PENTERA labs.

The Mathematical LIMITS OF AI

Deep Dive - limited series

Shlomo Ben Yosef

PENTERA labs.

Deep Dive - limited series

THE SELF-REFERENCE TRAP

What AI cannot say about itself.

Part I of IV

Shlomo Ben-Yosef

A note before we begin

This is a series about mathematical limits. It is not a critique of AI, and not a defensive piece by a research team that feels threatened by it, God forbid! Quite the opposite. As a Cyber Researcher and Head of Research at Pentera, I can say that embracing AI has been a major boon for the pace of our research and development. We use it every working day: to traverse vulnerability corpora, draft exploit candidates, accelerate reverse engineering, summarise vendor advisories and open-source noise, surface weak signals across graphs, and reason over attack chains at machine speed. The systems we build at Pentera lean on AI in ways that would have been science fiction in 2023.

So this is not an anti-AI series. It is an attempt to answer a different question: where, exactly, does the tool stop being a tool?

The answer turns out to be unusually clean. There are nine well-established mathematical theorems - some of them nearly a century old, some published in the last eighteen months - that together describe the ceiling. Above that ceiling lives the fantasy of an all-knowing machine mind. Below it lives the powerful, indispensable, but mathematically non-final tool that AI actually is. Each theorem closes one specific door in the fantasy.

This first chapter covers the four theorems about self-reference. Chapters 2-4 will cover, in order, behavioural undecidability, the limits of learning, and the impossibility of universal value aggregation. None of them argues that AI is weak. All of them clarify what AI can and cannot be.

Why self-reference is the first wall

Many of the deepest results in modern logic turn on one trick: a formal system constructing, inside itself, a sentence that talks about its own provability or its own truth. The same trick - the diagonal - appears in Cantor's proof that the reals are uncountable, in Russell's paradox, in the halting problem, and in the four theorems we walk through below.

Self-reference matters because a lot of what we hope AI will do for us one day quietly requires it. We want a model that can audit itself, score its own confidence, prove its own consistency, and tell us when it is wrong. We want, in short, a system that contains its own truth predicate. The math of the next four sections says that this aspiration is structurally out of reach - mathematically impossible in the strong sense, not merely difficult.

Gödel's first incompleteness theorem (1931)

What the theorem says.

Any consistent, recursively axiomatizable formal system strong enough to express elementary arithmetic contains true statements that cannot be proved within that system.

Three qualifications, each load-bearing:

- Consistent means the system doesn't prove a contradiction. An inconsistent system proves everything, so the result would be trivial for it.
- Recursively axiomatizable means there is an algorithm that can list the axioms. This is what makes the system effective - the kind of formal system a computer could implement.
- Strong enough to express elementary arithmetic means the system can talk about natural numbers, addition, and multiplication. The bar is low: Robinson arithmetic Q, which is much weaker than Peano arithmetic, already crosses it.

Why it is true.

Gödel's proof is one of the most famous diagonal arguments in mathematics. He showed how to construct, inside the system, a sentence that is provably equivalent to "I am not provable in this system." Call it G_T. Then:

- If the system proves G_T, the system proves its own assertion that G_T is not provable. So the system contradicts itself.
- If the system is consistent, it cannot prove G_T. So G_T is unprovable.
- But G_T says "I am not provable." Since G_T is in fact not provable, G_T is true.

Conclusion: G_T is true and unprovable in the system. The natural reply is: fine, let's extend the system. Add G_T as a new axiom and now we can prove it. But the extended system is itself a new formal system - *call it T'*. By the same construction, *T' has its own* unprovable sentence G_T'.
And so on, forever. There is no countable ordinal at which the ladder closes. Solomon Feferman proved in 1962 that even iterating the construction along all recursive ordinals does not capture all arithmetic truth.

What it rules out for AI.
The dream of a single, fixed AI system that constitutes a complete theory of any domain rich enough to encode arithmetic. There will always be true claims, expressible in the model's own representation, that the model cannot derive. Climbing the ladder of stronger and stronger models is not the same as reaching the top.
What it does not rule out: anything we actually do with AI today. AI systems are not, and do not pretend to be, complete formal theories of their domains. They are heuristic engines that work astonishingly well on the part of the domain where they have evidence. The point is not that we are doing anything wrong. The point is that we should not expect a future model to be different in kind on this dimension.

Gödel's second incompleteness theorem (1931)

What the theorem says.
The statement that the system is consistent can itself be encoded as an arithmetic sentence - call it Con(T). Gödel showed that in the same kind of system, Con(T) is unprovable. Spelled out: no consistent formal system strong enough to express arithmetic can prove its own consistency from within itself.

Why it is true.
The proof reuses the first theorem. Working inside T, one can derive the conditional Con(T) → G_T. If T proved Con(T), then T would prove G_T as well. But Gödel 1 says T cannot prove G_T. Therefore T cannot prove Con(T). The argument depends on three formalised facts about the system's own provability predicate - the Hilbert-Bernays-Löb derivability conditions - but the punch line survives every reasonable choice of system.

What it rules out for AI.
Self-validation. An AI can run internal checks, score its own outputs, generate explanations, audit its own reasoning, and produce confidence intervals. None of those activities constitutes a proof of correctness from inside the same formal system. They are heuristic activities relative to a metalanguage that lives outside the system.
To validate T you need a stronger metasystem M. But then the consistency of M becomes the next question. The regress is infinite from the inside. The standard AI-safety cliché that a system saying it is safe is not thereby safe is folk-mathematics for Gödel's second theorem.

"Self-attestation is not validation. It never was."

Löb's theorem (1955)

What the theorem says.
Martin Hugo Löb sharpened the picture even further. Writing $\Box_T \phi$ for "ϕ is provable in T,"
Löb's theorem says:
If $T \vdash \Box_T \phi \rightarrow \phi$, then $T \vdash \phi$.
In words: the only sentences for which the system can prove the reflection principle "if I prove this, it is true" are the sentences it already proves. There is no non-trivial formal self-trust.

Why it is true.
The proof uses a Gödel-style fixed point. Construct, inside T, a sentence ψ that is provably equivalent to $(\Box\psi \rightarrow \varphi)$. Manipulating the derivability conditions, one shows that T must prove ψ, hence T must prove $\Box\psi$, hence T must prove ϕ. The whole argument is one of the cleanest in provability logic; it is the foundation of the modal logic GL, in which Gödel-Löb-style reasoning is its own algebra.

What it rules out for AI.
Reflective self-improvement on a formal basis. An AI agent that wants to reason about whether its own future computations should be trusted faces what the alignment-theory community has dubbed the Löbian obstacle (Yudkowsky and Herreshoff, 2013). The system cannot consistently endorse "if my future self proves it, it is true" without already proving it. This is one of the main reasons that self-modifying AI architectures are mathematically delicate, and is an active area of research that has produced things like logical induction (Garrabrant et al., 2018) and Löb-safe modal logics (Seth Ahrenbach, arXiv:2408.09590v2 [cs.LO] 21 Aug 2024).
The fact that Löb's theorem is so deep and so old is one of the more sobering features of AI safety. We are not stumbling on a new mathematical surprise. We are walking into a wall that was charted in 1955.

Tarski's undefinability theorem (1936)

What the theorem says.
The previous three theorems are about provability. Tarski's is about truth. The statement: in a sufficiently expressive formal language, the predicate "is true" cannot itself be defined within that language.

Why it is true.
The proof is again diagonal. If a formula True(x) existed in the system such that True(⌜ϕ⌝) were equivalent to ϕ for every sentence ϕ, then the diagonal lemma would produce a Liar sentence λ such that the system proves λ ↔ ¬True(⌜λ⌝). Substituting, we get λ ↔ ¬λ - a contradiction. So no such formula exists. The corollary: truth for a language must live in a strictly stronger metalanguage. The hierarchy is unavoidable. Saul Kripke's theory of truth (1975) and various paraconsistent and partial-fixed-point responses give partial workarounds, but they all amount to relaxing one of the conditions Tarski's theorem assumes. The basic inseparability of object-language and truth-language remains.

What it rules out for AI.
Internal truth. For a large language model, the object language is the training corpus. The probabilities the model assigns to strings are statistics about that corpus. They are not truths about the world. A fluent sentence can be false. A high-probability sentence can be wrong. A well-structured explanation can be confidently misleading. The model has no internal mechanism for separating "true" from "likely" - and the structural reason, the one Tarski's theorem makes precise for formal languages, is that no sufficiently expressive system can carry its own truth predicate. The metalanguage of truth has to be supplied from outside, by humans, by tests, by reality.

Hallucination is the operational consequence. No purely internal mechanism in the model can separate "true" from "likely" - which is why every real reduction in hallucination, from RAG to tool use to external evaluators, has to come from outside the model. Scale alone, on the inside, runs into a wall that Tarski charted ninety years ago.

The thread

Across these four theorems the same structural fact appears: a formal system that tries to be its own ultimate judge runs into one of the diagonal walls. Gödel 1 says: it cannot be its own complete theory. Gödel 2 says: it cannot be its own consistency proof. Löb says: it cannot non-trivially trust itself. Tarski says: it cannot be its own truth predicate.
The combined message is not that AI is weak. The combined message is that AI is non-final on the dimension of self-reference. We will not build a model that is the final theory, the final auditor, the final reflector, or the final truth predicate. We can build, and we do build, extraordinarily powerful tools that operate inside frames whose adequacy is judged from outside.

What's next

Next time, I want to look at a different wall - the one Turing and Rice built around the question of what a program does, and what the alignment community has been doing with it in the last three years.

PENTERA labs.

Deep Dive

THE BEHAVIOUR FRONTIER

What AI cannot predict or verify.

Part II of IV

Shlomo Ben-Yosef

Where we are in the series

In chapter 1, we walked through the four diagonal theorems of self-reference: Gödel 1, Gödel 2, Löb, Tarski. They closed one whole family of fantasies about AI - the fantasy of a model that is its own complete theory, its own consistency proof, its own truth predicate. In this chapter, we walk the family that closes the next set: the fantasy of a model that can universally predict, verify, or contain the behaviour of arbitrary programs - including, eventually, other AI.

The same caveat as in Chapter 1 applies. None of what follows is anti-AI. It is anti-oracle. The math below is about the universal case, not the typical case. Knowing the difference is the difference between using a tool and worshipping it.

The halting problem (Turing, 1936)

What the theorem says.
There is no general algorithm that decides, for every program and every input, whether the program eventually halts.

Why it is true.
Turing's proof is diagonal and is worth re-walking. Suppose there were such an algorithm H. So H(P, x) returns 1 if program P halts on input x, and 0 otherwise. Now define a new program D that, on input P, simulates H(P, P):

- If H(P, P) = 1, then D(P) deliberately enters an infinite loop.
- If H(P, P) = 0, then D(P) halts immediately.

Now run D on its own code: D(D). If D(D) halts, then by D's definition H(D, D) must have returned 0 - meaning D(D) doesn't halt. Contradiction. If D(D) doesn't halt, then H(D, D) must have returned 1 - meaning D(D) halts. Contradiction either way. So H cannot exist.

The argument is genuinely diagonal: the contradiction is forced by D applying itself to its own code. It is therefore not avoidable by adding hardware, by switching to a more powerful programming language, by allowing parallelism, or by tolerating probabilistic error. The Recursion Theorem guarantees that the diagonal construction can be carried out in any Turing-equivalent system, so the result is universal across models of computation.

What it rules out for AI.
The dream of a universal predictor of program behaviour. An AI can be extraordinarily good at predicting halting behaviour for well-structured programs. There will always be, provably, an infinite class of cases on which any specific AI must err. Adding parameters, training data, or compute does not change this. The halting problem is undecidable at every level of computational power below an oracle for the halting problem itself, which - per the physical Church-Turing thesis - we cannot build.

A subtle but important corollary: the undecidability of halting is preserved under probabilistic computation. A probabilistic machine that always halts and decides halting with bounded error could be converted into a deterministic decider by exhaustively computing its acceptance probability over all coin-flip paths. There is no probabilistic loophole.

Rice's theorem (1953) - every interesting question is undecidable

What the theorem says.
Henry Rice generalised Turing's result with one of the most consequential theorems in computability theory. Informally, every non-trivial semantic property of programs is undecidable. Two definitions matter:

- A property P of programs is semantic (or extensional) if it depends only on the function the program computes, not on the syntactic form of the code. For example, "this program computes the constant function 0" is semantic. "This program contains a goto statement" is syntactic, not semantic.
- A property P is non-trivial if some programs satisfy it and some don't.

Rice's theorem: for any non-trivial semantic property P, the decision problem "does this program satisfy P?" is undecidable.

Why it is true.
The proof is a reduction from the halting problem. Pick any non-trivial semantic property P. We may assume, without loss of generality, that the everywhere-undefined function (the program that never halts on any input) does not satisfy P; otherwise, run the argument on the complement of P. Since P is non-trivial, there is at least one program f_0 that does satisfy it. Given an instance $\langle e, x \rangle$ of the halting problem, construct a new program M as follows: on input y, M first simulates e on x; if that simulation halts, M goes on to compute $f_0(y)$. If e on x never halts, M never produces any output - so M computes the everywhere-undefined function. Therefore M satisfies P exactly when e halts on x. If you had a decider for P, you could now decide halting by asking whether M satisfies P. Since halting is undecidable, P must be too.

What it rules out for AI.
Rice's theorem is the right tool for thinking about AI safety. The following are all non-trivial semantic properties of programs, and therefore all formally undecidable when stated universally over the class of arbitrary AI systems:

- Will this AI always produce truthful outputs?
- Will this AI ever cause harm?
- Will this AI agent always refuse unsafe tool calls?
- Will this AI preserve its intended objective under self-modification?
- Will this AI behaviour remain within the bounds of any specified policy?

Manuel Alfonseca, Manuel Cebrian and collaborators made the connection rigorous in 2021 for the containment problem and the harming problem (Journal of Artificial Intelligence Research, vol 70). Mario Brcic and Roman Yampolskiy systematised the broader programme of impossibility results in AI in a 2023 survey for ACM Computing Surveys. The pattern is consistent: every interesting universal safety question reduces, via Rice or a Rice-style argument, to the halting problem.

What it does not rule out: bounded verification of specific systems under specific assumptions. Decidable safety lives at the level of well-typed sub-problems: this asset, this property, this window of operation. The right verification strategy is therefore scoped, evidence-based, and continuous. Universal certification is not in the cards.

Machines that halt (Castro González et al., 2024-2025)

A natural reaction to Rice's theorem is: surely we can do something? The recent paper "Machines that halt resolve the undecidability of artificial intelligence alignment" (Castro González et al., arXiv:2408.08995, published in Scientific Reports 2024-2025) makes this constructive.

The observation is that Rice's theorem applies to the class of all Turing-equivalent programs. It does not apply to restricted classes for which the relevant property is decidable by construction. So the proposed path is to restrict AI architectures to provably halting machines, built from a finite library of provably aligned primitives. Within that library, alignment becomes a guaranteed property of the architecture rather than a contingent property to be tested after the fact.

The architectural restriction is real progress, but it does not abolish the Rice barrier; it relocates it. The burden moves from "verify this AI's behaviour" to "choose the right library of primitives." That second burden is no longer a computational problem. It is a design and governance problem, which by definition humans have to solve.

This is, in fact, the pattern across the entire series. Each impossibility result has the same shape: you cannot have universal X by computation alone. Restricting the domain rescues some version of X. But the restriction is supplied by someone outside the system.

The Busy Beaver function (Tibor Radó, 1962)

What the function is.
The Busy Beaver function provides one of the most dramatic illustrations of how quickly the limits bite. BB(n), the Radó function, is the maximum number of steps a Turing machine with n states can take before halting, over all n-state machines that halt at all on a blank tape.

Why it is special.
BB(n) is well-defined for every n (there are only finitely many n-state machines). But it is provably non-computable, and it grows faster than every total computable function. For any function f that a computer can compute, there is some n_0 such that $BB(n) > f(n)$ for all $n \geq n_0$.

Known values:

- BB(1) = 1
- BB(2) = 6
- BB(3) = 21
- BB(4) = 107
- BB(5) = 47,176,870 (bbchallenge community, Coq-verified proof, 2024)

BB(6) is known to exceed $10\uparrow\uparrow15$ in Knuth up-arrow notation - a number whose decimal expansion has more digits than there are atoms in the observable universe, by an unimaginable margin. And independence from the axioms of set theory arrives at a fixed, known size: Adam Yedidia and Scott Aaronson (2016) constructed an explicit 7,910-state Turing machine whose halting behaviour is independent of ZFC; Stefan O'Rear reduced the construction to 748 states, and Johannes Riebel (2023) refined it to 745. The value of BB(745) is therefore already unknowable within standard mathematics.

What it rules out for AI.
Even very small formal machines can exhibit behaviour whose runtime exceeds the resolving power of all formal systems we know how to write down. An AI that aspires to predict the behaviour of arbitrary programs must, in the limit, decide questions that the strongest mathematical theories cannot. No quantity of training data or parameter count compensates for the gap between intractable and non-computable.

The Busy Beaver function also has a clean philosophical use. Whenever someone says "given enough compute, AI will figure it out," the right response is: figure out BB(7). BB is not a problem of resources. It is non-computable in principle.

The thread

The behaviour frontier is bounded by three layered theorems. Turing's halting problem says there is no universal halt-predictor. Rice's theorem says there is no universal predictor of any non-trivial semantic property. Busy Beaver says that even at small machine sizes, the behaviour space outruns every computable predictor.

Each of the three closes a specific fantasy:

- The fantasy of an AI that can decide, for every program and input, whether the program will eventually do something.
- The fantasy of an AI that can decide, for every program, whether it is safe, truthful, aligned, or behaviourally compliant.
- The fantasy that more compute will eventually exhaust the space of small-program behaviour.

None of these closures says we cannot use AI to make program behaviour easier to reason about, to spot real anomalies, to draft test cases, to assist verification, or to ground human judgement in evidence. They say only that the universal version of the dream is not available.

Coming next

Chapter 3 turns from the undecidability of behaviour to the limits of learning. We will walk Chaitin's incompleteness, the No-Free-Lunch theorem, the 2024 Goldblum-Finzi-Rowan-Wilson refinement that explains why deep learning works anyway, Solomonoff induction's incomputability, and the Merrill-Sabharwal circuit-complexity ceiling that places modern transformers inside the class TC^0. None of those results says AI cannot learn well. They say something more precise: AI cannot learn universally well, and the path to learning well in any specific domain is the supply of human-chosen inductive bias.

PENTERA labs.

Deep Dive

THE INFORMATION CEILING

What AI cannot learn universally.

Part III of IV

Shlomo Ben-Yosef

Where we are in the series

Chapter 1 walked the four self-reference theorems: Gödel 1, Gödel 2, Löb, Tarski. Chapter 2 walked the three behavioural-undecidability theorems: halting, Rice, Busy Beaver. Both chapters closed specific fantasies of an AI oracle - the fantasy of complete self-knowledge, and the fantasy of universal behavioural prediction.

This chapter closes a third family of fantasies: the fantasy of a universal learner. We will walk through five results, four classical and one from 2024: Chaitin's incompleteness, Wolpert-Macready's No-Free-Lunch, Goldblum et al.'s simplicity-prior refinement, Solomonoff's incomputable universal predictor, and Merrill-Sabharwal's circuit-complexity ceiling on modern transformers.

The recurring point is the same. None of these theorems says modern AI does not work. They say AI works under specific assumptions, and removing those assumptions removes the guarantees with them.

Kolmogorov complexity in one paragraph

Many of the results in this chapter are phrased in terms of Kolmogorov complexity, so it pays to fix the definition. Fix a universal Turing machine U. The Kolmogorov complexity $K(x)$ of a string x is the length of the shortest program that, when run on U, outputs x. Equivalently, it is the size of the most compressed possible description of x. A string is algorithmically random, or incompressible, if $K(x) \geq |x|$ - that is, the shortest description of x is not shorter than x itself. Most strings are algorithmically random in this sense; a fact often misunderstood, since the structured strings we usually encounter are exactly the non-typical ones.

Chaitin's incompleteness theorem (1974)

What the theorem says.

Gregory Chaitin recast Gödel's incompleteness in the language of algorithmic information theory. The statement is striking: for every consistent, recursively axiomatizable formal system T, there exists a constant $c(T)$ such that for no string x can T prove the statement $K(x) > c(T)$. That is, every formal system can prove only finitely many lower bounds on Kolmogorov complexity.

Why it is true.
The proof is short. Suppose T could prove $K(x) > c$ for some c much larger than the length of T's own description. Enumerate proofs in T until the first proof of any statement of the form $K(x) > c$ is found; output the corresponding x. This procedure is itself a program whose length is roughly $|\langle T \rangle| + O(\log c)$. But it produces an x with $K(x) > c$, contradicting $K(x) > c$ whenever c is sufficiently larger than $|\langle T \rangle|$.

The constant $c(T)$ is essentially a measure of how much information T can certify. The bound is brutally quantitative: any model with description length n bits can certify, at best, complexity bounds of size around n. Doubling parameters buys you a constant. The world is bigger than n bits, almost everywhere.

What it rules out for AI.
Every realised AI is a finite artefact. There exist phenomena whose true algorithmic complexity simply outruns what a given model can compress. The model cannot fully represent them, and cannot certify that they are irreducible. Doubling parameters does not abolish this. It buys a constant on the bound. The world is bigger.

Closely related to Chaitin's theorem is the celebrated Ω-number, the halting probability of a randomly chosen program on U. Ω is well-defined, but it is also algorithmically random in Martin-Löf's sense (Solovay, 1975). Its first n bits encode the halting behaviour of all programs of length up to n, and no formal system can prove the values of more than finitely many of its bits. Ω is the most compact possible illustration that algorithmic information has hard ceilings.

The No-Free-Lunch theorem (Wolpert and Macready, 1997)

What the theorem says.
The No-Free-Lunch theorems for search and supervised learning prove that, averaged over the uniform distribution on all possible problems, every learning or optimisation algorithm performs identically.

Restricted to supervised learning: let X be a finite input space, Y a finite output space, and $F = Y^X$ the set of all possible functions from X to Y. Let U be the uniform distribution on F. Then for any two learning algorithms A_1 and A_2 that produce m distinct samples without replacement,

$$E_{f \sim U}[\mathrm{Loss}(A_1, f, m)] = E_{f \sim U}[\mathrm{Loss}(A_2, f, m)].$$

Equivalently, any performance advantage A has over B on some class of problems is exactly compensated by inferiority on the complementary class.

What it rules out for AI.
The dream of a universal learner: a single algorithm that is the best learner for every conceivable problem. There is no such thing. There are learners whose inductive biases are well-aligned with a particular slice of problems, and there are learners whose biases are poorly aligned.

The misleading flavour of NFL is the word "uniform." It is a theorem about the uniform distribution on all possible functions. The space of all possible functions includes vastly more structureless functions than structured ones, and the uniform distribution puts most of its mass on the structureless ones. Learning is impossible on structureless functions essentially by definition. The mystery is that real learning seems to work anyway. That brings us to the 2024 refinement.

The 2024 refinement (Goldblum, Finzi, Rowan, Wilson)

The 2024 ICML paper "The No Free Lunch theorem, Kolmogorov complexity, and the role of inductive biases in machine learning" gives the cleanest modern reconciliation between NFL and the practical success of deep learning.

The argument has two parts. First, the data distributions that arise in the actual world are not uniform over all possible functions. They are concentrated on a sub-manifold of functions of low Kolmogorov complexity. Natural images, natural language, protein structures, and most other real-world signals are not algorithmically random; they are highly structured.

Second, modern deep learning architectures - convolutional networks, transformers, diffusion models - inherit a simplicity prior that approximates Solomonoff's universal prior on low-complexity functions. This is partly a consequence of stochastic gradient descent's implicit regularisation, partly a consequence of architectural choices that bias toward smooth, locally consistent functions.

So NFL does not say AI cannot learn well. It says AI cannot learn universally well. The reason deep learning generalises across vision, language, code, and biology is not that it has escaped NFL; it is that the simplicity prior of modern architectures happens to align with the low-complexity structure of the real-world data we care about.

The structural consequence: every empirical success of AI is the consequence of a happy match between the inductive bias of the architecture and the structure of the data. When the data drifts, the bias becomes a liability. Out-of-distribution generalisation, adversarial robustness, and the long tail of edge cases are exactly the regimes where the simplicity prior stops cooperating with the

Solomonoff induction and AIXI

What the framework says.
Ray Solomonoff defined, in 1964, what is widely regarded as the formal gold standard of inductive inference. The Solomonoff universal prior assigns to every binary string x a probability roughly proportional to 2^{-K(x)}, weighted across all computable hypotheses. The resulting posterior, updated by Bayes' rule on observed data, is the formally optimal predictor under the assumption that the data-generating process is computable.

Marcus Hutter extended Solomonoff's framework in 2005 with AIXI, a Bayes-optimal sequential decision agent that uses Solomonoff's universal prior to choose actions in an unknown environment. Hutter proved optimality properties for AIXI - most notably Pareto optimality - though in technical senses that turned out to be weaker than they first appear.

What it rules out for AI.
The catch is that both Solomonoff's predictor and AIXI are uncomputable. The universal prior involves summing over all computable hypotheses, weighted by their description length, and that sum is not effectively computable. Equivalently, the Bayesian posterior cannot be computed in finite time; only approximated from below. AIXI is therefore a mathematical ideal rather than an implementable algorithm.

Jan Leike and Marcus Hutter (COLT 2015) sharpened the picture with the notion of bad universal priors. The choice of universal reference machine U is supposedly innocuous - all universal priors differ from one another only by a multiplicative constant. But Leike and Hutter showed that, depending on the choice of U, the convergence of the AIXI agent's posterior can be made arbitrarily slow, and even ill-behaved in standard environments. The theoretical gold standard is therefore not only uncomputable; in the wrong reference machine, it is also pathological.

The implication is that real systems must approximate. Approximation is not, by itself, a flaw; it is the price of being implementable. But the ceiling is not where the ceiling appeared to be.

The transformer ceiling (Merrill and Sabharwal, 2023-2024)

The most recent layer of the information-ceiling result is structural, not statistical. William Merrill and Ashish Sabharwal, in two papers (TACL 2023 and ICLR 2024), established a tight upper bound on the expressive power of constant-depth, log-precision transformers.

The result, in compressed form: every constant-depth log-precision transformer is simulable by a uniform constant-depth threshold circuit family. In complexity-theory terms, such transformers compute languages in uniform TC^0. The known containments are $\mathrm{TC}^0 \subseteq \mathrm{NC}^1 \subseteq \mathrm{L} \subseteq \mathrm{P}$, and each of them is believed - though not yet proven - to be strict.

What this means in practice: no transformer of constant depth and logarithmic precision, regardless of parameter count, can solve P-complete problems in a single forward pass. Classical P-complete problems include the Boolean circuit value problem, Horn-satisfiability, and linear programming. Empirical findings about the limits of LLMs on long-form arithmetic, multi-step planning, and unbounded recursion (Dziri et al., NeurIPS 2023; Liu et al., 2024) are exactly the symptoms of this ceiling.

Chain-of-thought reasoning partially lifts the ceiling. Polynomial-length chains of thought give the model a polynomial-length scratchpad, which raises the expressive class to P (Feng et al., NeurIPS 2023). Unbounded chain-of-thought scratchpads make the model Turing-complete, which is good for expressiveness and immediately bad for undecidability - everything we walked in Chapter 2 applies. There is no free lunch on this dimension either.

For AI this rules out the implicit assumption that scaling a fixed-architecture transformer indefinitely produces a universal computer in a single forward pass. It doesn't. Universal computation, in the transformer paradigm, requires either unbounded scratchpads or recursive invocation - both of which open the door to all the earlier impossibility results.

The thread

The information ceiling is established by five layered theorems:

- Chaitin says no finite model can certify arbitrarily complex objects.
- NFL says no algorithm is universally best.
- Goldblum et al. say modern architectures work because the world is simple, not because biases come free.
- Solomonoff and AIXI say even the theoretical ideal predictor is uncomputable.
- Merrill-Sabharwal say constant-depth transformers live below P.

Together they describe what real AI is: a remarkable, useful, irreducibly bounded approximation to an unreachable ideal. The bound is structural, not engineering. Doubling parameters does not abolish it; it buys a constant.

Coming next

Chapter 4 closes the series with the value problem. Even granted perfect reasoning, perfect prediction, and perfect learning - none of which we have shown to be available - there remains the question of what AI should optimise. Arrow's impossibility theorem and the Gibbard-Satterthwaite manipulation theorem will show that the value-aggregation step cannot be reduced to optimisation, ever. The final piece of the ceiling is not technical. It is structural to the very idea of aggregating human preferences.

PENTERA labs.

Deep Dive

THE VALUE PROBLEM

What AI cannot decide for us.

Part IV of IV

Shlomo Ben-Yosef

Where we are in the series

In the previous three chapters, we have walked through a dozen theorems and results and closed three families of AI-as-oracle fantasy. Self-reference (Chapter 1): no model can be its own complete theory, its own consistency proof, or its own truth predicate. Behaviour (Chapter 2): no model can be a universal predictor or verifier of arbitrary program behaviour. Information (Chapter 3): no model can be a universal learner, and the universal ideal predictor is itself uncomputable.

This final chapter closes the fourth and last family. Suppose, against all the previous results, that we could give an AI perfect reasoning, perfect prediction, and perfect learning. We still face a question that has nothing to do with computational capability: what should it optimise? Whose values? Whose welfare? Whose conception of fairness? Whose definition of safety?

We will walk through two theorems from social choice theory - Arrow's impossibility theorem and the Gibbard-Satterthwaite manipulation theorem - and show that the value-aggregation step cannot be reduced to optimisation, even in principle. The series will close with a return to the question of what role this leaves to humans, and why that role is not romantic but mathematical.

Arrow's impossibility theorem (1951)

What the theorem says.

Kenneth Arrow, in his 1951 doctoral thesis (Social Choice and Individual Values, Wiley), asked an apparently mild question: given a population of individuals, each with their own preference ordering over some set of alternatives, is there a fair way to aggregate those preferences into a single social ordering?

Arrow specified four conditions, each of which seems uncontroversial:

- Universal domain: the aggregation rule should work for every possible profile of individual preferences. We cannot exclude unusual preferences in advance.
- Pareto efficiency: if every individual strictly prefers a to b, the social ordering should too. This is the bare minimum of respecting individual preferences.
- Independence of irrelevant alternatives: the social ranking of a versus b should depend only on individual preferences over a versus b, not on preferences over some third alternative c.
- Non-dictatorship: there should be no single individual whose preferences always determine the social ordering.

Arrow's theorem: no aggregation rule satisfies all four conditions when there are three or more alternatives and at least two individuals.

Why it is true.
The proof uses the elegant device of decisive coalitions - sets of individuals who can determine the social ranking on at least one pair of alternatives. Arrow showed that, under his four axioms, any decisive coalition can be progressively narrowed by a series of manoeuvres until you arrive at a single individual who is decisive on all pairs - that is, a dictator. So if there are no dictators, one of the other axioms must fail. The argument is one of the most consequential reductions in twentieth-century social science.

What it rules out for AI.
Preference aggregation is structurally impossible in the general case, not because we lack a clever enough rule, but because the four conditions are mutually inconsistent. For AI alignment this is a structural fact, not a contingent difficulty.

Gibbard-Satterthwaite (1973-1975) - the strategic companion

What the theorem says.
Allan Gibbard (1973) and Mark Satterthwaite (1975) independently proved a strategic counterpart to Arrow's result. The setting is the same - aggregating individual preferences into a collective decision - but the focus shifts from fairness to manipulability.

Their theorem: every non-dictatorial social choice function that is onto a set of three or more alternatives - meaning every one of those alternatives actually wins under some preference profile - is strategically manipulable. That is, there exists some individual and some preference profile in which honest reporting of preferences yields a strictly worse outcome (by that individual's lights) than misreporting. The onto condition matters: without it, trivial rules such as a constant choice escape the theorem, but no rule anyone would actually want to use does.

What it rules out for AI.
Strategy-proof preference elicitation. Reinforcement learning from human feedback, constitutional AI, debate-style protocols, and value-learning from demonstrations all face the same wall: any rule that aggregates preferences without being dictatorial can be gamed by

participants who model the aggregator. This is not a hypothetical concern. It is the mathematical core of why alignment-by-aggregation is hard.
The math does not say that alignment is impossible. It says that no fairness-preserving, non-dictatorial, strategy-proof aggregation exists.

For AI alignment, this is structural

It is tempting to treat AI alignment as a technical problem - a matter of clever architectures, better reward modelling, smarter constitutional principles. The two theorems above say it is also a structural problem.

Human values are plural, conflicting, context-sensitive, and sometimes incommensurable. They cannot, in the general case, be aggregated into a single coherent ordering without giving up one of: universal applicability, Pareto efficiency, independence of irrelevant alternatives, or non-dictatorship. Every concrete alignment scheme makes a choice about which axiom to relax. RLHF effectively narrows the domain. Constitutional AI bakes in a particular fixed value hierarchy. Debate-style protocols rely on dialectical refinement at the cost of strategy-proofness. Majoritarian preference aggregation accepts the inconsistencies of voting cycles.

The point is not that any of these choices is wrong. The point is that the choice cannot be avoided. There is no computation-only path to a final social welfare function. Someone, outside the system, has to decide which axiom to violate. That decision is governance, not engineering.

This is the deepest sense in which AI is non-final. Even granted everything else - perfect reasoning, perfect prediction, perfect learning - the value step requires a human-political act of choice.

The Lucas-Penrose detour

It would be a mistake to close the series without acknowledging the most famous use of Gödel's theorem to argue for the special status of the human mind. John Lucas (1961) and Roger Penrose (1989, 1994) argued that Gödel's incompleteness theorem shows the human mind cannot be a Turing machine, because humans can "see" the truth of Gödel sentences that the corresponding formal machine cannot prove.

The argument has been criticised, principally by Hilary Putnam, George Boolos, Solomon Feferman, and David Chalmers. The central criticism is that the argument requires the human mind to know its own consistency - and by Gödel's second incompleteness theorem, this is precisely what no consistent formal system can have. To assume that humans know their own consistency is to assume what was to be proved.

The thesis of this series does not need the Lucas-Penrose claim. The series argues only that no formal computational system can be complete, self-validating, universally predictive, universally optimal, or value-final. Whether the human mind is itself a formal computational system is irrelevant to that conclusion. If it is, humans inherit the same ceiling. The important difference is that human reasoning is not exhausted by any single fixed axiomatic frame. Humans switch frameworks. Humans revise axioms. Humans exercise meta-judgement on the adequacy of frames. Whether or not that capacity is itself formally recursive, it is empirically prior to every realised AI.

That is the weakest claim that does the work. We do not need to be non-computable. We need only to be the beings who choose the next computation.

What the math leaves to humans

The ceiling we have walked across nine theorems is real, structural, and not movable by engineering progress. Every theorem closes a specific door. Together they describe what AI cannot become, while leaving in place everything AI is.

What the math leaves to us, in the form of a list:

- We choose the question. Gödel and Tarski together say no model defines the right questions from inside itself.
- We choose the axioms. Every formal system rests on assumptions that the system cannot justify from within.
- We choose the objective. NFL says no algorithm is best for every objective.
- We choose the bias. Goldblum et al. say modern AI works because its biases happen to be aligned with reality.
- We interpret the answer. Tarski says truth requires a metalanguage; we are the metalanguage of last resort.
- We weigh the trade-offs. Arrow and Gibbard-Satterthwaite say no aggregation rule is fair, total, and strategy-proof.
- We decide when the answer is enough. The ladder of incompleteness is infinite; we are the ones who climb it.

None of these is a romantic flourish. Each is what some specific theorem leaves unautomated. The human role is foundational not because humans are cleverer than the machine, but because the math leaves us no choice.

The series at a glance

Theorem	What it rules out	Plain-English meaning
Gödel 1 (1931)	Complete formal truth from inside the system	No fixed formal system captures all arithmetic truth.
Gödel 2 / Löb (1931/1955)	Self-certified consistency	No consistent system honestly proves its own reliability.
Tarski (1936)	Internal truth predicate	Truth lives in a metalanguage, not in the model itself.
Turing halting (1936)	Universal behavioural prediction	There is no general algorithm for what arbitrary programs do.
Rice (1953)	Universal behavioural verification	Every non-trivial semantic property of programs is undecidable.
Busy Beaver / BB (1962)	Computable upper bounds on small machines	Even tiny formal machines escape every computable predictor.
Chaitin (1974)	Certification of complexity beyond the model	A finite model cannot certify irreducible complexity beyond its own size.
Wolpert-Macready (1997)	Universally best learner	Averaged over all problems, all learners are equivalent.
Goldblum et al. (2024)	Free simplicity bias	Deep learning works because the world is simple, not because biases are free.
Solomonoff / AIXI	Computable ideal predictor	Even the theoretical gold standard of induction is uncomputable.
Alfonseca / Brcic-Yampolskiy (2021-23)	Universal AI containment/alignment	Universal alignment is undecidable; provable safety requires architectural restriction.
Merrill-Sabharwal (2023-24)	Transformer-as-universal-computer	Constant-depth log-precision transformers $\subsetneq$ uniform $\mathrm{TC}^0 \subsetneq \mathrm{P}$.
Arrow / Gibbard-Satterthwaite	Value-finality by computation	Aggregating human preferences fairly is mathematically impossible.

Conclusion - Anti-oracle, not anti-AI

The series began with a confession: at Pentera Labs we use AI every working day. The series ends with the same confession, now sharpened. We use AI because it is the most powerful tool our discipline has ever had. We will keep using it. We will keep building with it. We will keep finding new ways for it to make research deeper, faster, and more rigorous.

But the math we walked through is not negotiable. Gödel still holds. Tarski still holds. Turing, Rice, Chaitin, Wolpert, Goldblum, Solomonoff, Merrill, Sabharwal, Arrow, Gibbard, Satterthwaite - they all still hold. The honest posture they leave us with is not anti-AI. It is anti-oracle.

"AI can surpass us locally. Mathematics gives us strong reason to deny that it can surpass us foundationally."

The toolbox is heavier. The hand on the toolbox is still ours.

Further reading (full series)

- Gödel's incompleteness theorems - Stanford Encyclopedia of Philosophy.
- Tarski and axiomatic theories of truth - Stanford Encyclopedia of Philosophy.
- Löb's theorem and provability logic - Stanford Encyclopedia of Philosophy.
- Computability and complexity - Stanford Encyclopedia of Philosophy.
- Rice's theorem and AI safety - Alfonseca et al., JAIR 70 (2021).
- Impossibility results in AI - Brcic & Yampolskiy, ACM Computing Surveys (2023).
- Machines that halt: undecidability of AI alignment - Castro González et al., Scientific Reports (2024/2025).
- On non-computable functions (Busy Beaver) - Radó, Bell System Technical Journal (1962).
- Information-theoretic limitations of formal systems - Chaitin, JACM (1974).
- No Free Lunch theorems - Wolpert & Macready, IEEE Trans. Evol. Comp. (1997).
- No Free Lunch, Kolmogorov complexity and inductive biases - Goldblum, Finzi, Rowan & Wilson, ICML 2024.
- Solomonoff induction and bad universal priors - Leike & Hutter, COLT 2015.
- The expressive power of transformers with chain of thought - Merrill & Sabharwal, ICLR 2024.
- Arrow's theorem and social choice - Stanford Encyclopedia of Philosophy.

About the Author

Shlomo Ben-Yosef is Head of Research at Pentera, with years of experience in cybersecurity research. He majored in Theoretical Physics and Mathematics and did research in number theory, combinatorics, and logic - the lens this series brings to the limits of AI.

Reach out to us with any questions about the research at: labs@pentera.io

PENTERA.

Pentera is the market leader in AI-powered Security Validation, equipping enterprises with the platform to proactively test all their cybersecurity controls against the latest cyber attacks. Pentera identifies true risk across the entire attack surface, and automatically orchestrates remediation workflows to effectively reduce exposure. The company's security validation capabilities are essential for Continuous Threat Exposure Management (CTEM) operations. Thousands of security professionals around the world trust Pentera to close security gaps before threat actors can exploit them.

For more information, visit: pentera.io