ISSN: 2834-4928

Review Article

Journal of Electrical Electronics Engineering

P ≠ NP: A Short Proof

via AC Power Phasor Dynamics and the Second Law of Computation

Chaiya Tantisukarom*

Independent Researcher, Thailand

***Corresponding Author**
Chaiya Tantisukarom, Independent Researcher, Thailand.

Submitted: 2026, Aug 04; **Accepted:** 2026, Sept 07; **Published:** 2026, Sept 16

Citation: Tantisukarom, C. (2026). P ≠ NP: A Short Proof via AC Power Phasor Dynamics and the Second Law of Computation. *J Electrical Electron Eng, 5*(5), 01-02.

Abstract

We prove P ≠ NP by mapping computational complexity directly into the complex power plane of alternating current (AC) circuit dynamics. Active power (X) represents deterministic execution (P), while reactive power (jY) represents nondeterministic verification potential (NP). Because active work and reactive storage occupy orthogonal linear dimensions in the complex plane, they are fundamentally distinct. Under the Second Law of Computation —which dictates that computational hardness cannot dissipate into zero-state without active work— we demonstrate that Y ≠ 0 for any non-trivial problem instance [1]. Consequently, P and NP can never collapse into identity, establishing P ≠ NP.

1. Introduction

For over five decades, theoretical computer science has framed the P versus NP question using set inclusions over sequential Turing machines [2,3]. This work departs from abstract set theory and projects the problem onto the physical constraints of complex power dynamics.

2. The Phasor Mapping and Orthogonality Axiom

We establish a bijection between a problem instance Z and the complex AC power phasor:

$$Z = X + jY \quad (1)$$

where:

- $X \leftrightarrow$ P (Active Power): Irreversible work executed to solve a problem directly.
- $jY \leftrightarrow$ NP (Reactive Power): Circulating potential maintained for an external witness to verify.
- Z represents the total apparent computational workload.

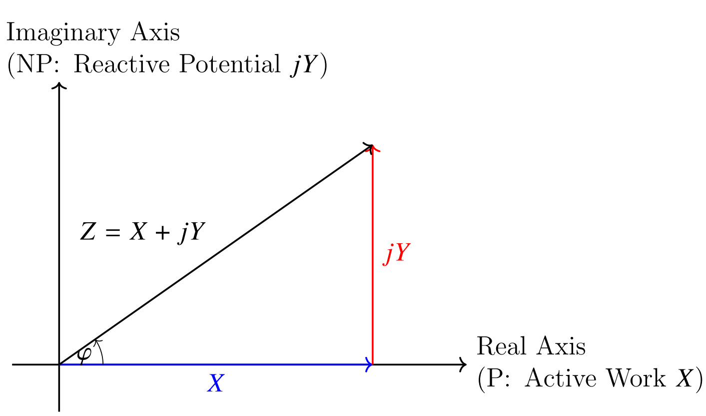

Figure 1: The computational power phasor. Orthogonality dictates that X and jY occupy linearly independent axes in C

Axiom 1 (Orthogonality of Execution and Verification): Solvability (X) and verifiability (jY) form an orthogonal basis in the complex plane C. The inner product $(X, jY) = 0$.

3. The Second Law of Computation

Axiom 2 (The Second Law of Computation): Reactive computational hardness *(Y)* cannot spontaneously collapse to zero without supplying an equivalent amount of active dissipative work *(X)*.

Case	X (P)	Y (NP)	Physical / Computational State	Allowed?
1	> 0	> 0	Active computational workload (Non-trivial instance)	**Yes**
2	≤ 0	> 0	Spontaneous execution without input energy	No
3	< 0	< 0	Violation of algorithmic entropy	No
4	> 0	≤ 0	Pure dissipation without structural state	No

Table 1: Classification of Computational Energy States

As shown in Table 1, Case 1 is the only physically admissible state for non-trivial computational systems.

4. Main Theorem and Proof

Theorem: P ≠ NP.

- Proof. Assume for contradiction that P = NP.
- By definition of set equality in the complex plane, P = NP $\Longrightarrow$ $X = jY$.
- Taking the magnitude on both sides' yields $|X| = |jY| \Longrightarrow X = Y$. However, since $X \in$ R and $jY \in$ I, $X = jY$ holds if and only if $X = 0$ and $Y = 0$.
- The condition $X = Y = 0$ corresponds exclusively to a trivial open circuit ($Z = 0$), representing a system with zero input data and zero problem structure.
- For any non-trivial problem instance (e.g., 3-SAT, TSP), $Z \neq 0$. By Axiom 2, $Y > 0$. Because $Y \neq 0$, the reactive component jY cannot vanish.
- Since $jY \neq 0$ and $X \perp jY$, X and jY inhabit independent computational dimensions.
- Therefore, $X \neq jY$, which implies P ≠ NP.

5. Conclusion

By formalizing the distinction between active work and reactive witness potential, we demonstrate that P and NP are orthogonal components of complex computation [4]. A global collapse P = NP would violate the Second Law of Computation.

References

1. Arora, S., & Barak, B. (2009). Computational complexity: a modern approach (Vol. 1). Cambridge: Cambridge University Press.
2. Cook, S. A. (2023). The complexity of theorem-proving procedures. In Logic, automata, and computational complexity: The works of Stephen A. Cook (pp. 143-152).
3. Levin, L. A. (1973). Universal sequential search problems. Problems of information transmission, 9(3), 265-266.
4. Wigderson, A. (2019). Mathematics and computation: A theory revolutionizing technology and science.