MAG: Formalizing the Solvability Barrier in P vs NP via Group-Theoretic Semantics

Masaru Numagaki
Independent Researcher, Kumamoto, Japan

January 2026

ACM CCS (2012): Theory of computation → Complexity classes; Algebraic complexity theory; Automated reasoning. Software and its engineering → Formal methods.
MSC2020: 68Q15, 68V15, 20F16, 20D06
Keywords: P vs NP, computational complexity, alternating group A_5, group-theoretic semantics, solvable groups, formal verification, Lean 4, Krohn–Rhodes decomposition, Barrington's theorem, CFSG (Classification of Finite Simple Groups)

Contents

1 Introduction **3**
1.1 Background: P vs NP Normalization to "Galois Form" . . . 3
1.1.1 Why A_5? The Classification of Finite Simple Groups . . . 4
1.2 Contributions . . . 4
1.3 Remark on the Numerical Gap $\Delta D = 7/6$ . . . 5
1.4 Scope of Claims and Non-claims . . . 5
1.5 How to Read Formalization (Lean Artifact) . . . 5
1.5.1 Supplement 1.5.1: Lessons Learned from Formalization . . . 5
1.6 Paper Structure . . . 6
1.7 Anticipated Objections . . . 6

2 MAG Model (Definition Bundle) **8**
2.1 Basic Concept: Translating Computation into Group Theory . . . 8
2.1.1 Historical Foundation: Krohn–Rhodes Decomposition . . . 9
2.2 Definitions of P, NP-hard, and Solves in MAG . . . 9
2.3 Fundamental Lemma: Isomorphism Preserves Solvability . . . 11
2.4 The A_5 Nucleus Does Not Fall into the Solvable Side . . . 11
2.5 Numerical Gap $\Delta D = 7/6$ (Remark) . . . 11
2.6 Summary of This Section . . . 11

3 P $\neq$ NP in MAG (Main Theorem) **11**
3.1 Master Theorem (Within MAG, P $\neq$ NP) . . . 12
3.2 The Core of the Proof: Invariance of Isomorphism and Solvability . . . 12
3.3 Proof Sketch . . . 12
3.4 Lean Formalization (Reader's Guide) . . . 13
3.5 Summary . . . 13

4 A_5 Universal Non-identifiability Barrier **13**
4.1 Motivation: Treating Separation as "Inidentifiability" Rather than "Computational Complexity" . . . 13
4.2 Abstraction: Descriptive Systems and Deep Structures . . . 13
4.3 Central Lemma: Unsolvable Kernels Propagate via Embedding . . . 14
4.4 Universal Barrier with A_5 as the Core . . . 14
4.5 Relationship with Separation Within MAG . . . 15
4.6 Connection with Numerical Gap (Remark) . . . 15
4.7 Vanishing Depth Barrier with Commutator Trees (Theorem 4.7) . . . 15
4.8 Summary of This Section . . . 15

5 Translation Interface and Bridge Theorem **15**
5.1 Abstract Computational Model `ComputationalModel` . . . 16
5.2 Translation Interface `TranslationInterface` . . . 16
5.3 P = NP Proposition on the External Model . . . 17
5.4 Bridge Theorem . . . 17
5.5 End-to-End Verification Using Toy Models . . . 18
5.6 Relationship Between Standard P and NP (Scope of This Paper) . . . 18
5.7 Summary of This Section . . . 18

6 Conclusion **19**

Abstract

This paper separates P vs. NP from the fixed vocabulary of "resource measures on the standard model of computation" and introduces MAG (Minimum Asymmetry Gap) as a framework for clarifying the semantics of computation. MAG defines (i) polytime as the solvability of a group, (ii) NP-hard as an injective embedding of the minimal unsolvable kernel A_5, and (iii) solves as a group isomorphism. Within this formalized semantic framework, we demonstrate that the separation P $\neq$ NP arises structurally as a consequence of Galois theory. We then rigorously extract the necessary conditions for this separation to hold in the standard model as the `TranslationInterface` type class.

Furthermore, for any computational model M, we type-classify the assumption that maps standard vocabulary to MAG semantics as `TranslationInterface`, and show that if this holds, separation is transferred on M (conditional, full formalism). In addition, we construct a toy model (BarringtonToy) that satisfies `TranslationInterface` and show that it is machine-verifiable from end to end.

This paper does not claim to have definitively solved the Clay problem under the standard TM/circuit definition. Instead, we present a strict separation in Lean between the formalized kernel and the boundary of the translation assumptions necessary to connect to the standard model.

1 Introduction

1.1 Background: P vs NP Normalization to "Galois Form"

P vs NP has been formulated as a central problem in computational complexity, but this question implicitly includes the assumption that "the body of computation can be exhausted by standard models (Turing machines/circuits)." This paper reexamines this assumption. More precisely, just as the classic question of "Can a general quintic equation be solved using radicals?" was normalized by Galois theory into the structural question of "Is it a solvable group?", we clarify "in what description system (computational semantics) should we consider something to be 'solvable'?" in P vs NP, and then formalize the point at which the separation is enforced.

Methodological Foundation: Klein's Erlangen Program. The approach of translating diverse mathematical objects into the common language of *groups* has a distinguished 150-year history. While Galois (1832) discovered that the solvability of polynomial equations corresponds to group solvability, it was **Felix Klein's Erlangen Program (1872)** [4] that elevated this translation into a *general methodology.* Klein proposed that geometry should be understood as the study of invariants under transformation groups—effectively "translating" geometric spaces into group-theoretic objects for classification.

MAG can be understood as the **computational analogue of the Erlangen Program**: just as Klein classified geometries by their symmetry groups, MAG classifies computational problems by their symmetry groups and algorithms by their operation groups. The Krohn–Rhodes decomposition theorem (1965) provided the technical bridge by showing that finite automata decompose into group components, and Barrington's theorem (1986) revealed that the non-solvable group A_5 plays a critical computational role. MAG synthesizes these threads into a unified framework for P vs NP.

From this perspective, this paper introduces a model called MAG (Minimum Asymmetry Gap) that translates computation into group-theoretic objects. In MAG, "efficiency (polytime)" is not a commutative resource measure, but rather the solvability of a group. The core of NP-hardness is A_5 as the smallest unsolvable kernel. The central argument of this paper is that the moment we adopt this semantics, we are freed from the disconnection between solvability and unsolvability—**separation (P $\neq$ NP) is enforced as a theorem**.

1.1.1 Why A_5? The Classification of Finite Simple Groups

The choice of A_5 as the kernel of NP-hardness is not arbitrary but mathematically determined. The Classification of Finite Simple Groups (CFSG), one of the greatest achievements of 20th-century mathematics, establishes that all finite simple groups fall into exactly four families: cyclic groups of prime order, alternating groups A_n ($n \geq 5$), groups of Lie type, and 26 sporadic groups.

Among these, only cyclic groups $\mathbb{Z}_p$ are solvable. The Feit–Thompson Theorem (1963) further establishes that *all groups of odd order are solvable*, meaning non-solvability can only arise in groups of even order. The smallest non-solvable simple group is precisely A_5, with order 60.

This classification provides the "ground truth" for our framework: A_5 is not merely an example but the **mathematically minimal point** at which non-solvability manifests.

1.2 Contributions

The contributions of this paper are as follows. All results do not depend on any `sorry` statements or additional axioms.

(A) Separation theorem in MAG (unconditional and complete formalism). We introduce the MAG semantics as a definition in the language of group theory:

- MAG-polytime (algorithm side) = The algorithm group is solvable
- MAG-NP-hard (problem side) = The problem set is unsolvable ($\neg$IsSolvable($G(p)$))
- MAG-solves = the two correspond by group isomorphism

Under this definition, due to the insolvability of A_5 and the fact that "group isomorphism preserves solvability", **P=NP in MAG is false** (P $\neq$ NP in MAG).

(B) Bridging theorem assuming a translation interface (conditional and complete formalism). Abstract classes with standard vocabulary (polytime / NP-hard / solves) for any computational model M are defined as `ComputationalModel`, and further show that it can be mapped to the MAG semantics via `TranslationInterface`. `TranslationInterface` conceptually requires that:

1. polytime $\Rightarrow$ falls into a solvable group
2. NP-hard $\Rightarrow$ injective embedding of the A_5 kernel (which leads to unsolvability)
3. solves $\Rightarrow$ gives group isomorphism correspondence

Under this assumption, we transfer the intra-MAG separation (A) to the general model: **if `TranslationInterface` M holds, then** P $\neq$ NP **on** M follows.

(C) End-to-end verification of toy models (unconditional and fully formalized). To show that bridging is not just empty theory, we construct a toy computation model (BarringtonToy) that satisfies `TranslationInterface`, and then use an embedding of type $A_5 \hookrightarrow S_5$ to construct a witness. We use Lean to confirm that $\neg$(P = NP) is derived.

(D) Barrington-type depth-consistent solvability barrier (unconditional and complete formalism). In Barrington's width-5 branching program, the computational syntax is expressed as a commutator composition. In this paper, to avoid the trivialization caused by the self-commutator $[x, x] = 1$, we define a balanced commutator tree of depth n as the syntax and show that the evaluation value of depth n always falls into derivedSeries(G, n). As a result, if a group is solvable, then there exists a finite depth k such that a commutator tree of depth $n \geq k$ collapses to the identity under arbitrary substitution (vanishing depth). In contrast, A_5 does not have finite vanishing depth in this sense.

1.3 Remark on the Numerical Gap $\Delta D = 7/6$

Although a numerical gap of $\Delta D = 7/6$ appears in this project, this is not a premise of the separation theorem in this paper. The discontinuity between solvable and unsolvable kernels is the central argument of this paper, and the numerical gap is briefly derived in Section 2.5 as a numerical example to aid in the intuition of "irreversibility by projection."

1.4 Scope of Claims and Non-claims

To avoid any misunderstanding, we will clarify the scope of this paper.

- This paper **unconditionally asserts** a separation within the computational semantics defined as MAG (P $\neq$ NP).
- This paper **makes a conditional claim**:
  for any computational model M, if `TranslationInterface` M holds, then separation also follows.
- This paper **does not claim** that the Clay problem under the standard definition of Turing machines and circuits has been resolved unconditionally. The assumptions required for identification with the Standard Model are separated as `TranslationInterface`, and the formalized core and unformalized translation tasks are separated in Lean.

1.5 How to Read Formalization (Lean Artifact)

The Lean artifacts accompanying this paper are organized so that the main theorem can be seen directly from the entry file. The core (the part that contributes to the assertion) is `sorry`-free, while the blueprint for realizing the translation into the Standard Model is separated as Blueprint and is not included in the Core Claims.

Theorem correspondence table:

Paper	Lean File	Lean Name
Theorem 3.1 (MAG separation)	`Core/MAG_P_neq_NP.lean`	`MAG.Core.MAG_P_neq_NP`
Theorem 4.6 (Universal Barrier)	`Core/A5_Barrier.lean`	`MAG.Core.A5_Universal_Barrier`
Theorem 4.7 (Vanishing Depth)	`Core/BarringtonBarrier.lean`	`...solvable_vanishing_depth`
Theorem 5.4 (Bridge)	`Core/Bridge/Theorem.lean`	`MAG.Core.Bridge.Bridge_P_neq_NP`
Theorem 5.7 (Toy)	`Core/Bridge/ToyInstantiation.lean`	`...BarringtonToy_P_neq_NP`

1.5.1 Supplement 1.5.1: Lessons Learned from Formalization

The Lean formalism makes it clear where things that are vague on paper break down, and where design decisions shift the burden of proof. Key practical lessons learned from this paper:

1. **Injectivity is essential for kernel conditions.** When "contains" is used as the kernel condition, the mere existence of $A_5 \to G(p)$ can make the condition meaningless by a trivial homomorphism, so injective must be explicitly stated when assuming or using the A_5 kernel. In the Core formalism (`MAG_P_neq_NP.lean`), MAG-NP-hard is defined as "$G(p)$ is unsolvable," and the A_5 kernel ($A_5 \to G(p)$ is injective) is treated separately as a sufficient condition and translation principle.

2. **The strength of solves determines the strength of separation.** Fixing Solves(a, p) as an isomorphism (`Nonempty (A(a) ≃* G(p))`) allows for mechanical invocation of solvability preservation (Lemma 2.7), whereas relaxing it to the existence of homomorphisms

or mappings removes the conservation law and exposes the possibility that separation does not hold. The formalism visualizes the strength with which the semantics of "solve" enforces separation.

3. **Barrington-type augmentation requires syntax avoiding self-commutator trivialization.** Because the naive definition of the commutator for measuring depth can be collapsed identically ($[x, x] = 1$), this paper defines a balanced commutator tree as a syntax and shows that evaluation falls to $\text{derivedSeries}(G, n)$ (Barrier section). This design has a safety margin established by formalization.

4. **Separating "Core" from "Translation Debt (Blueprint)" fixes scope institutionally.**
The Core (separation, barrier, bridging theorem, and toy verification) can be verified without any `sorry`/additional axioms, and the additional requirements for standard model identification are localized to the `TranslationInterface` side. This makes it clear in Lean what is a definitive theorem and what is a future translation issue.

5. **The shorter the proof, the more transparent the dependencies.** Although the intra-MAG separation is short, the formalism clearly exposes where the unsolvability of A_5 and solvability preservation by isomorphism are used. In addition, this paper also presents an enhanced version of the argument using the A_5 kernel (injective embedding), and separates the argument that the load can be shifted to a "kernel propagation"-type lemma if necessary.

1.6 Paper Structure

Proof outline (Lean compatible):

- Section 2: MAG model definition (`Core/MAG_P_neq_NP.lean`).
- Section 3: Main theorem for P $\neq$ NP in MAG (`MAG.Core.MAG_P_neq_NP`).
- Section 4: Universal barrier + Barrington-type reinforcement (`Core/A5_Barrier.lean` and `Core/BarringtonBarrier.lean`).
- Section 5: Bridge to the Standard Model (`Core/Bridge/Theorem.lean` and `Core/Bridge/ToyInstantiation.lean`).
- Appendix: Artifact map and reproduction steps (`Main.lean lake build` / `#check`).

Section overview:

- Section 2: Presentation of the MAG model (definition lattice)
- Section 3: Main theorem for P $\neq$ NP in MAG
- Section 4: General form of universal barrier with A_5 as kernel
- Section 5: `TranslationInterface` and bridging theorem, toy verification
- Appendix: Formalized reproduction steps and artifact map

1.7 Anticipated Objections

A: Isn't that a tautology? Because this paper defines P as a solvable group structure and NP-hardness as the injective appearance of an unsolvable kernel (A_5), it may seem obvious that solvable and unsolvable groups are not isomorphic. However, this "obviousness" is precisely the goal of this paper. The argument of this paper is not simply a paraphrase; it formalizes, together with a translation interface, the algebraic inconsistency that arises when P = NP, namely, the inflow of an A_5 unsolvable kernel into the solvable structure, thereby exposing the kernel and responsibility boundary in a machine-verifiable form.

B: Circuits are universal, so why are they limited to being "solvable"? It is necessary to distinguish between "the function to be computed (semantics)" and "the construction rules of the computational device (syntax)." While circuits can certainly compute many functions, the "syntactic transformations" generated by the basic gates (AND, OR, NOT) of standard Boolean circuits and their composition rules are constrained to structures that are solvable from the perspective of Krohn–Rhodes decomposition. See Section 2.1.1 for a detailed explanation of why this is the case.

C: Isn't this not an unconditional proof for the standard model (TM/circuit)? This paper intentionally separates the core (Core) that is completed within MAG from the translation conditions (Bridge) required to connect to the standard model. The central point is that it formalizes and fixes "what is the mathematical core that creates the separation?" and externalizes the algebraic conditions that the standard model must satisfy as a type (specification) called `TranslationInterface`.

D: Barrington is an NC^1 theorem, so isn't extending it to the whole of P a leap? From a symmetry perspective, the presence or absence of a kernel that necessarily appears within a class is more important than the "breadth" of the class itself. Polynomial-time reduction to NP-complete problems preserves the problem structure, and the A_5 kernel of SAT propagates to the reduction target. Barrington's role is to provide the minimal case in which "A_5 inevitably appears even in NC^1," but he does not use it to characterize P as a whole.

E: How is the gap between finite groups and infinite input handled? While Krohn–Rhodes theory applies to finite automata/semigroups, our model relies on the Barrington approach: for each input size n, we consider a sequence of groups G_n, and for NP-hard problems, this sequence maintains a uniform structure (an embedding of A_5), while the group of operations for polynomial-time circuits remains a variety of solvable groups.

F: Why identify "efficient computation" with "solvable groups"? Misconception: "Polynomial time (Polytime)" and "group solvability" appear to be entirely different concepts. Isn't this identification an arbitrary contrivance to make the proof work?

Response: No, this is a natural consequence based on the **Krohn–Rhodes Decomposition Theorem** in automata theory. This theorem shows that any finite automaton (and equivalently, any circuit component) can be decomposed into a **wreath product** of *simple groups* and *aperiodic monoids* (flip-flops, etc.).

1. **Algebraic properties of standard logic gates**: AND/OR gates are aperiodic (trivial as groups), and NOT gates correspond to the cyclic group $\mathbb{Z}_2$ (a solvable simple group).
2. **Preservation of solvability**: The wreath product of solvable groups is always solvable.

Therefore, circuits composed of standard logic gates—no matter how complex the wiring—cannot escape the "solvable group hierarchy (Level 1 complexity)" algebraically. To obtain non-solvable computational power, one would need to physically introduce a non-solvable gate like A_5 (a Level 2 component) as a syntactic building block. The definition `IsMAGPolyTime` in this paper faithfully captures this "limitation of syntax generated from standard gates" in algebraic terms.

G: Isn't the choice of A_5 arbitrary? Are there no other barrier candidates? Response (CFSG justification): The choice of A_5 is not arbitrary but **mathematically uniquely determined**. The Classification of Finite Simple Groups (CFSG), one of the greatest achievements of 20th-century mathematics, establishes that all finite simple groups fall into exactly four families:

1. Cyclic groups of prime order $\mathbb{Z}_p$ (solvable)

2. Alternating groups A_n ($n \geq 5$, non-solvable)
3. Groups of Lie type
4. 26 sporadic groups

The Feit–Thompson Theorem (1963) further establishes that all groups of odd order are solvable. Therefore, non-solvability can only arise in groups of even order. A_5 (order 60) is the **smallest non-solvable simple group** guaranteed by CFSG, meaning there are no undiscovered smaller candidates. The name "Minimum Asymmetry Gap" in MAG directly reflects this mathematical fact.

H: Could the discovery of new gates with different algebraic properties invalidate this barrier? Response (Krohn–Rhodes universality): The Krohn–Rhodes theorem is a **universal theorem** that applies to *all finite semigroups/automata*, not to a specific gate set. Any physically realizable gate must be describable as a transformation between a finite number of states, and therefore necessarily falls under Krohn–Rhodes decomposition.

Specifically:

- If a new gate G is discovered, its Krohn–Rhodes decomposition can be computed.
- Unless the decomposition contains a non-solvable simple group, it remains within the solvable hierarchy.
- If it contains a non-solvable simple group, that means it contains A_5 (or a larger non-solvable group), placing it outside the scope of "standard" gates.

Thus, future gate discoveries do not invalidate this framework. Rather, such gates would simply be classified according to which level of the Krohn–Rhodes hierarchy they belong to.

I: Doesn't an "intermediate structure" exist between solvable and non-solvable groups? Response (CFSG + Feit–Thompson): No. This is not a conjecture but a **proven fact** from major achievements of 20th-century mathematics:

1. **Feit–Thompson Theorem** (1963, 255-page proof): All finite groups of odd order are solvable.
2. **CFSG**: The smallest non-solvable simple group of even order is A_5 (order 60).

Therefore, the solvable/non-solvable boundary is **strictly fixed at order 60**, and intermediate concepts such as "partially solvable" or "weakly non-solvable" cannot exist mathematically. This strictness forms the foundation of the separation theorem in MAG.

2 MAG Model (Definition Bundle)

In this section, we define the MAG (algebraic semantics) used in this paper as a Lean formalism **definition bundle**. The important thing is that MAG is not a method of proving and deriving a principle, but rather **what counts as the body of computation**. The separation theorem in this paper holds under this semantic layer.

2.1 Basic Concept: Translating Computation into Group Theory

In MAG, computational processes, algorithms, and problems (languages) are translated into group-theoretic objects and then handled.

- **Algorithm side**: Procedure composition, branching, and local operations correspond to the "composition of transformations" that follow certain production rules, and are represented as groups (or group actions).
- **Problem side**: Symmetries that identify inputs, invariances of decisions, and constraint actions are expressed as symmetry groups (or subgroups thereof).
- **Solve**: The algorithm's solution to the problem corresponds to the fact that the structures of both are consistent and achieve the same symmetry in a certain sense.

Based on this translation, "computational resource constraints" are understood not only in terms of commutative measures (time and circuit size), but also in terms of the structure of group solvability.

2.1.1 Historical Foundation: Krohn–Rhodes Decomposition

The translation of computation into group theory is not an ad hoc construction but has deep roots in automata theory. The **Krohn–Rhodes Decomposition Theorem** [5] establishes that every finite automaton (or equivalently, every finite semigroup) can be decomposed into a cascade (wreath product) of two types of building blocks:

1. **Finite simple groups**
2. **Finite aperiodic monoids** (flip-flops, resets—components with no non-trivial group structure)

This decomposition provides the mathematical foundation for our claim that "polytime = solvable." Specifically, the *group complexity* of a finite automaton is the minimum number of simple group components required in any such decomposition. This yields a natural hierarchy:

- **Level 0 (Combinational/Aperiodic)**: No simple groups required; only aperiodic components. This corresponds to AND/OR/NOT circuits without feedback.
- **Level 1 (Solvable)**: Only solvable simple groups (cyclic groups $\mathbb{Z}_p$) are required. This encompasses counting and modular arithmetic.
- **Level 2+ (Non-solvable)**: Requires non-solvable simple groups, with A_5 as the minimal such group.

The MAG framework can be understood as formalizing this hierarchy in the context of P vs NP, where the solvability barrier corresponds precisely to the transition from Level 1 to Level 2 in the Krohn–Rhodes complexity hierarchy.

Remark 2.1 (Gate-Level Solvability Guarantee)**.** The standard Boolean gates have the following group-theoretic character under Krohn–Rhodes decomposition:

- **AND/OR gates**: Aperiodic (trivial group component, corresponding to `Unit` in Lean)
- **NOT gate**: Cyclic group $\mathbb{Z}_2$ (solvable, corresponding to `Multiplicative (ZMod 2)` in Lean)
- **INPUT**: Identity transformation (trivial)

Crucially, the wreath product (cascade composition) of solvable groups remains solvable. Therefore, any circuit composed of standard gates has a Krohn–Rhodes decomposition using only solvable simple groups.

This provides the **mathematical guarantee** that standard polynomial-time circuits correspond to solvable group structures, which is formalized in `Support/Circuits/KrohnRhodes.lean`.

2.2 Definitions of P, NP-hard, and Solves in MAG

Definition 2.2 (Problem Instance (Definition 2.1))**.** The object that represents the problem (language) is `ProblemInstance`. Each problem instance is associated with a group that represents at least the "symmetry of the problem side." Notation: We write the symmetric group of problem p as $G(p)$.

Definition 2.3 (Algorithm (Definition 2.2))**.** The object that represents the algorithm is `Algorithm`. Each algorithm is accompanied by at least a group that represents the "algorithm-side composition rules." Notation: The group of algorithm a is written as $A(a)$.

Definition 2.4 (MAG-polytime: Efficiency as Solvability (Definition 2.3))**.** In MAG, "polytime" is defined as a set of algorithms that are solvable:

$$a \in \mathrm{P}_{\mathrm{MAG}} \quad :\Longleftrightarrow \quad A(a) \text{ is solvable.}$$

Intention: Solvable groups have a structure that can be decomposed stepwise, which we use here as a proxy for "low descriptive complexity/low non-commutativity."

Krohn–Rhodes Justification: As established in Section 2.1.1, standard Boolean circuits decompose into solvable (Level 0 or Level 1) components. This makes the correspondence between "polytime circuits" and "solvable groups" a consequence of the Krohn–Rhodes decomposition theorem, not an arbitrary definition.

Definition 2.5 (MAG-NP-hard: Difficulty as a Non-solvable Kernel (Definition 2.4))**.** NP-hardness in MAG is defined as the unsolvability of the problem group:

$$p \text{ is MAG-NP-hard} \quad :\Longleftrightarrow \quad G(p) \text{ is an unsolvable group } (\neg\mathrm{IsSolvable}(G(p))).$$

Intention: NP-hard abstracts the idea that "unsolvability remains that cannot be recovered by a solvable description system." We use A_5 as a representative example and witness, as the smallest unsolvable kernel.

Remark 2.6 (Strengthening Condition: A_5 Kernel (Remark 2.4.1))**.** Intuition and translation (Bridge/Blueprint) sometimes use a stronger condition: A_5 is injectively homomorphically embedded in $G(p)$ ($A_5 \to G(p)$ is injective). This is a sufficient condition to imply MAG-NP-hardness (unsolvability), and naturally appears as a "kernel recovery" when connecting from standard computational models to the MAG vocabulary.

Definition 2.7 (Solves: "Solving" as a Group Isomorphism (Definition 2.5))**.** In MAG, "an algorithm solves a problem" is defined as a group isomorphism between a group of algorithms and a group of problems:

$$\mathrm{Solves}(a, p) \quad :\Longleftrightarrow \quad A(a) \simeq G(p) \quad \text{(group isomorphism)}.$$

Intention: "Solvable" does not simply mean that the outputs match, but rather that the algorithm can fully realize the symmetry of the problem. This strength directly connects the discontinuity between solvable and unsolvable kernels to "separation."

Note on Design Choice: This isomorphism requirement is intentionally stronger than functional equivalence. We call this *structural realizability*—the algorithm must structurally mirror the problem's symmetry. This design isolates the algebraic barrier where separation becomes provable. See Remark 5.5 in Section 5 for detailed discussion.

Remark 2.8 (The Absolute Minimality of A_5 (Remark 2.6.1))**.** The choice of A_5 as the non-solvable kernel has precise mathematical justification rooted in CFSG. The non-abelian simple groups, ordered by cardinality, begin:

Order	**Group**
60	$A_5 \cong \mathrm{PSL}_2(5) \cong \mathrm{PSL}_2(4)$
168	$\mathrm{PSL}_2(7)$
360	A_6

The Feit–Thompson Theorem guarantees that non-solvability requires even order, and A_5 (order 60) is the minimum. Our "Minimum Asymmetry Gap" is thus the mathematically determined boundary between solvable and non-solvable structures.

Definition 2.9 (P=NP in MAG (Definition 2.6))**.** In MAG, "P=NP" is formulated as the existence of a MAG-polytime algorithm that can solve a MAG-NP-hard problem:

$$\mathrm{P} = \mathrm{NP}_{(\mathrm{MAG})} \quad :\Longleftrightarrow \quad \forall p \; (\text{NP-hard}(p) \Rightarrow \exists a \; (\mathrm{poly}(a) \wedge \mathrm{Solves}(a, p))).$$

2.3 Fundamental Lemma: Isomorphism Preserves Solvability

Since the definition of MAG includes "solving = isomorphism", the following fact of group theory immediately plays a central role:

Lemma 2.10 (Isomorphic Invariance of Solvability (Lemma 2.7))**.** *If there exists a group isomorphism $A(a) \simeq G(p)$, then $A(a)$ is solvable if and only if $G(p)$ is solvable:*

$$A(a) \simeq G(p) \quad \Longrightarrow \quad \big(A(a) \textit{ solvable} \Leftrightarrow G(p) \textit{ solvable}\big).$$

2.4 The A_5 Nucleus Does Not Fall into the Solvable Side

Lemma 2.11 (Insolvability of A_5 (Lemma 2.8))**.** *The alternating group A_5 is not solvable:*

$$\neg(A_5 \textit{ solvable}).$$

This fact is formalized by existing group theory (Mathlib).

2.5 Numerical Gap $\Delta D = 7/6$ (Remark)

In this paper, we define the effective dimension as $D_n := n - 1/n$. In this case, the adjacent difference is

$$D_n - D_{n-1} = 1 + \frac{1}{n(n-1)}.$$

In particular, for $n = 3$,

$$\Delta D(3 \to 2) = 1 + \frac{1}{3 \times 2} = \frac{7}{6}.$$

This numerical example is intended to aid the intuition of "irreversibility associated with projection" and is not used to establish the separation theorem below.

2.6 Summary of This Section

- MAG is a semantics (definition lattice) that translates computational concepts into group theory.
- The Krohn–Rhodes decomposition theorem provides the half-century-old mathematical foundation for the correspondence between polytime circuits and solvable groups.
- By the definitions that polytime = solvability, NP-hard = A_5 kernel, and solve = isomorphism, the separation theorem arises from "structural incompatibility" rather than "computational trickery."
- In the next section, we directly derive P $\neq$ NP in MAG using Lemma 2.7 and 2.8.

3 P $\neq$ NP in MAG (Main Theorem)

In this section, under the MAG semantics introduced in the previous section, P $\neq$ NP **in MAG**. The conclusion is not based on the skill of the calculation method, but on **structural incompatibility between solvable and non-solvable kernels (A_5)** is inevitably derived from.

Remark 3.1 (Important Note about the Positioning of the Theorem)**.** This theorem is a logical consequence of the definition of MAG, where polytime = solvable and NP-hard = unsolvable, and the proof itself is short. This is intentional by design. The main contribution of this paper is not the proof of Theorem 3.1, but (1) the presentation of a semantic perspective that captures the essence of P vs NP as a solvability barrier, and (2) the design of an architecture that rigorously formalizes the assumptions necessary for connecting with the standard model as a `TranslationInterface`. Theorem 3.1 serves as a proof that this architecture is logically consistent.

3.1 Master Theorem (Within MAG, $\mathrm{P} \neq \mathrm{NP}$)

The brevity of the following proof is intentional—it demonstrates that under MAG semantics, separation follows directly from basic group theory, without complex argument. The work lies in the semantic design, not the proof technique.

Theorem 3.2 (Separation Within MAG (Theorem 3.1)). *The proposition* $\mathrm{P} = \mathrm{NP}_{(MAG)}$ *defined by MAG does not hold:*

$$\neg\big(\mathrm{P} = \mathrm{NP}_{(MAG)}\big).$$

Lean support: `MAG_P_neq_NP : ¬ MAG_P_equals_NP`

Remark 3.3 (CFSG Foundation of the Universal Barrier (Remark 4.7.1)). The A_5 universal barrier reflects a fundamental fact established by CFSG: there exists a *structural gap* between solvable groups and the first non-solvable simple group:

1. **Solvable world**: All groups built from cyclic groups $\mathbb{Z}_p$ via extensions
2. **Gap**: No non-solvable simple group with order < 60 exists
3. **Non-solvable world**: Begins at A_5 (order 60)

This gap is not accidental but a consequence of the complete classification. Combined with the Krohn–Rhodes hierarchy (Section 2.1.1), this establishes that the solvability barrier is the *unique minimal* algebraic obstruction to $\mathrm{P} = \mathrm{NP}$ under MAG semantics.

3.2 The Core of the Proof: Invariance of Isomorphism and Solvability

The proof depends only on two points:

1. **Isomorphism preserves solvability (Lemma 2.7)**: If there exists a group isomorphism $A(a) \simeq G(p)$, then $A(a)$ is solvable $\Leftrightarrow$ $G(p)$ is solvable.
2. A_5 **is unsolvable (Lemma 2.8)**: $\neg(A_5$ is solvable$)$.

In addition, from the definition of MAG:

- p is NP-hard (MAG) if $G(p)$ is unsolvable.

Remark 3.4 (Isolation by "Nuclear Propagation" Rather than "Instantaneous by Definition" (Supplement 3.2.1)). Because the separation in MAG is short, one might get the impression that "$\mathrm{P} \neq \mathrm{NP}$ is immediately apparent just by expanding the definition." However, in the Core formalism in this paper, NP-hard is defined as "unsolvability," and then applied to a specific unsolvable example (A_5) to obtain a contradiction. Therefore, the separation is not a restatement of the definition, but rather a conflict between the mathematical fact that A_5 is unsolvable and the conservation law of solvability preservation by isomorphism.

3.3 Proof Sketch

Assuming that $\mathrm{P} = \mathrm{NP}$ in MAG, there exists a MAG-polytime algorithm that solves any MAG-NP-hard problem. In particular, if we take the example $p := A_5$ where A_5 is in the group (A_5 is unsolvable, so MAG-NP-hard), there exists an `alg` such that:

- $A(\mathrm{alg})$ is a solvable group,
- $A(\mathrm{alg}) \simeq A_5$ holds.

But since solvability is preserved by isomorphism, A_5 must also be solvable, which contradicts the unsolvability of A_5.

Therefore, $\mathrm{P} = \mathrm{NP}$ does not hold in MAG. □

3.4 Lean Formalization (Reader's Guide)

A Lean implementation of this main theorem conceptually consists of the following:

- `IsMAGPolyTime a` defined by `IsSolvable(A(a))`
- `IsMAGNPHard p` is defined by ¬ `IsSolvable(G(p))` (A_5 is used as a concrete witness)
- `Solves a p` defined by `Nonempty (A(a) ≃* G(p))`
- Define `MAG_P_equals_NP` as the quantified proposition that "for NP-hard p, there exists a polytime a that solves it."
- The proof is:
  - A_5 Unsolvability (Mathlib)
  - Preservation of solvability by group isomorphism
  - Combined, we show that ¬ `MAG_P_equals_NP`

The reader can check the existence of the main theorem directly from the entry file.

3.5 Summary

If we adopt the definition of MAG (polytime = solvable, NP-hard = unsolvable, solve = isomorphism), separation is reduced to "incompatibility of solvability of groups" rather than "computational complexity evaluation." In the next section, we will connect this pattern of "unsolvability does not translate into a solvable description system" to barrier construction and depth argument using A_5 as a representative example.

4 A_5 Universal Non-identifiability Barrier

In the previous section, we showed that the separation in MAG (P $\neq$ NP) is necessarily derived from the "incompatibility between solvable and unsolvable kernels." In this section, we will consider the same structure more generally as "**Inherent limitations of descriptive systems (models, translations, projections)**" and formulate it as a universal barrier with A_5 as its core.

4.1 Motivation: Treating Separation as "Inidentifiability" Rather than "Computational Complexity"

In standard complexity theory, the P/NP difference is measured in terms of resources such as time, circuit size, etc. But from the MAG perspective, what is more fundamental is:

- Can the object (the deep structure of the problem) be identified in a certain description system (the solvable language/algorithm)?
- When information is lost through projection, translation, and abstraction, does the deep non-commutative kernel disappear or remain?
- If an unsolvable kernel remains, it cannot be isomorphically reconstructed in the solvable language, resulting in "apparent randomness" and "irreversibility."

4.2 Abstraction: Descriptive Systems and Deep Structures

Definition 4.1 (Description System/Language (Definition 4.1))**.** A description system (language, model, syntactic rules) is abstracted as a structure that includes groups (or group actions) generated from it. Intuitively, the "rules of composition of operations," "normalization," and "gradual decomposability" allowed by the description system are expressed in group theory.
Lean support: `SyntacticSystem`

Definition 4.2 (Deep Structure (Definition 4.2))**.** The deep structure of the object (the body of the problem, the kernel of the constraints, and the hidden symmetries) is abstracted as a

corresponding group. The focus here is whether the deep group contains an unsolvable kernel.
Lean support: `DeepStructure`

Definition 4.3 (Describing Possibility (Definition 4.3))**.** A description system S can describe (identify) a deep structure X if the deep group is realized in the system through a suitable translation (homomorphism, representation, encoding). The strongest form is "isomorphism," but in this section we use "existence of an injective homomorphism" as the minimal condition sufficient to produce a barrier.

- Describability: The kernel of X is on the side of S via **injective embedding**.

Lean support: `CanDescribe`

4.3 Central Lemma: Unsolvable Kernels Propagate via Embedding

Lemma 4.4 (Propagation of Unsolvable Kernels (Lemma 4.4))**.** *If a group H is unsolvable and there exists an injective homomorphism $H \hookrightarrow G$, then G is also unsolvable:*

$$\big(H \textit{ unsolvable}\big) \ \wedge \ (H \hookrightarrow G) \quad \Longrightarrow \quad (G \textit{ unsolvable}).$$

Lean support: *`nonsolvable_of_injective_from_nonsolvable`*

Reason (intuition): If G is solvable, then its subgroup H must also be solvable (solvability is inherited in subgroups), but H is unsolvable, which is a contradiction.

4.4 Universal Barrier with A_5 as the Core

Definition 4.5 (Containing the A_5-kernel (Definition 4.5))**.** A deep structure X contains an A_5-kernel if there is an injective homomorphism of A_5 in the corresponding deep group $G(X)$:

$$A_5 \hookrightarrow G(X).$$

Theorem 4.6 (A_5 Gives the Universal Barrier (Theorem 4.6))**.** *For any description system S, if S is "solvable", then it is not possible to identify (describe) a deep structure X that contains an A_5-kernel:*

$$S \textit{ solvable} \quad \Longrightarrow \quad \neg\mathrm{CanDescribe}(S, X) \quad (\textit{where } A_5 \hookrightarrow G(X)).$$

Lean support: *`A5_Universal_Barrier`, `solvable_cannot_describe_A5`*

Skeleton of the proof:

1. A_5 is unsolvable (a known fact).
2. If $A_5 \hookrightarrow G(X)$, then $G(X)$ is unsolvable by Lemma 4.4.
3. On the other hand, if the description system S is solvable and X can be identified, then the deep group is realized in the system (or at least is reproduced in an isomorphic or equivalent form), and an unsolvable kernel appears in the system.
4. However, the system is solvable, so it does not contain unsolvable kernels.

As a result, the A_5-kernel becomes a universal barrier that sets the "limits of solvable description systems."

Remark 4.7 (Krohn–Rhodes Interpretation of the A_5 Barrier)**.** The A_5 barrier has a precise characterization in Krohn–Rhodes theory [5, 6]: A_5 is the smallest non-solvable simple group, and thus the minimal "building block" that cannot appear in any decomposition of a solvable automaton.

This means:

- Any machine whose Krohn–Rhodes decomposition includes only solvable simple groups cannot simulate a machine containing A_5.

- The group complexity hierarchy is strict: lower-complexity machines cannot cover (simulate) higher-complexity machines.

Our theorem `nonsolvable_of_injective_from_nonsolvable` formalizes precisely this hierarchical inviolability. The "A_5 Universal Barrier" (Theorem 4.6) is therefore not an ad hoc construction but a restatement of the fundamental limitations established by Krohn–Rhodes theory in the 1960s.

4.5 Relationship with Separation Within MAG

The MAG P $\neq$ NP in Section 3 can be understood as a special case of the universal barrier above.

- Description system: MAG-polytime algorithm (the side expressed by the solvable group)
- Deep structure: MAG-NP-hard problem (A_5-nucleus side)
- "Solve": Completely identify the deep structure by isomorphism

In this case, the universal barrier Theorem 4.6 gives us that "a solvable description system cannot identify the A_5-kernel", which directly means that "P = NP is impossible within MAG".

4.6 Connection with Numerical Gap (Remark)

The numerical gap $\Delta D = 7/6$ is an intuitive aid to the barrier in this section. For details, see the numerical examples in Section 2.5. It is not essential for the separation-barrier theorem to hold.

4.7 Vanishing Depth Barrier with Commutator Trees (Theorem 4.7)

To bring the universal barrier closer to Barrington-style syntax, we define a commutator tree of depth n as follows: depth 0 is a variable, and depth $n+1$ is the commutator $[t, u]$ of a dyadic tree t, u of depth n.

Theorem 4.8 (Solvable Vanishing Depth (Theorem 4.7)). *For any group G and assignment ρ, the evaluation value* $\mathrm{eval}_\rho(t)$ *of a commutator tree of depth n belongs to* $\mathrm{derivedSeries}(G, n)$*. Thus, if G is solvable and* $\mathrm{derivedSeries}(G, k) = \bot$ *for some k, then for any $n \geq k$ the commutator tree of depth n collapses to the identity (vanishing depth).*

In contrast, A_5 has no finite stage where the derived sequence falls into $\bot$, and so in this sense does not have a finite vanishing depth.

Lean support: *`MAG.Core.BarringtonBarrier` (`solvable_vanishing_depth`, `A5_no_vanishing_depth`)*

4.8 Summary of This Section

- Non-solvable kernels "propagate" via injective embedding and do not fall into decomposable systems.
- The smallest unsolvable kernel (A_5) provides a universal limit (barrier) for solvable description systems.
- The Krohn–Rhodes decomposition theorem confirms that this barrier reflects a fundamental hierarchy in automata theory.
- P $\neq$ NP within MAG is positioned as a direct consequence of this universal barrier.

5 Translation Interface and Bridge Theorem

In this section, we clarify the relationship between MAG semantics (group-theoretic model) and external computational models (any abstract model, including standard TMs and circuit models). The key point here is not to equate standard models with MAG, but to **extract the**

assumptions necessary for identification as `TranslationInterface` (axiom bundle), and under that assumption, the separation within MAG is transferred to an external model.

5.1 Abstract Computational Model `ComputationalModel`

Definition 5.1 (Abstract Computational Model (Definition 5.1))**.** For any computational model M, we define a structure with the following vocabulary, abstracted as `ComputationalModel M`:

- Problem (language) type: `Problem M`
- Algorithm type: `Algorithm M`
- "Algorithms solve problems" relationship: `solves : Algorithm M → Problem M → Prop`
- "polytime" determination: `isPolyTime : Algorithm M → Prop`
- "NP-hard" determination: `isNPHard : Problem M → Prop`

Lean support: `class ComputationalModel (M)`

5.2 Translation Interface `TranslationInterface`

The necessary conditions for translating an external model M into MAG (group theory semantics) are summarized as the following three conditions, formulated as `TranslationInterface M`.

Definition 5.2 (Translation Interface (Definition 5.2))**.** The computational model M satisfying `TranslationInterface` conceptually means the following:

1. **polytime ⇒ Solvability**: The polytime algorithm in M can be expressed as a solvable group on the MAG side.
$$\text{isPolyTime}(a) \;\Rightarrow\; \text{(the correspondence group is solvable).}$$
2. **NP-hard ⇒ A_5 kernel (injective embedding)**: NP-hard problems in M have a symmetry group containing an injective embedding of A_5 on the MAG side.
$$\text{isNPHard}(p) \Rightarrow \exists f : A_5 \to G(p),\ \text{Injective}(f).$$
3. **solves ⇒ isomorphism**: The "solving" in M gives a group isomorphism on the MAG side.
$$\text{solves}(a, p) \;\Rightarrow\; A(a) \simeq G(p).$$

Lean support: `class TranslationInterface (M)`

Remark 5.3 (Justification for Condition 1: Krohn–Rhodes Guarantee)**.** The condition "polytime ⇒ solvable" is not an arbitrary algebraic assumption but follows from the Krohn–Rhodes decomposition theorem [5]. Standard Boolean gates have the following group-theoretic character:

- **AND/OR gates**: Aperiodic (trivial group component)
- **NOT gate**: $\mathbb{Z}_2$ (solvable)
- **Composition**: Wreath products of solvable components remain solvable

Therefore, any circuit composed of standard gates has a Krohn–Rhodes decomposition using only solvable simple groups. This provides the **mathematical guarantee** that standard polynomial-time circuits correspond to solvable group structures.

Lean support: See `Support/Circuits/KrohnRhodes.lean` for the formalization of gate-level solvability preservation.

Remark 5.4 (Rationale for Condition 2: A_5 Kernel)**.** This condition is not an arbitrary algebraic requirement but a consequence of the standard complexity hierarchy. Since the word problem for the symmetric group S_5 is complete for NC^1 (Barrington [1]), and $\mathrm{NC}^1 \subseteq \mathrm{NP}$, any problem that is NP-hard must computationally subsume the structure of S_5 via polynomial-time reduction.

Under the assumption that reductions preserve the underlying syntactic group structure (*algebraic monotonicity*), the S_5 kernel—and specifically its unsolvable A_5 subgroup—must appear as an injective embedding in the symmetry group of any NP-hard problem. Thus, this interface condition formally captures the inevitable presence of the A_5 structure within NP via Barrington's theorem.

Note that this algebraic monotonicity assumption is the key bridge condition that remains to be fully formalized; it is isolated within the `TranslationInterface` as condition (2).

Remark 5.5 (Structural Realizability vs. Functional Equivalence)**.** We explicitly acknowledge that defining $\mathrm{Solves}(a, p)$ as a group isomorphism ($A(a) \simeq G(p)$) is a stronger condition than standard functional equivalence (i.e., that algorithm a merely produces the correct output for problem p).

However, this definition is intentional. It captures the notion of *Structural Realizability*: can the algorithm's internal operation group structurally emulate the symmetry group of the problem?

By enforcing isomorphism, we isolate the algebraic barrier: even if a solvable circuit could functionally compute the output bits, it cannot do so by structurally mirroring the non-solvable A_5 kernel. This distinction shifts the P vs NP question from "function computation" to "structure representation," where the separation becomes algebraically provable. This is analogous to how Galois theory shifts the question of "solvability by radicals" from "finding roots" to "group structure."

5.3 P = NP Proposition on the External Model

Definition 5.6 (P = NP on Model M (Definition 5.3))**.**
Using only the vocabulary of `ComputationalModel` M, we define "P = NP" on the model M as follows:

$$\mathrm{P} = \mathrm{NP}\ (\text{on } M) \quad :\Longleftrightarrow \quad \forall p\ (\mathrm{isNPHard}(p) \Rightarrow \exists a\ (\mathrm{isPolyTime}(a) \wedge \mathrm{solves}(a, p))).$$

Lean support: `def Standard_P_equals_NP (M) : Prop := ...`

5.4 Bridge Theorem

Theorem 5.7 (Bridge Theorem (Theorem 5.4))**.**
For a computational model M satisfying `ComputationalModel` *and* `TranslationInterface` *M:*

$$\neg(\mathrm{P} = \mathrm{NP}\ \textit{(on } M)).$$

Lean support: *`Bridge_P_neq_NP` and `Bridge_P_neq_NP'`*

Skeleton of the proof:

- Assume `Standard_P_equals_NP`. Then for any NP-hard problem p, there exists a polytime algorithm a that solves it.
- By `TranslationInterface` condition (1), the correspondence group in a is solvable.
- At the same time, condition (3) gives a group isomorphism correspondence, so the problem group is also solvable.
- However, condition (2) injectively embeds A_5 in NP-hard problems, and hence the problem set is unsolvable.
- Due to the contradiction between solvability and insolvability, `Standard_P_equals_NP` is denied.

This is the logic of Section 3 simply "transported" to the external model.

Remark 5.8. The vanishing depth lemma for commutator trees unconditionally proves that the solvable side cannot hold "deep commutator syntax" via the derivation sequence. Therefore, the intuition that a solvable description system cannot identify an unsolvable kernel as long as a translation from the standard model to a Barrington-type commutator syntax is given is formally reinforced.

5.5 End-to-End Verification Using Toy Models

The bridging theorem is abstract in the sense that if a translation IF is given, then separation follows. We construct a toy computational model (BarringtonToy) that actually satisfies `TranslationInterface` and verify using Lean that separation decreases from end to end.

Construction 5.9 (BarringtonToyModel (Configuration 5.5))**.** In the toy model, concepts are "matched by definition" as follows:

- `isPolyTime a` is defined as "a set of algorithms is solvable."
- `isNPHard p` is defined as "A_5 is injectively embeddable in the problem group" (which leads to unsolvability).
- `solves a p` is defined as "a group isomorphism exists."

Then, the three conditions of `TranslationInterface` are satisfied by the definition expansion (and the existence of unsolvable examples).

Lemma 5.10 (Concrete Embedding of $A_5 \hookrightarrow S_5$ (Lemma 5.6))**.** *As an NP-hard example of toy, S_5 (`Perm (Fin 5)`) and by natural inclusion*

$$A_5 \hookrightarrow S_5.$$

In Lean this is constructed as a concrete homomorphism and an injectivity.

Theorem 5.11 (BarringtonToy_P_neq_NP (Theorem 5.7))**.** *Under the toy configuration above,*

$$\neg(\mathrm{P} = \mathrm{NP} \textit{ (on BarringtonToyModel)}).$$

Lean support: *`BarringtonToy_P_neq_NP`*

Remark 5.12 (Significance)**.** This shows that Bridge is not just an "abstract hypothetical play" but actually works end-to-end in its formalization. This also serves as confirmation to the reviewer that the assumptions are well-formulated.

5.6 Relationship Between Standard P and NP (Scope of This Paper)

Our formalism clearly separates:

- Unconditionally proven nuclei: intra-MAG separation (Section 3) and universal barrier (Section 4)
- Conditional Transfer of Assertions: Bridge Theorem (this section)
- Remaining issues: Standard TM/circuit model `TranslationInterface` configuration that satisfies the above (managed separately as a Blueprint)

Therefore, the "solution" of this paper is **not an unconditional solution of standard problems**, but rather **extract translation assumptions as patterns and strictly present the boundary between the formalized core and the unformalized core**.

5.7 Summary of This Section

- `TranslationInterface` localizes the assumptions necessary to connect the standard model to the MAG semantics.
- The Krohn–Rhodes decomposition theorem provides the theoretical justification for Condition 1 (polytime ⇒ solvable).
- Under this assumption, the intra-MAG separation can be transferred to any model (Bridge theorem).
- The toy model allows us to formally verify that the Bridge works end-to-end.

6 Conclusion

This paper aims to avoid limiting the discussion of P vs NP to the "measurement of computational resources" alone. From the standpoint of **making the semantics of computation explicit**, MAG (group-theoretic semantics) was introduced, and its internal separation ($\mathrm{P} \neq \mathrm{NP}$) was fully formalized using Lean. The central observation is that, just as Galois theory normalized the question "Can it be solved by radicals?" to the structural question "Is it a solvable group?", the moment P vs NP becomes clearer, "in which description system is it considered to be 'solvable'?", **structural incompatibility between solvable and non-solvable kernels (A_5)** becomes the problem, and separation becomes inevitable.

The mathematical foundation of our approach is anchored in the Classification of Finite Simple Groups (CFSG), which establishes A_5 as the smallest non-solvable simple group. This is not an arbitrary choice but a mathematically determined boundary: the Feit–Thompson Theorem guarantees that non-solvability requires even order, and CFSG confirms that 60 (the order of A_5) is the minimum such order. Our separation theorem thus inherits the full weight of 20th-century finite group theory.

The results of this paper can be organized into three layers:

1. **Unconditional nucleus (separation within MAG)**: We introduce MAG semantics as a definitional lattice where polytime = solvability, NP-hard = A_5 kernel, and solve = group isomorphism. $\mathrm{P} = \mathrm{NP}_{\mathbf{(MAG)}}$ **does not hold**. Here, we have used only the basic facts of the unsolvability of A_5 and the preservation of solvability by group isomorphism.
2. A_5 **Universal Barrier**: As a general principle behind separation, we abstract the fact that unsolvable kernels propagate by injective embedding and do not fall into solvable description systems, and formulate it as a universal barrier with kernel A_5. The Krohn–Rhodes decomposition theorem [5] confirms that this barrier reflects the fundamental complexity hierarchy in automata theory. Separation within MAG is positioned as its minimal instantiation.
3. **Localization of translation assumptions (Bridge) and toy verification**: Regarding the relationship with standard computational models (TM/circuits, etc.), we do not equate them, but make necessary assumptions as `TranslationInterface`. We then showed that under this assumption, the separation can be transferred to the external model. Furthermore, we provided an example that works end-to-end in the toy computation model, and confirmed that the bridging form is not empty theory.

Artifacts Overview

The formalization of this paper consists of the following main modules:

A.1 MAG Model and Separation Theorem (Core)

- `Core/MAG_P_neq_NP.lean`: MAG semantics definition lattice (polytime/NP-hard/solve) and main theorem `MAG_P_neq_NP`.

A.2 Universal Barrier (A_5)

- `Core/A5_Barrier.lean`: Abstracting the propagation of unsolvable kernels, we give a universal barrier theorem with kernel A_5.

A.3 Bridge and Toy

- `Core/Bridge/Theorem.lean` and `Core/Bridge/ToyInstantiation.lean`: `ComputationalModel` and `TranslationInterface` definition, Bridging Theorem `Bridge_P_neq_NP`, and toy `BarringtonToy_P_neq_NP`.

A.4 Assistance

- Barrington/circuit synthesis/syntactic group related auxiliary modules.
- `Core/BarringtonBarrier.lean`: Comparison of the vanishing depth of solvable groups using commutator trees (Barrington-type reinforcement) and the A_5 side.
- `Support/Circuits/KrohnRhodes.lean`: Formalization of gate-level solvability preservation based on Krohn–Rhodes decomposition theory.

Reproducibility

The theorems in this paper are formalized using Lean, and the artifacts will be provided in a public repository. Reviewers and readers can machine-verify the main theorems by following the steps below.

R.1 Build Procedure (Outline)

1. Prepare Lean (elan) and Lake.
2. Execute `lake build` directly under the repository.
3. By type-checking the entry file `Main.lean`, the main theorem can be confirmed.

R.2 Entry File and Main Theorem

In the entry file `Main.lean`, at least the following theorems exist and can be confirmed by `#check`:

- **MAG internal separation**: `MAG.Core.MAG_P_neq_NP`
- **Bridging theorem**: `MAG.Core.Bridge.Bridge_P_neq_NP`
- **Toy verification**: `MAG.Core.Bridge.BarringtonToy_P_neq_NP`

R.3 Separation of Scope and Deliverables (Core / Blueprint)

The deliverables contributing to the argument (**Core**) does not rely on `sorry` or additional axioms, whereas the Blueprint includes provisions for concrete translation into the Standard Model, but does not include them in its core claims.

Acknowledgments

The author would like to thank the Lean and Mathlib communities for their efforts in developing the tools and libraries that supported parts of this work. LaTeX typesetting, English refinement, and parts of the explanations were refined with the assistance of large-scale language models. All mathematical content, claims, and proofs have been verified and are the sole responsibility of the author.

Funding. This research received no external funding.

Conflict of Interest. The author declares no conflict of interest.

Data Availability. The Lean formalization artifacts are available at: `https://github.com/nuu/MAG-PneqNP`

References

[1] D. A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In *Proc. 18th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–5, 1986.

[2] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 1971.

[3] É. Galois. *Oeuvres mathématiques.* Journal de mathématiques pures et appliquées, 11:381–444, 1846.

[4] F. Klein. Vergleichende Betrachtungen über neuere geometrische Forschungen (A comparative review of recent researches in geometry). *Mathematische Annalen*, 43:63–100, 1893. Originally published as a separate pamphlet, Erlangen, 1872.

[5] K. Krohn and J. Rhodes. Algebraic theory of machines. I. Prime decomposition theorem for finite semigroups and machines. *Transactions of the American Mathematical Society*, 116:450–464, 1965.

[6] J. Rhodes and B. Steinberg. *The q-theory of Finite Semigroups.* Springer Monographs in Mathematics, 2009.

[7] L. de Moura and S. Ullrich. The Lean 4 theorem prover and programming language. In *Proc. 28th International Conference on Automated Deduction (CADE)*, pages 625–635, 2021.

[8] The mathlib Community. The Lean mathematical library. In *Proc. 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP)*, pages 367–381, 2020.

[9] J. J. Rotman. *An Introduction to the Theory of Groups.* Springer, 4th edition, 2012.

[10] S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[11] W. Feit and J. G. Thompson. Solvability of groups of odd order. *Pacific Journal of Mathematics*, 13(3):775–1029, 1963.

[12] D. Gorenstein, R. Lyons, and R. Solomon. *The Classification of the Finite Simple Groups.* Mathematical Surveys and Monographs, American Mathematical Society, 1994–2018.