The Attralucian Essays:
Exploring the Finite

First Edition

 Typeset in LaTeX

The Attralucian Essays

The P vs NP Problem: A Geofinitist Lens

Kevin R. Haylett

The P vs NP Problem: A Geofinitist Lens

Overview

The P vs. NP problem is one of the central open questions in theoretical computer science, traditionally framed in terms of asymptotic complexity over idealised computational models. While this abstraction has proven extraordinarily powerful for classification, it intentionally suppresses constants, noise, finite resources, and instance-dependent structure.

This paper does not reject the classical formulation. Rather, it asks how these categories behave when computation is treated as a *measured physical process*. Within a Geofinitist framework—where all quantities are finite, measured, and accompanied by uncertainty—the question shifts from a metaphysical statement about infinite limits to an operational inquiry about *finite-regime separability*.

A Motivating Analogy

Consider a locked safe. Given a candidate combination, verification is immediate: the safe opens or it does not. However, discovering the correct combination from scratch may require extensive search. The classical P vs. NP question asks whether this apparent asymmetry is fundamental: if verification is efficient, must discovery also be efficient?

This intuitive distinction—between recognition and construction—has guided decades of research, yet remains unresolved.

Classical Formulation

Formally, the P vs. NP problem concerns two classes of decision problems:

- P **(Polynomial Time):** Problems solvable in time bounded by a polynomial function of input size n.
- NP **(Nondeterministic Polynomial Time):** Problems for which proposed solutions (certificates) can be verified in polynomial time.

The central question is whether $P = NP$. If true, every efficiently verifiable problem would also be efficiently solvable. If false, there exist problems whose solutions can be checked quickly but not found quickly.

Since the work of ***:contentReference[oaicite:0]index=0*** and ***:contentReference[oaicite:1]index=1*** in the early 1970s, this question has been formalised through NP-completeness, with the Boolean satisfiability problem (SAT) as a canonical example. Despite extensive study, no resolution has been found, and the problem remains one of the Clay Mathematics Institute's Millennium Prize Problems.

A Measured Perspective

Classical complexity theory deliberately abstracts away many features of real computation. In practice:

- Computation occurs on finite machines with bounded memory and energy.
- Runtime exhibits variability across instances of equal size.
- Hardware introduces noise, latency, and architectural effects.
- Problem difficulty depends on structure, not merely size.

Geofinitism treats these not as nuisances but as primary features. Complexity becomes a *measured trajectory* rather than a single asymptotic classification.

Geofinitist Pillars Applied to Complexity

1. Geometric Container Space

Classical View: Complexity is a scalar function $T(n)$.

Measured View: Computation traces a trajectory through a high-dimensional space of:

- input structure (e.g. graph density, clause ratio),
- algorithmic pathways,
- hardware states.

Complexity is thus a path within a *container space*, not a point on a curve.

2. Approximations and Measurements

Classical View: $T(n)$ is exact up to asymptotic equivalence.

Measured View: Runtime is a measured quantity with uncertainty:

$$T(n) \pm \varepsilon(n),$$

where $\varepsilon(n)$ reflects both hardware noise and distributional variation across instances.

3. Dynamic Flow of Computation

Classical View: A problem has a single complexity class.

Measured View: Computation unfolds across stages:

$$C(n) = \frac{1}{K} \sum_{i=1}^{K} T_i(n),$$

capturing preprocessing, solving, verification, and post-processing. Classical complexity collapses these layers into a single scalar.

4. Useful Fiction

The equality $P = NP$ is a meaningful abstraction within asymptotic theory. However, it does not directly correspond to measurable reality. It is best treated as a *useful fiction*—a limiting statement whose operational relevance must be grounded.

5. Finite Reality

All computation occurs within finite bounds:

- finite input sizes,
- finite precision,
- finite time and energy.

Claims about infinite scaling must therefore be interpreted through finite observations.

A Formal Geofinitist Framework

Let Π be a problem family. For each input size n, define a measured instance registry:

$$\mathcal{I}_n \subset \mathbb{M}^{d(n)},$$

with provenance describing generation and perturbation processes.

A solver A produces measured runtime:

$$\mathsf{A}(I) = (v_T(I), \varepsilon_T(I), P_{\mathsf{A}}).$$

Define empirical runtime statistics:

$$\hat{T}(n) = \mathrm{median}_{I \in \mathcal{I}_n} v_T(I), \quad \widehat{\sigma}(n), \quad \widehat{\mathrm{IQR}}(n).$$

Define a finite scaling exponent:

$$\widehat{\alpha}(n) = \frac{\Delta \log \hat{T}(n)}{\Delta \log n}.$$

Finite Separability

We define:

- **Polynomial behaviour:** $\hat{T}(n) \lesssim Cn^d$ within uncertainty bands.
- **Verification dominance:** $\hat{T}_V(n) \ll \hat{T}(n)$.

- **Finite separation:** persistent divergence of scaling exponents across observed ranges.

This reframes P vs. NP as a question of *empirical separability*:

Finite Separability: Do solver and verifier trajectories exhibit robus

If not, the question is *underdetermined at the given resolution.*

Robustness via Perturbation

Introduce perturbation operator P_η. Define smoothed runtime:

$$\hat{T}_\eta(n) = \mathbb{E}[v_T(\mathsf{P}_\eta(I))].$$

- Stability under perturbation $\Rightarrow$ robust tractability.
- Persistent divergence $\Rightarrow$ robust hardness.

Interpretation

Within this framework, complexity classes are not fixed sets but *regions in a measured scaling space.* The classical question:

$$P \stackrel{?}{=} NP$$

becomes:

Are solving and verification trajectories empirically indistinguishable a

Conclusion

The classical P vs. NP problem remains unresolved within its asymptotic formulation. Geofinitism does not attempt to resolve it in that domain. Instead, it reframes the question as a measurable, reproducible protocol grounded in finite computation.

This shift replaces metaphysical classification with empirical geometry:

- from infinite limits to finite ranges,
- from exact classes to uncertainty bands,
- from static membership to dynamic trajectories.

In doing so, it provides a practical methodology for understanding computational difficulty as it is actually encountered: within finite systems, under measurable constraints, and across structured instance spaces.