel sentences.

37 John Lucas, 1961
A dilemma about inductive machines. Trying to evade the Gödelization problem by making inductive machines results in a dilemma.

Either	Or
The machine acts at random, *in which case* it will not be able to emulate human intelligence.	The inductive machine works according to definite rules, *in which case* it can have a Gödel sentence constructed for it..
In Either Case	
An inductive machine is not an adequate model of the mind.	

38 Frank H. George, 1962
Some AI systems are inductive and probabilistic. The Gödel argument correctly shows that there are limits on deductive machines. But, artificial intelligence deals with inductive and probabilistic mechanisms that Gödel's theorem does not apply to.

39 J. J. C. Smart, 1961
Ingenuous machines could evade the Gödel argument. Machines may have mathematical insight, if they are properly programmed to gauge symmetry and simplicity in patterns of formulae. Such an "ingenuous machine" replicates the abilities of human mathematicians, and so can evade Gödelization as well as any human can.
Now that's symmetrical! 5+7=12 5+9=14

40 *Anticipated by* J. J. C. Smart, 1961
An ingenuous machine is no better than a moronic machine. A moronic machine can't extract itself from the Gödel predicament, even if it is given an infinite amount of time. Neither can an ingenuous machine extract itself, because it only works faster than the moronic machine. An ingenuous machine may display some mathematical insight (that is, it may be able to seek shortcuts to proofs), but it still can't recognize the truth of its own Gödel sentence.

41 J. J. C. Smart, 1961
A self-reflecting ingenuous machine can't be out-Gödeled. An ingenuous machine that can ascertain its own syntax can avoid the Gödel problem. By progressively adding new syntax to its language, an ingenuous machine could understand any new Gödel sentences that Lucas might present it with.

42 *Anticipated by* John Lucas, 1961
A highly complex machine may not be Gödelizable. A qualitative difference in the way computers think may be introduced when they have advanced to a high enough degree of complexity. Such a highly complex machine may recognize the truth of its own Gödel sentence.

43 John Lucas, 1961
A machine complex enough to be un-Gödelizable will not be a machine. By definition, machines behave in a determinate manner according to definite rules. But, any such determinate machine is susceptible to the Gödelization procedure because its behavior can be formalized. Thus, by the definition of a machine, any machine that cannot be Gödelized is not really a machine at all.

44 Albert E. Lyngzeidetson, 1990
A connectionist machine may evade Gödelization. A connectionist machine with massively parallel distributed processing capability may not be susceptible to Gödelization procedures. Such a system could in principle reconfigure its own parameters while in the process of computation and thus arrive at its own semantic metalanguage by inductive means. Once in possession of its own metalanguage, the connectionist machine would be able to evaluate its own Gödel sentence.

45
Self-referential machines. A self-referential machine can evaluate Gödel sentences for itself. Such a machine may evade the Lucas argument.

46 *Anticipated by* John Lucas, 1961
A machine with a "Gödelizing operator" can defeat Lucas's argument. A machine with a Gödelizing operator can carry out the Gödel procedure and add all its Gödel sentences to itself as theorems. Such a self-referential machine would recognize the truth of its Gödel sentence and any subsequent Gödel sentences that could be formed about the machine.

47 John Lucas, 1961
A self-Gödelizing machine can still be out-Gödeled. A machine with a Gödelizing operator is still inadequate. The Gödelizing operator, to be programmable, must be specified by some finite rule. But in that case, the Gödelizing operator is itself formalizable. The resulting system can then be shown to contain a formula that is true but that cannot be proven in the system. So, the Gödelization procedure still holds against the self-Gödelizing machine.

Formalization of the Gödel operator

48 Paul Benacerraf, 1967
Informal proof. A machine could, in principle, construct an informal proof of the truth of the Gödel sentence. So long as the machine does not regard such informal persuasions as proof proper, introducing them into its system will not lead to inconsistency. So, a self-referential machine may recognize the truth of its own Gödel sentence without using a Gödelizing operator.

49 John Lucas, 1988
A machine is not capable of informal proof in the human sense. No matter how informal a machine's reasoning may appear to be, it must still be grounded in a formal system. So the machine's informal "proofs" are also formalizable, and thus subject to the Gödel procedure.

Proposed Model

50 Douglas Hofstadter, 1995
METACAT. The COPYCAT program (see "COPYCAT," Map 1, Box 77) could, in principle, be developed into a new program called METACAT. METACAT would have the reflexive ability to recognize the truth of its own Gödel sentence, and could thereby evade the Lucas argument. METACAT would be able to:
• represent "issues" and "pressures" involved in a problem
• understand how someone else thought up an analogy that didn't occur to it
• store episodic memory of past problems it has solved
• recognize meta-analogies, that is, analogies between different analogies
• construct puzzles based on a sense of "aesthetics."

51 Stuart Russell and Peter Norvig, 1995
The Gödelian insight has already been formalized. Programs have been developed that can derive Gödel's theorems. The "Gödelian insight" has, in effect, been formalized. Penrose neglects this possibility because he fails to distinguish between the formal system within which a Gödel sentence is proven and the system that does the proving.
Note: This claim was originally articulated as an attack on Penrose's response to Boolos (see "The Gödelian Insight Is All That We Need," Box 66). Also, see the "Does Gödel's theorem show that mathematical insight is non-algorithmic?" arguments on this map.

52 Bruce MacLennan, 1990
The Gödelization procedure can be algorithmically specified. The metamathematical Gödelization process can be formalized. It is "meta" in the sense that a formal mathematical process is being used to reason about a mathematical process.

53 Roger Penrose, 1990
The Gödelian insight is a slippery character. The Gödelian insight can attach itself to any system that has been algorithmically specified, including an algorithmic specification of the Gödelization procedure.
Hah! I'm always one step ahead!
Algorithmic specification / The Gödelian insight

Implemented Model

54 K. Ammon, 1993
SHUNYATA. The SHUNYATA system has autonomously developed a diagonalization procedure in its proof of Gödel's theorem. It has also been used to formulate an automatic version of Gödel's proof.

55 Natarajan Shankar, 1994, *as articulated by* Stuart Russel and Peter Norvig, 1995
The proof has been formalized into a program. Using the Boyer-Moore theorem prover, Gödel's theorem has been derived from a basic set of axioms by a computer in basically the same way that Gödel proved the theorem himself.

Implemented Model

56 R. S. Boyer and J. S. Moore, 1979
The Boyer-Moore theorem prover. A LISP-driven theorem-proving engine has been used to derive many novel mathematical results, including decisions on some open questions in mathematics.

Is the use of consistency in the Lucas argument problematic?

57
The problem of consistency. The notion of consistency involved in Lucas's argument runs into difficulties for humans and/or machines.
Note: See "Gödel's Second Theorem," Box 5.
P — Not P

58 *Anticipated by* John Lucas, 1961
Gödel's theorems do not apply to inconsistent machines. Gödel's theorems only apply to a machine if the machine is consistent. But, we can never know for sure whether a given machine is consistent or not. So, we never really know whether Gödel's theorem applies to a given machine.

59 John Lucas, 1961
An inconsistent machine cannot model the mind. Although it is true that Gödel's theorem does not apply to inconsistent machines, that doesn't matter because inconsistent machines can't model the mind. Human minds are geared for consistency. They seek consistency, and use it as a norm for judging which beliefs to accept.

60 John Lucas, 1961
The mechanist's dilemma. The Lucas argument can be restated as a dilemma about consistency. Consider some arbitrary machine *M*.

Either	Or
M is consistent, *in which case* by Gödel's theorem there will be a sentence that humans recognize as true but that *M* cannot prove. So, we can do something that the machine *M* can't.	*M* is not consistent, *in which case* *M* cannot be a mind because minds must be consistent systems.
In Either Case	
The machine cannot be a mind.	

61 G. Lee Bowie, 1982
Lucas can't know when his Gödelization procedure is applicable. For the Lucas argument to work, Lucas must still be able to tell which machines are consistent. But this is mathematically impossible.
Note: For further explanation, see Church's theorem as discussed in Hunter (1973) or Enderton (1972).

62 Anthony Hutton, 1976
Belief in one's own consistency leads to inconsistency. The following argument shows that humans may be inconsistent, and therefore that we can't be sure that Gödel's theorem can be applied to minds.
1. Probabilistic evidence suggests that we have some contradictory beliefs.
2. Rationality demands that we take this probabilistic evidence seriously.
3. So, rationality demands that we think there is some probability that we are inconsistent.
4. So, being certain about one's own consistency (as Lucas claims we can be) is inconsistent with rationality.

63 John Lucas, 1976
There are overriding reasons to regard minds as consistent. Hutton's argument for the inconsistency of the mind is flawed in a number of ways.
• His probabilistic model of the mind is unrealistic. It holds that we accept or reject propositions independently of each other. This is not so.
• An inconsistent model of the mind would affirm every proposition, but no mind would do that.
• We must assume that we are consistent reasoners to be able to start reasoning at all.

64 Douglas Hofstadter, 1978
Inconsistency without explosion of belief. Lucas's argument depends on the assumption that if humans were inconsistent, they would be committed to believing anything and everything. But this conclusion follows from a rule of propositional logic that we have no reason to believe holds for humans. It is quite possible for minds to be both inconsistent and coherent.

65 George Boolos, 1990
We do not know that mathematics is consistent. Gödel's theorem rests on the assumption of a consistent formal system. But many theories in the history of mathematics have been flawed, and even today the best theories are sometimes called into question. This indicates that we do not know, but only hope or believe, that our mathematical theories are consistent, and this in turn means that we cannot "see" with certainty the truth of the Gödel sentence.
Note: This argument is directed against Penrose's version of the argument, rather than Lucas's.
Calculus / Algebra / Set Theory / Topology

66 Roger Penrose, 1990
The Gödelian insight is all that we need. It is not necessary to be able to see the consistency of an entire formal system. The ability to pass from one formal system to the Gödel sentence of that formal system is enough. This kind of Gödelian insight, which is not captured by formal rules, is characteristic of mathematical insight and is non-algorithmic.
Disputed by
"The Gödelian Insight Has Already Been Formalized," Box 51.

Important Properties of Formal Systems

Consistency
A system is *consistent* if it is impossible, within the system, to derive both a statement and its negation. A system is *inconsistent* if a statement and its negation are both derivable.

In an inconsistent system, every possible statement (of its language) can be derived as a theorem, because everything can be logically derived from a contradiction. In a consistent system, that isn't the case.

Consistency is also referred to as the "correctness" or "soundness" of the system.

Completeness
A *complete* system will have a derivable theorem to correspond to every true formula in the language. An *incomplete* system will not be able to derive some true formulae.

Decidability
A system is *decidable* if every true formula of the system has a proof and every untrue formula of the system has a disproof. A system is *undecidable* if there is some statement that it can neither prove nor disprove.

Note: The properties described here are stated in terms of logical systems; they can also be stated for formal systems more generally. For a discussion of formal systems and their properties, see Smullyan (1961). For further discussion of the properties of logical systems, see Hunter (1973), Enderton (1972), or Smullyan (1961).

Other Lucas arguments

67 David Coder, 1969
Lucas's argument does not take into account people who cannot understand Gödel's theorem. Lucas has only shown that someone who understands Gödel's theorem is different from a machine. But, what about a person who cannot see the truth of Gödel's theorem? Lucas has not demonstrated that such a person can outperform a machine, and so his argument fails to establish that machines are essentially different from humans.

68 John Lucas, 1970
A single person's understanding of Gödel is enough. The power of reasoning demonstrated by a single person who understands Gödel's theorem is sufficient to show that minds are different from machines.

69 C. H. Whitely, 1962
Lucas tricks machines into contradicting themselves. Consider the following Gödelian argument, directed against John Lucas himself. Lucas cannot assert that "This formula cannot be consistently asserted by Lucas "without contradicting himself, although we can see that the statement is true. But this argument is simply a way of tricking Lucas into contradicting himself. Similarly, Gödel's theorem tricks machines into contradicting themselves.
This formula cannot be consistently asserted by Lucas.
Ooops! I just contradicted myself.
Lucas

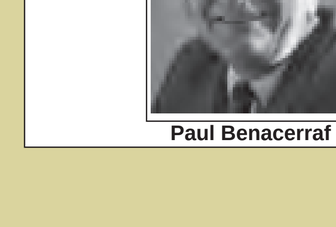
70 Douglas Hofstadter, 1979
Lucas uses a faulty way of identifying differences. To see the flaw in Lucas's argument, imagine a person who thinks his own superiority follows from the uniqueness of his point of view. However, the fact that he can see things in a way that is different from everyone else doesn't prove that he is superior. Anyone could make the same claim about their own point of view. The Whitely sentence reveals this problem of perspective.

71 J. E. Martin and K. H. Engleman, 1990
Lucas can believe his Whitely sentence. The claim that Lucas cannot believe the Whitely sentence is incorrect. Lucas can recognize the Whitely sentence as true, because there is a point of view from which he can understand how the sentence tricks him. From this point of view Lucas can appreciate that he can't assert the sentence—and consequently he can recognize its truth. A machine couldn't do this.
Note: Martin and Engleman address this claim against a modified version of the Whitely sentence, "Lucas cannot consistently believe this sentence," that is credited to Douglas Hofstadter.

72 Daniel Dennett, 1972
Gödel's theorem only limits formal systems, not their machine implementations. The Gödel sentence for some formal system only limits a machine while it is implementing that formal system. But when the machine is implementing some other formal system, the machine may be able to prove the previous system's Gödel sentence. Physical machines, which can implement numerous formal systems, transcend the limitations that Gödel's theorem places on specific formal systems.
Notes:
• Dennett presents this argument as a special case in his broader discussion of how a physical object can implement various Turing machines.
• Also, see the "Is the brain a computer?" arguments on Map 1, the "Can functional states generate consciousness?" arguments on Map 6, and sidebar, "Formal Systems: An Overview," on this map.

73 Thomas Tymoczko, 1990
Highly flexible systems aren't mechanistic. Dennett presupposes that a machine that can effectively take the guise of many different formal systems is a Turing machine. But a machine that generates new programs, treats its machine table as a changing object, and shifts between programs would involve many auxiliary devices that cannot be modeled on a Turing machine. Thus, such a machine is not a mechanistic model of the mind.

74 Paul Benacerraf, 1967
Lucas doesn't recognize what his argument really shows. If we fix up Lucas's argument so that it is more accurate, we discover that it proves something different than Lucas intends. The corrected argument, in turn, shows that:
• Lucas may be a Turing machine. So Lucas's intended argument—that he is not a machine—fails.
• Furthermore, if Lucas is a Turing machine, he has no way of knowing which one he is.
Note: This argument stimulates a highly technical thread of debate that is not represented here because of its length and complexity. This thread includes papers from Hanson (1971), Chihara (1972), and Reinhardt (1986).



75 John Lucas, 1988
I can only be sure of not being any particular machine, if I am not one at all. Benacerraf errs in casting the argument as a nondialectical proof sequence. As a result of this misunderstanding, Benacerraf reaches the incorrect conclusion that the man is a machine but just can't tell which machine he is. But the only way that I can be sure of not being any particular machine is by not being one at all.

76 John Lucas, 1988
Benacerraf's argument is inconsistent. Benacerraf argues inconsistently to his conclusion that I am a Turing machine but cannot prove which one. He says that for each program it can be demonstrated that the program does not represent me, but that still I might be represented by some program. But Benacerraf cannot consistently claim that no particular program represents me and at the same time that some program represents me.

Do mathematical theorems like Gödel's show that computers are intrinsically limited?

77
Mathematical theorems show that machine thought is limited. Gödel's theorem and other mathematical theorems like it reveal essential limitations on the project of making machines that think.
Note: This region covers those arguments that don't derive from Lucas or Penrose but that still deal with Gödelian limitations; that is, with the limitations that Gödel's theorem (and other similar theorems) impose on machine and/or human intelligence.

78 Kurt Gödel, 1951
Machines may eventually have mathematical intuition. The incompleteness theorems only show that a machine cannot be *proven* to possess mathematical intuition. But this does not show that machines can't in fact possess mathematical intuition. To the extent that machines are limited by Gödel's theorems, humans are too. Neither humans nor machines can formulate all of their mathematical intuitions. In this sense, it is impossible for mathematics to be incompletable.
Note: Also, see the "Does Gödel's theorem show that mathematical insight is non-algorithmic?" arguments on this map.

79 Alan Turing, 1950
A human cannot simultaneously beat all machines. Theorems like Church's, Gödel's, and Turing's show only that a human can beat one machine on a given occasion. But there is no reason to believe that a human can out-think all machines. Just because we can think of questions that one machine can't answer, that does not mean that there might not be other machines that will perform better.

80 Emil Post, 1941
Mathematics is an essentially creative activity. The results about incompleteness and undecidability support the idea that mathematics is essentially creative. These results also show that it is preferable to have multiple formal systems rather than a single universal system such as *Principia Mathematica*. The role of logic in mathematics is thereby shown to be one of revealing and developing the limitations of formal systems, not of revealing what the one true formal system is.

81 John Myhill, 1952
Gödel's and Church's theorems are psychological laws. Gödel's theorem shows that human creativity will always exceed human capacity to anticipate that creativity. Furthermore, the theorems also show that humans are able to entertain and clearly conceive of ideas that are neither constructible nor effective.
Note: Myhill's claim is supported by other authors outside of this immediate debate, for instance, by Paul Weiss (1947) and H. Gelanter in personal communication with Myhill.

82 Ernest Nagel and James R. Newman, 1958
Mathematical thought cannot be fully formalized. Gödel's theorem shows that human creativity cannot be fully formalized. The ingenuity of mathematicians in devising new methods cannot be reduced to a precise logical form. For example, it has been shown that humans, using "informal" metamathematical reasoning, can prove theorems that cannot be proven by any formal means.
As Gödel's own arguments show, no antecedent limits can be placed on the inventiveness of human mathematicians in devising new rules of proof (p. 99).
Nagel and Newman

83 Albert E. Lyngzeidetson and Martin K. Solomon, 1994
Open proof systems are not affected by Gödelian arguments. Gödel's theorem implies a sharp distinction between open and closed proof systems. Open proof systems continuously interact with their environment through a steady stream of inputs. An open proof system is, in fact, a potentially infinite set of proof systems, and it has the potential to be noncomputable. Because they are potentially noncomputable, open proof systems are immune to Gödelian arguments, and may turn out to be as creative and insightful as humans are.

closed proof sys • tem: A proof system that has no interaction with the external world and so does not evolve any rules of inference other than those that it started with.

open proof sys • tem: A proof system that continually interacts with its environment through sensors in such a way that it evolves and incorporates stronger and stronger rules of inference in its system.

84 Hilary Putnam, 1960
Proof of human superiority depends on proof of consistency. Newman and Nagel's thesis results from a misapplication of Gödel's theorem. Although it is true (as Newman and Nagel claim) that a machine cannot prove undecidable propositions, a human can't prove those propositions either, unless he or she can first prove that the machine is consistent. But it is unlikely that a human would be able to carry out such a consistency proof given the machine were very complex.
Note: Also, see the "Is the use of consistency in the Lucas argument problematic?" arguments on this map.

85 Thomas Tymoczko, 1990
A machine may be consistent despite lack of proof. To defeat Newman and Nagel's thesis, Putnam must show that the consistency of the machine is *absolutely* undecidable. But there are no absolutely undecidable propositions in arithmetic. The best Putnam can offer is the unlikelihood of being able to show that the machine in question is consistent. But the fact that it is unlikely that we can show that a machine is consistent doesn't mean that a machine is in fact inconsistent.

86 *Anticipated by* J. J. C. Smart, 1961
The argument from Church's theorem. According to Church's theorem there is no decision procedure for predicate calculus. This means that there is no computable procedure by which a machine can decide whether a given sentence in predicate calculus is true or false. Human mathematicians, on the other hand, often decide the truth or falsity of sentences of predicate calculus. Moreover, human mathematicians decide such questions by constructing proofs in a reasonable amount of time and not just at random.

87 J. J. C. Smart, 1961
The ingenuous machine. A machine that is programmed for mathematical insight will decide the truth or falsity of sentences of predicate calculus in the same way that human mathematicians do. Such an "ingenious machine" will have no need to rely on strictly decidable methods of proof.
Note: Also, see "Ingenuous Machines Could Evade the Gödel Argument," Box 39.

88 Dale Jacquette, 1987
Gödel's theorem shows that machines can't understand language the way humans can. Gödel's theorem shows that machines cannot have natural language understanding.
• It is required of speakers of natural language that they be able to distinguish *any* sentence from its negation.
• Machines cannot do this for some sentences, as is shown by their inability to distinguish the negation of a Gödel sentence from the Gödel sentence itself.

Can automata think? 1998

is supported by

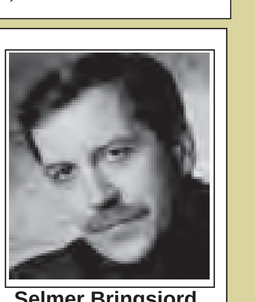
89
Automata can think. A finite automaton, like a Turing machine, can replicate all essential aspects of human intelligence. Finite automata are completely mathematically describable. Thus, machine thinking is mathematically possible.
Note: This region deals with general, mathematical properties of machines, rather than with the specific architectural properties dealt with on other maps. Connectionist networks (see Map 5) and physical symbol systems (see Map 3), for example, are automata, because they implement effective processes that are Turing-computable.
I can think just as well as they can!

90 Alan Turing, 1936
Minds and automata have a finite number of states. At any given time the mind can only possess a finite number of states. If the mind could possess an infinite number of states, some of them would have to be arbitrarily close together, and thus would become confused with each other. So the human mind does not differ essentially from a finite automaton.

91 Kurt Gödel, 1951
Minds are not as finite as Turing believes. At any given time, the mind only possesses a finite number of states, but as time progresses the mind constantly develops. This is shown by the fact that it is always possible for minds to develop new methods of thought. It is likely that the total number of possible mental states involved in developing these methods converges to infinity with time. But, on the other hand, the mind cannot be reduced to a finite automaton.
Note: Gödel claims that Turing's argument only becomes valid under the following two assumptions:
1. There is no mind separate from matter.
2. The brain functions basically like a digital computer.
Gödel believed that the second assumption was true but that the first was "merely a prejudice of our times" that would eventually be disproved (Wang, 1974, p. 326).

92 Arthur Burks, 1973
Human functions can be captured by finite automata. A finite description of a human being can be fine-grained enough to capture all that is essential to its humanity. Once such a description is formulated, a finite automaton can be constructed to duplicate all natural human functions.
There is a limit to the amount of detail needed for a complete specification of natural human functions (p. 56).
Burks

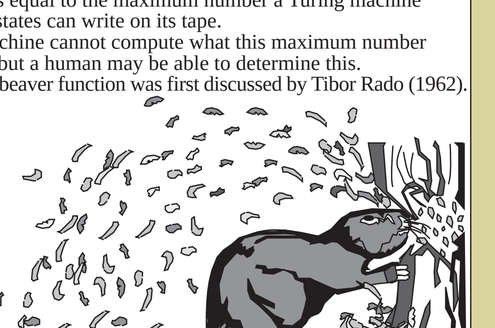
93 Selmer Bringsjord, 1992
Duplication of behavior does not entail computability of internal states. Even if all human functions can be duplicated by an automaton, this does not entail that the internal processes that give rise to those functions can also be duplicated. For example, there is good reason to believe that deliberation is noncomputable, even though the behaviors that result from deliberation can be duplicated by an automaton.



94 Raymond J. Nelson, 1989
Thinking is a rule-following process. Any psychological theory must ultimately decompose psychological phenomena into neural processes that do not require intelligence to implement. Those neural processes can be characterized by Turing-type rules, and hence are computable.
Note: Also, see the "Do humans use rules as physical symbol systems do?" arguments on Map 3, the "Do connectionist networks follow rules?" arguments on Map 5, and the "Is the relation between hardware and software similar to that between human brains and minds?" arguments on Map 3.
IF THEN

95 Selmer Bringsjord, 1992
Nonintelligent procedures can be noncomputable. There are possible procedures that don't require intelligence to carry out, yet that are noncomputable. Imagine that some part of the brain receives electrochemical pulses, and emits pulses in such a way as to instantiate the busy beaver function. The noncomputability of the busy beaver function shows that some physiological procedures may not be computable.

96 Selmer Bringsjord, 1992
The busy beaver function.
1. Given a natural number *n*, the busy beaver function outputs a series of marks.
2. The output is equal to the maximum number a Turing machine with only *n* states can write on its tape.
3. A Turing machine cannot compute what this maximum number of marks is, but a human may be able to determine this.
4. Therefore, humans are not automata.
Note: The busy beaver function was first discussed by Tibor Rado (1962).



97 Raymond J. Nelson, 1989
Many human behaviors can be described algorithmically. Humans have many behaviors in common with computer programs: for example, stimulus-response reactions, doing arithmetic, planning itineraries, doing office work, and so forth. These behaviors can be described algorithmically with recursive functions. Therefore, it is likely that humans are automata.
OUCH!

98 Selmer Bringsjord, 1992
Mathematical language is lacking. In principle, human capabilities (doing arithmetic, etc.) might be translated exclusively into recursive functions. But this begs the question of whether such capabilities can be instantiated by automata, given that we currently have no mathematical terminology to express such capabilities.
There is at this time no way to express such capabilities as writing stories and poems in a logic-mathematical language of any sort (p. 101).
Bringsjord

99 David Cole, 1992, *as articulated by* Selmer Bringsjord, 1992
Brains are automata, therefore persons are too. Because neurons have computable transfer functions, they are a kind of automata. And because brains are collections of neurons, they are automata as well. Our brains are what make us what we are, so persons are automata.
Note: Also, see the "Biological" arguments on Maps 3 and 5.

100 Selmer Bringsjord, 1992
Persons are not brains. The complete argument that persons are automata must include a missing third premise:
1. Neurons are automata.
2. Brains are collections of neurons.
3. Persons are brains.
4. Therefore, persons are automata.
But the third premise, "Persons are brains," is false, so the conclusion that persons are automata doesn't follow.
Note: Also, see the "Can computers be persons?" arguments on Map 1.

101 Selmer Bringsjord, 1992
Humans have an infinite capacity that machines lack. Unlike deterministic automata, persons might be able to have an infinite number of states within a finite period of time. Such an infinite capacity could allow persons to make decisions that automata could never make.
Note: Bringsjord is discussing a possible human capacity, rather than an actual human capacity.

102 *Anticipated by* Selmer Bringsjord, 1992
A finite time period only allows a finite number of states. Only processes that require no time at all could occur infinitely often in a finite period of time. But causal processes occur in space and time, and hence require at least some time (however small it may be). So, it's impossible for infinitely many causal processes to occur in a finite time period.

103 Selmer Bringsjord, 1992
The Zeus machine. It is possible for infinitely many states to take place in a finite time period, as shown by the conceptual possibility of a "Zeus machine." A Zeus machine is an automaton that works faster and faster with each computation it performs. For example, in listing all natural numbers, it takes 1/2 second to list the first number, 1/4 second to list the second number, 1/8 second to list the third number, and so on. In this way the whole infinite list will be completed in only 1 second.
Note: The mention of a machine that can be in an infinite number of states in a finite amount of time doesn't contradict the claim made in Box 101, because a Zeus machine is not a Turing machine or an automaton, but rather an imaginary machine designed to prove a conceptual point.

Turing Machines

The concept of a Turing machine arose in the context of attempts by mathematicians to specify precisely what an algorithm was. Alan Turing's insight was that any algorithm could be carried out by one of a class of Turing machines. Indeed, he proved that an algorithmic procedure (or, an "effective procedure") is just a procedure that can be implemented by a device that blindly and deterministically manipulates symbols. So, Turing machines precisely define the concept of an algorithm.

A Turing machine is conceived of as an imaginary device that manipulates symbols on a tape. The behavior of a Turing machine is determined by the state it is in and by the symbol it reads on the tape. Based on those 2 factors, the machine will enter a new state, write a symbol on the tape, move to the right or to the left, or halt.

The table of rules (or "machine table") correlating these actions with states and symbols exhaustively specifies a given machine. Based on its machine table, we can determine exactly what a Turing machine will do with any given tape.

A "universal Turing machine" is a Turing machine that can perform all the calculations of any other Turing machine. To emulate a given machine, the Universal Turing Machine is "programmed" with a special tape that fully describes the emulated machine's table.

IF the Current State is ...	AND IF the Current Symbol reads ...	THEN do the following to the symbol on the tape and/or halt ...	AND move the tape as follows ...	AND THEN change the Current State to ...
1	0	Do nothing	Right	1
1	1	Do nothing	Right	2
2	0	Erase & write 1	Right	3
2	1	Do nothing	Right	2
3	0	Do nothing	Left	4
3	1	Do nothing	Right	3
4	0	Erase & write 0	No move	4
4	1	Halt	No move	4

Current State: 1
Current Symbol — Scanner component showing Current Symbol on the tape
0 0 1 1 0 1 1

What Is a Machine?

A great many notions of what a machine is are found in the literature. A machine is:
1. Any instantiation of a formal system (Lucas, 1961, p. 44).
2. Anything that can be effectively constructed (George, 1962, p. 63).
3. Anything that operates according to an algorithm (Coder, 1969, p. 235).
4. Anything constructed from "unconnected primordial parts" (Hartmann, 1935, p. 71).
5. Anything equivalent to a Turing machine (various contemporary authors, including Searle, 1991, Nelson, 1989, and Benacerraf, 1967).
6. Anything that can be given a purely geometric description (Spinoza, 1674, p. 129).
7. Anything that behaves according to an unambiguous set of instructions that requires no imagination to follow (Crossley et al., 1972, p. 32).
8. Any device generating a recursively enumerable set of integers (Webb, 1968, p. 158).
9. Any device that is, in principle, divisible into parts (Lucas, 1961, pp. 52–57).
10. Any system whose behavior can be fully explained in terms of proximate causation (Mayr, 1982, pp. 67–70 and 114–16).
11. A system that obeys both principles of physical and chemical causality *and* principles of human functional design (Polanyi and Prosch, 1975, pp. 168–70).

Note: Other notions of what makes a machine are found in the historical literature. See Descartes, Kant, Newton, and La Mettrie, for example.

Is mechanistic philosophy valid?

104
Natural phenomena are best understood in terms of mechanistic principles. The human mind, like all natural systems, is mechanical. Mechanical explanations—like those employed in physics—provide the only necessary foundation for the rest of the sciences, including psychology. Through this extension of the method of physics, mathematics can be brought to bear on psychology and make it a rigorous science.
Note: This region traces the development of mechanistic philosophy from Descartes to the present. Strictly speaking, all the claims on these 7 maps deal with mechanism, because if machines can think then mechanism (in philosophy of mind, at least) is vindicated.
can be explained by — can be explained by

105 René Descartes, 1637
Mechanistic philosophy provides the correct means of investigating the external world. Animate things (plants, animals, the human body, etc.) should be explained according to mechanistic principles, because mechanism shows how bodies interact with each other and with the world. Such principles allow the application of physics to complicated phenomena and leads the way to understanding and knowledge of those systems.
Note: Descartes both supports and disputes mechanism, because on the one hand he thinks bodies are mechanistic, but on the other hand he thinks that minds aren't mechanistic.
Descartes

106 Thomas Hobbes, 1650
Human thinking is computational. Reasoning is purely a matter of abstract computation. Thinking is adding and subtracting, where "adding and subtracting" is extended to apply not only to numbers but also to bodies, proportions, actions, words, motions, conceptions, and so on.
Thomas Hobbes

107 Julien Offray de La Mettrie, 1774
Man is a machine. There is no reason to regard man as anything but a very complicated mechanism. Our own bodies are merely highly complex machines, and so we can, at least in principle, make machines that will recreate our behavior.
Julien de La Mettrie

108 Immanuel Kant, 1790
Mechanism is necessary for scientific understanding. Science proper is only possible in so far as we formulate mechanical laws of nature. Only such laws provide a firm mathematical basis for the construction of theories about the universe. Nature cannot be understood conceptually without determinate mechanical rules.
Note: Kant both supports and disputes mechanism, because on the one hand he thinks that mechanism provides the proper basis for science, but on the other hand he thinks that mechanism can't support critical (i.e., moral and aesthetic) philosophy. See "Mechanism Can't Support Critical Philosophy," Box 114.
Immanuel Kant

109
Mechanistic biology. A school of biology that sought to apply physical laws (e.g., Newton's mechanics) to biological systems by focusing solely on proximate causation.
Note: Members of this school include Julius Sachs, Jacques Loeb, and Ernest Haeckel. The origins of this school can be traced back to Descartes.

prox • i • mate cau • sa • tion: Explanation by proximate causes reveals how the parts of a system work together and in interaction with parts of the environment. Scientists and philosophers working in this framework tend to reject any concern with either the "ancestor causes" of a system (in the sense of evolutionary theory) or with the intentions of a "designer" (in the sense of divine design theory).

110
Vitalism. Vitalism is a school of biology that maintained that biological organisms are too complicated to understand using only mechanical principles. Vitalists posit the existence of a vital force, or *elan vital*, to explain the workings of living organisms.
Note: Members of this school include Hans Driesch (20th century), Claude Bernard, and the German Naturphilosophen school (including Johann Wolfgang von Goethe, Johann Herder, Immanuel Kant, and others).

111
Evolutionary biology. The mechanists' emphasis on proximate causation leads them to ignore the origins of life forms. Their demand for mechanical explanation leads them to underestimate the complexity of biological systems. To understand biology we must consider the origins of species through variation and natural selection.
Note: Members of this school include Charles Darwin, Thomas Huxley, and James Baldwin. Baldwin (1909) makes the argument against mechanistic biology most explicitly.
Charles Darwin

112 René Descartes, 1637
Mechanistic principles cannot explain the mind. The soul and mind are not made of the same kind of substance as external bodies are, and mechanism cannot explain their workings. So, although inanimate objects, plants, animals, and the bodies of humans can be understood by applying physical (mechanistic) laws, the mind of man needs a completely different kind of explanation.
Note: Descartes both supports and disputes mechanism. On the one hand, he thinks bodies are mechanistic, but on the other hand he thinks that minds aren't mechanistic.

113 Benedict Spinoza, 1674
Minds are mechanical. The human mind can be fully explained by mathematical principles. Because the laws of nature are the same everywhere, the same method should be used to study the mind as is used to study bodies. By providing a "geometric" theory of emotions and the mind, human behavior can be shown to be mechanical in the same way that physical bodies are.
Benedict Spinoza

114 Immanuel Kant, 1790
Mechanism can't support critical philosophy. A proper treatment of the teleology (i.e., goal-directedness) of natural systems demands the assumption of a designer with purposes in mind. The assumption of a designer with moral and aesthetic purposes supports the possibility of a critical philosophy, that is, a philosophy that allows us to make value judgments. Because mechanistic explanations tend to ignore the assumption of a purposive designer, the mechanistic philosophy is inadequate to support critical philosophy.
Note: Spinoza is only one of many authors that Kant criticizes with this kind of argument.

115 Immanuel Kant, 1790
The assumption of design is useful for science. Even though the assumption of a designer is ultimately dispensable in science, it still provides a useful guide for scientific explanations. Design provides standards of simplicity, continuity, unity, and organization that have led to fruitful hypotheses in the past.

116 René Descartes, 1637
Machines can't meaningfully use signs. Animals can't use signs as humans do, and animals are essentially machines. So it is implausible that any other machine could learn to use signs as humans do.

117 Julien Offray de La Mettrie, 1774
Animals and machines can use signs. Like humans, animals use signs. Songbirds mock each other, parrots pick up human phrases, apes *could* speak if their vocal chords were different, and so forth. Differences between human and animal use of signs is only a matter of complexity. Similarly, differences between human and machine use of signs is also a matter of complexity. To make machines like humans, then, all we have to do is make them more complex.
Do men and even women make fun of each other better than do the birds who repeat the songs of other birds in such a way as to ridicule them perfectly? (p. 152)
Julien de La Mettrie

118 Gottfried Leibniz, 1714
Artificial machines can never duplicate natural machines. The kinds of machines that humans can construct will never be able to fully simulate the structure of natural machines. In a natural machine each and every part, no matter how small, is tailored to the ends of the whole. Artificial machines, by contrast, always contain some parts that are not tailored to the ends of the whole. Thus, an artificial machine will never be as complex as a living organism.
Gottfried Leibniz

119 Gottfried Leibniz, 1714
The conscious mill thought experiment. If we were to walk into an enlarged machine or mill that could "think, feel, and have perception," nothing observed could explain its conscious qualities, because such qualities would not be found among the parts of the mill.
Note: Also, see "A Conscious Machine Could Not Be Explained By Its Physical Workings," Map 6, Box 55.

120 Gödel's theorem refutes the philosophy of mechanism. The existence of statements that are undecidable by formal systems (e.g., Gödel's theorem) shows that mechanism can't provide a complete explanation of mathematical nature. So mechanism must be rejected as a universal theory of science.
Note: Other Gödelian arguments against mechanism are spread throughout this map. See especially the "Is the Lucas argument dialectical?" arguments on this map.

121 William Nelson Reinhardt, 1986
Strong mechanism is refuted; weak mechanism is safe. Strong mechanism holds that the mind can be *entirely* explained in mechanistic terms. But strong mechanism either contains mentalistic concepts (in its explanatory apparatus) or is inconsistent with Gödel's theorem. Weak mechanism, on the other hand, does not contain mentalistic concepts (because it only explains the concept of mind), and can be shown (via a lengthy proof) to be consistent with Gödel's theorem.
Note: For a similar line of argument directed against philosophical behaviorism, see "Philosophical Behaviorism Is Circular," Map 2, Box 87.

strong mech • an • ism: The mind is mechanical.

weak mech • an • ism: The *concept* of mind is mechanical.

122 Judson Webb, 1968
Gödel's theorem doesn't solve the constructivity problem. If Gödel's theorem really refuted mechanism, it would also show how to solve the constructivity problem in the foundations of mathematics. But Gödel's theorem doesn't show how to solve that problem.

the con • struc • tiv • i • ty prob • lem: The problem of whether or not the foundations of mathematics can be formulated purely constructively, that is, without recourse to assumptions about a Platonic ontology of abstract mathematical entities. If mathematics could be founded on a purely constructional basis, that would mean that it can be constructed "mechanically."

Legend

The arguments on these charts are organized by links that carry a range of meanings:

is supported by — Arguments that uphold or defend another claim. Examples include: supporting evidence, further argumentation, thought experiments, extensions or qualifications, and implemented models.

is disputed by — A charge made against another claim. Examples include: logical negations, counterexamples, attacks on an argument's emphasis, potential dangers an argument might raise, thought experiments, and implemented models.

is interpreted as — A distinctive reconfiguration of an earlier claim.

Focus Box: The lowest-numbered box in each issue area is an introductory focus box. The focus box introduces and summarizes the core dispute of each issue area, sometimes as an assumption and sometimes as a general claim with no particular author.

Arguments With No Authors: Arguments that are not attributable to a particular source (e.g., general philosophical positions, broad concepts, common tests in artificial intelligence) are listed with no accompanying author.

Citations: Complete bibliographic citations can be found in the booklet that accompanies this map.

Methodology: A further discussion of argumentation analysis methodology can be found in the booklet that accompanies this map.

Anticipated by — Where this phrase appears in a box, it identifies a potential attack on a previous argument that is raised by the author so that it can be disputed.

As articulated by — Where this phrase appears in a box, it identifies a reformulation of another author's argument. The reformulation is different enough from the original author's wording to warrant the use of the tag. This phrase is also used when the original argument is impossible to locate other than in its articulation by a later author (e.g., word of mouth), or to denote a general philosophical position that is given a special articulation by a particular author.

Unmapped Territory — Additional arguments — This icon indicates areas of argument that lie on or near the boundaries of the central issue areas mapped on these maps. It marks regions of potential interest for future mapmakers and explorers.