Languages, Parameters, and the Well-Formedness of the P vs NP Question

YANLIN LI
Independent Researcher

Abstract

The classical P vs NP question is formulated over languages defined as subsets of Σ^*. In practice, however, languages arise from layered structures including syntax, semantics, and rule-based admissibility constraints. This paper makes explicit the parameters underlying language formation and shows that meaningful comparison between P and NP requires these parameters to be held fixed. We further provide concrete parameter systems under which P and NP collapse, as well as systems under which they are strictly separated, without appeal to unproven complexity conjectures. Motivated by considerations of structural invertibility, we examine the conditions under which feasibility comparisons between P and NP are well-formed.

Proposition 1 (Type Consistency)**.** *A comparison between* P *and* NP *is meaningful only when both are evaluated under the same parameter system* Θ*.*

Remark 1 (Type-Consistent Feasibility)**.** *We refer to the requirement that feasibility predicates be evaluated under a shared parameter system as* type-consistent feasibility*. Comparisons that violate this requirement are ill-typed and lack a well-defined interpretation.*

1 Formal Languages

Definition 1 (Alphabet and Language)**.** *Let* Σ *be a finite alphabet. A* formal language *over* Σ *is any set*

$$L \subseteq \Sigma^*,$$

where Σ^* *denotes the set of all finite strings over* Σ*.*

Remark 2. *This definition is intentionally minimal. It treats languages purely as sets of strings and does not distinguish between syntactic well-formedness, semantic interpretation, or contextual legitimacy.*

2 Classical Complexity Classes

Definition 2 (**P**)**.** *A language* $L \subseteq \Sigma^*$ *is in* **P** *if there exists a deterministic Turing machine deciding membership in* L *in polynomial time.*

Definition 3 (**NP**)**.** *A language* $L \subseteq \Sigma^*$ *is in* **NP** *if there exists a polynomial-time verifier* V *such that*

$$x \in L \iff \exists y \text{ with } V(x, y) = 1,$$

where the verification runs in time polynomial in $|x|$*.*

Remark 3. *In both definitions, the sole explicit object of classification is the language* L*. Encoding choices, admissibility rules, and interpretation constraints are implicit.*

3 Languages as Parameterized Constructions

We now make explicit the components commonly suppressed in the classical view.

Definition 4 (Language Parameters). *Let*

$$\Theta = (\Sigma, \mathsf{Syn}, \mathsf{Sem}, \mathsf{Prag}),$$

where Syn *specifies syntactic well-formedness,* Sem *provides semantic interpretation, and* Prag *imposes rule-based admissibility constraints.*

Definition 5 (Generated Language). *The language induced by* Θ *is*

$$\mathcal{L}(\Theta) \subseteq \Sigma^*,$$

consisting of strings admissible under the combined constraints of Θ*.*

4 Parameterized Feasibility

Definition 6 (Parameterized Predicates). *Define predicates*

$$\mathsf{P}(\Theta) = 1 \iff \mathcal{L}(\Theta) \in \mathbf{P}, \qquad \mathsf{NP}(\Theta) = 1 \iff \mathcal{L}(\Theta) \in \mathbf{NP}.$$

Remark 4. *This does not alter classical complexity theory. It makes explicit that P and NP are evaluated relative to a fixed parameter bundle* Θ*.*

5 Well-Formed Comparison

Proposition 2 (Type Consistency). *A comparison between* P *and* NP *is meaningful only when both are evaluated under the same parameter system* Θ*. Equivalently, statements of the form "*$P(\Theta_1)$ *versus* $NP(\Theta_2)$*" are well-formed only when* $\Theta_1 = \Theta_2$*.*

Proof. Fix the ambient complexity classes P and NP as classes of languages (i.e., subsets of Σ^*), as in Definitions 2–3. Under Definition 5, each parameter bundle Θ induces a specific language object $L(\Theta) \subseteq \Sigma^*$.

Under Definition 6, the feasibility predicates are abbreviations for class membership of that induced language:

$$P(\Theta) = 1 \iff L(\Theta) \in P, \qquad NP(\Theta) = 1 \iff L(\Theta) \in NP.$$

Hence any combined statement comparing $P(\Theta_1)$ and $NP(\Theta_2)$ is, upon expansion, a statement simultaneously about the two language objects $L(\Theta_1)$ and $L(\Theta_2)$.

If $\Theta_1 \neq \Theta_2$, then in general $L(\Theta_1) \neq L(\Theta_2)$ because Θ contains syntactic, semantic, and admissibility components that determine which strings are included (Definition 4). Therefore, a cross-parameter comparison such as

$$P(\Theta_1) \text{ vs. } NP(\Theta_2)$$

does not compare two properties of a single fixed object, but rather compares properties of *distinct* objects without any specified identification map or correspondence between them.

In other words, the judgement "compare P and NP" is type-correct only relative to a single shared Θ that fixes the underlying language $L(\Theta)$; otherwise the comparison is ill-typed because the implicit subject of classification varies across the two predicates. □

Observation 1 (Decision–Verification Asymmetry)**.** *Even under a fixed parameter system* Θ*, the feasibility of decision does not, in general, entail the feasibility of verification. That is, the existence of a polynomial-time decision procedure for* $L(\Theta)$ *does not guarantee the existence of an admissible polynomial-time verification relation as required by* $\mathbf{NP}(\Theta)$*.*

Remark 5 (Feasibility over syntactic and admissibility regimes)**.** *Fix the alphabet* Σ *and the semantic interpretation* Sem*. For varying syntactic well-formedness and admissibility rules, define*

$$\Theta(\mathrm{Syn}, \mathrm{Prag}) \ := \ (\Sigma, \mathrm{Syn}, \mathrm{Sem}, \mathrm{Prag}).$$

The predicates

$$\{\, P(\Theta(\mathrm{Syn}, \mathrm{Prag}))\,\}_{(\mathrm{Syn},\mathrm{Prag})} \quad \textit{and} \quad \{\, NP(\Theta(\mathrm{Syn}, \mathrm{Prag}))\,\}_{(\mathrm{Syn},\mathrm{Prag})}$$

thus form families indexed by syntactic and admissibility regimes. Their unions across varying $(\mathrm{Syn}, \mathrm{Prag})$ *correspond only to projections over parameter space and do not constitute well-defined complexity classes; in particular, they admit no intrinsic inclusion or comparison relation.*

6 Concrete Instances

We now give explicit realizations under fixed parameter systems.

6.1 Collapse by Exclusion

Let Θ_1 impose the rule that no witness is ever legitimate. Then

$$\mathsf{P}(\Theta_1) = \mathsf{NP}(\Theta_1) = \varnothing.$$

6.2 Finite Input Collapse

Let Θ_2 restrict admissible inputs to a finite set. Then all languages are decidable by constant-time lookup, yielding

$$\mathsf{P}(\Theta_2) = \mathsf{NP}(\Theta_2).$$

6.3 Structural Separation

Let Θ_3 permit existence and verification of legitimate witnesses while forbidding systematic construction. Then there exists a language L such that

$$L \in \mathsf{NP}(\Theta_3) \quad \text{but} \quad L \notin \mathsf{P}(\Theta_3),$$

and hence

$$\mathsf{P}(\Theta_3) \subsetneq \mathsf{NP}(\Theta_3).$$

7 Conclusion

The relation between P and NP is not a property of languages in isolation, but of languages interpreted relative to a fixed parameter system. Making these parameters explicit clarifies the conditions under which feasibility comparisons are well-formed.

Within this perspective, both equality and strict separation between P and NP arise as realized possibilities under appropriate parameter choices, rather than as conjectural outcomes of a single universal comparison. The classical P versus NP question thus corresponds to a highly constrained slice of a broader feasibility landscape, in which comparisons are meaningful only when the underlying syntactic and admissibility regimes are held fixed.

Acknowledgements

This work represents an early-stage foundational exploration intended to inform the future development of the Open Governance Modeling Framework (OGMF). No formal organization or research collective under this name has been established at the time of writing.