Persistence, Memory, and the Structural Impossibility of P = NP

Abstract

We investigate structural limitations on polynomial-time decision procedures for NP-complete problems arising from persistence and memory. Rather than proposing algorithmic lower bounds or resolving the P versus NP problem outright, this work identifies a general obstruction shared by a broad class of computational models that admit reusable state, verification, and cumulative memory. We show that any uniform decision procedure capable of reuse must accumulate persistent information proportional to the discrimination it performs, and that such persistence necessarily induces irreversible constraints on future computation. We further characterize an alternative regime in which correct decisions may occur without persistent memory, and show that this regime is non-algorithmic and inaccessible to systems capable of verification or reuse. These results delineate a structural boundary on the kinds of computational mechanisms that could plausibly support polynomial-time decision procedures for NP-complete problems.

---

1. Statement of Result

We identify a structural obstruction that excludes polynomial-time decision procedures for NP-complete problems within any class of computational models admitting persistent, reusable memory.

Specifically, we show that any putative polynomial-time decision procedure would require a form of computation that is non-persistent and non-reusable, and therefore cannot be realized within systems capable of memory, verification, or uniform reuse of information.

Since the statement *P = NP* requires a persistent and uniform decision procedure, such procedures are structurally excluded under the assumptions considered here.

2. Minimal Structural Framework

We assume only properties shared by all known models of computation and physical information processing:

- **Pressure**: Nontrivial decision problems induce multiple incompatible futures.
- **Exploration**: Resolving pressure requires discrimination among alternatives.

- **Memory**: Any reusable resolution requires persistence of information.
- **Irreversibility**: Persistent memory induces irreversible constraints on future computation.

These assumptions are descriptive rather than axiomatic and do not depend on any specific machine model.

---

3. Persistence Implies Cost

Definition 3.1 (Persistent Decision Procedure)

A **persistent decision procedure** is a decision procedure whose outputs may be reused, verified, or communicated across multiple invocations, and which therefore admits internal state or memory that persists across time.

Lemma 3.2 (Persistence–Memory Coupling)

Any persistent decision procedure capable of reuse or verification must store information proportional to the discrimination it performs among candidate outcomes.

Proof sketch. Reuse and verification require the ability to distinguish correct outputs from incorrect ones under repetition. This distinction must be encoded in persistent internal state. The amount of such state scales with the effective discrimination performed by the procedure.

Lemma 3.3 (Memory–Irreversibility Coupling)

Persistent memory induces irreversible constraints on future computation.

Proof sketch. Once information is stored, future computations must either respect or explicitly overwrite it. Both possibilities impose irreversible constraints on system evolution, independent of the specific computational model.

---

4. Persistence Obstruction

Theorem 4.1 (Persistence Obstruction)

No persistent decision procedure can decide NP-complete problems in polynomial time within any computational model admitting reusable memory and irreversible state evolution.

Proof sketch. Assume a persistent polynomial-time decision procedure for an NP-complete problem. By Lemma 3.2, the procedure must store information encoding its discriminations. By Lemma 3.3, this storage induces irreversible structure. For NP-complete problems, the amount of discrimination required grows superpolynomially with input size, implying superpolynomial growth of persistent structure, contradicting the assumed polynomial bound.

---

5. Ephemeral Resolution

Definition 5.1 (Ephemeral Resolution)

An **ephemeral resolution** is a correct decision event that leaves no persistent internal state and cannot be reused, verified, or communicated.

Such resolutions do not constitute algorithms, cannot be composed, and cannot support uniform decision procedures.

Theorem 5.2 (Inaccessibility of Ephemeral Resolution)

Ephemeral resolutions do not constitute decision procedures and cannot be composed into uniform algorithms.

Proof sketch. Any attempt to reuse, verify, or communicate an ephemeral resolution requires persistence. Persistence reintroduces memory and irreversibility, placing the system back within the scope of Theorem 4.1.

Related Work

The P versus NP problem was formulated by Cook (1971) and Karp (1972) and has since become central to computational complexity theory. Standard references such as Arora and Barak (2009) survey the landscape of known techniques, including diagonalization, circuit lower bounds, proof complexity, and algebraic methods, none of which have yielded a resolution of the problem.

A substantial body of work has identified broad barriers to existing proof techniques, most notably relativization (Baker, Gill, and Solovay, 1975), natural proofs (Razborov and Rudich, 1997), and algebrization (Aaronson and Wigderson, 2008). These results show that wide classes of standard approaches are insufficient to resolve P versus NP, but they do not assert that separation is impossible.

The present work is orthogonal to these barriers. Rather than studying limitations of proof techniques, we identify a structural obstruction arising from persistence and memory that applies across a broad range of computational models. Our focus is not on the expressive power of specific formalisms, but on properties common to all decision procedures that admit reuse, verification, and cumulative state.

Related perspectives appear in lower-bound arguments for space-bounded computation, proof complexity, and pebbling games, where irreversibility and accumulation of information play a central role. However, such results typically operate within fixed models. In contrast, the obstruction identified here is model-agnostic and applies to any computational framework in which decision procedures are persistent.

---

6. Conclusion

We have identified a structural obstruction to polynomial-time decision procedures for NP-complete problems within any class of computational models admitting persistent, reusable memory and irreversible state evolution. This obstruction rules out a broad family of approaches based on cumulative state, reuse, or verification.

While alternative computational regimes that avoid persistence can be formally described, such regimes do not support uniform decision procedures and therefore do not constitute algorithms in the standard sense. Any proposal for polynomial-time decision procedures for NP-complete problems must explicitly evade the persistence obstruction identified here.

Non-Claims

This document does not claim the nonexistence of abstract mathematical models in which $P = NP$. It claims that no such model can be realized within any computational framework admitting persistent memory and uniform reuse or verification of information.

References

Cook, S. A. (1971). *The complexity of theorem-proving procedures*. Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC), 151–158.

Karp, R. M. (1972). *Reducibility among combinatorial problems*. In R. E. Miller & J. W. Thatcher (Eds.), Complexity of Computer Computations, 85–103. Plenum Press.

Arora, S., & Barak, B. (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press.

Baker, T., Gill, J., & Solovay, R. (1975). *Relativizations of the P = NP question*. SIAM Journal on Computing, 4(4), 431–442.

Razborov, A. A., & Rudich, S. (1997). *Natural proofs*. Journal of Computer and System Sciences, 55(1), 24–35.

Aaronson, S., & Wigderson, A. (2008). *Algebrization: A new barrier in complexity theory*. ACM Transactions on Computation Theory, 1(1), Article 2.

Version 1.0 - January 2026
Author webpage: https://paullynch.me