Structural Dynamics of Polynomial Solvability: Layered State Spaces and Polynomial Navigability

Alexey A. Nekludoff

AstraVerge Research

E-mail: an@astraverge.org
ORCID: 0009-0002-7724-5762

06 May 2026

Abstract

This note introduces a structural and geometric viewpoint on polynomial-time solvability based on the dynamics of computational state spaces. Rather than treating P and NP primarily through the opposition between search and verification, we focus on the evolution, compression, and navigability of layered state spaces arising during computation.

The paper is exploratory in character and intended as an introductory step toward a broader structural approach to complexity theory. Its purpose is not to propose a completed alternative foundation of complexity theory, but to develop a preliminary vocabulary for discussing mechanisms of global structural collapse in computation.

A computational formalism is represented as a triple $F = (M, R, D)$ consisting of a class of computational models, a resource scale, and a class of admissible descriptions. This distinction is used to separate genuinely polynomially realizable structures from representations whose apparent efficiency depends on stronger primitive operations or alternative resource assumptions.

The central technical notion is that of a structural realization of a language by a state system with divergence, bounded frontier, and completion. We show that any language admitting such a realization lies in P. The framework is illustrated on Horn-SAT, 2-SAT, bipartite matching, and 3-SAT, emphasizing different forms of structural collapse and different obstructions to polynomial navigability.

The resulting interpretation is that polynomial solvability is not merely the absence of search, but the existence of global mechanisms that prevent uncontrolled expansion of the live computational frontier. The concepts introduced here are intended primarily as organizational and programmatic tools for future development rather than as a completed structural theory.

Keywords: computational complexity; polynomial solvability; layered state spaces; polynomial navigability; structural realizations; structural collapse; state-space dynamics; proof complexity; SAT; computational formalisms; admissible compression; robust simulation; layered dynamics; frontier growth; global structure

Contents

1 Introduction

The classical formulation of P versus NP asks whether every language whose positive instances admit polynomially checkable certificates is also decidable by a deterministic polynomial-time algorithm [1], [2], [3], [4], [5]. The standard informal interpretation is that NP captures efficient verification, while P captures efficient solution.

This interpretation is operational, but only partially structural. Both verification and search are computations over evolving state spaces, and many known polynomial-time algorithms succeed not by avoiding search altogether, but by organizing exponentially many potential trajectories into globally manageable structures.

The present note is exploratory in character. Its goal is not to propose a completed alternative foundation of complexity theory, nor to resolve the classical P versus NP question. Instead, the aim is to introduce a structural vocabulary and a geometric viewpoint that may be useful for describing why certain computational state spaces remain polynomially navigable while others appear to resist global compression.

We therefore view computation not primarily as the traversal of individual trajectories, but as the evolution of entire layers of states. From this perspective, polynomial-time solvability appears to arise when the state space admits representations in which:

- false trajectories diverge early,
- the live frontier remains polynomially controlled,
- and surviving states can be completed or refuted efficiently.

We formalize these notions through the concept of a *structural realization*. The resulting framework is intentionally mixed in style: some notions are introduced formally, while others are presented at a schematic or geometric level whose role is primarily explanatory and programmatic.

A central perspective of the paper is geometric. We introduce layered state spaces (Y_t) and ask whether exponentially large layers can nevertheless admit polynomially realizable representations and dynamics. This leads to a guiding principle:

> There is no free compression of computation: a polynomial representation of an exponentially large state space is meaningful only if the induced operations remain polynomially realizable.

Within this viewpoint, the apparent difficulty of problems such as 3-SAT can be reformulated not as the mere existence of exponentially many trajectories, but as the absence – or unknown existence – of a global collapse mechanism that keeps the layered dynamics polynomially navigable.

In parallel, we make explicit the dependence of complexity-theoretic statements on the underlying computational formalism. We represent a formalism as a triple

$$F = (M, R, D),$$

consisting of a computational model, a resource scale, and a class of admissible descriptions.

This distinction is used not to redefine the classical P versus NP problem, but to separate two situations:

- structures whose operations admit polynomial simulation inside standard robust computational models;
- and structures whose apparent efficiency depends on stronger primitive operations or alternative resource scales.

From this perspective, compact representation alone is insufficient: the computational meaning of a global structure depends on the realizability of operations over that structure.

The contribution of this note is therefore primarily conceptual and organizational rather than complexity-separating. In particular, the paper aims to:

- introduce a structural language based on divergence, frontier, completion, and layered dynamics;
- relate polynomial solvability to forms of global structural collapse;
- connect these notions to proof complexity, symbolic representations, and state-space methods;
- and outline a possible direction for a more systematic structural theory of computational complexity.

2 Computational formalisms

Definition 1 (Computational formalism)**.** *A computational formalism is a triple*

$$F = (M, R, D),$$

where M is a class of computational models, R is a resource scale, and D is a class of admissible descriptions of algorithms, objects, or state structures.

The standard formalism takes M to include Turing machines, RAM-like models with reasonable word size, or Boolean circuits; R to be time, number of elementary operations, or circuit size/depth; and D to be finite descriptions such as finite programs or finite circuits. Different choices inside this robust family are polynomially inter-simulable in the usual sense.

Definition 2 (P_F and NP_F)**.** *For a formalism $F = (M, R, D)$, let P_F be the class of languages decidable by models in M using resources bounded by a polynomial in the input length, according to R and using descriptions in D.*

Let NP_F be the class of languages whose positive instances admit certificates of polynomial length, with a verifier in M using polynomial resource according to R and descriptions in D.

Definition 3 (Robust class of formalisms)**.** *A class $\mathfrak{F}_{\mathrm{rob}}$ of formalisms is robust if any two models in the class simulate one another with at most polynomial overhead in the selected resource scale, and primitive operations do not hide super-polynomial work as a single elementary step.*

Proposition 1 (Invariance inside a robust class)**.** *If $F_1, F_2 \in \mathfrak{F}_{\mathrm{rob}}$, then*

$$\mathrm{P}_{F_1} = \mathrm{NP}_{F_1} \quad \Longleftrightarrow \quad \mathrm{P}_{F_2} = \mathrm{NP}_{F_2}.$$

Proof sketch. By robustness, computations in either formalism can be simulated in the other with polynomial overhead. Polynomial-time deciders and polynomial-time verifiers are therefore preserved under the simulation. Hence equality or separation of the corresponding classes is invariant inside the robust family. □

Remark 1. *If one extends F by allowing primitives that operate on exponentially large layers as a single resource-bounded step, then one has changed the resource scale or model class. Such an extension may be legitimate as a new theory of computation, but it does not answer the classical* P *versus* NP *question unless the new primitives are simulable in a robust formalism with polynomial overhead.*

3 Structural realizations

We now define a structural notion of polynomial solvability. The goal is to express when search does not explode because the space of live states has a polynomially controlled dynamics.

Definition 4 (State system for an input)**.** *For a language $L \subseteq \{0,1\}^*$ and input x, a state system consists of a finite or finitely represented state space $\mathcal{S}_x$, an initial state $s_0(x) \in \mathcal{S}_x$, a transition relation $\to_x \subseteq \mathcal{S}_x \times \mathcal{S}_x$, and a set of accepting states $\mathcal{A}_x \subseteq \mathcal{S}_x$.*

Definition 5 (Trajectory)**.** *A trajectory is a finite sequence*

$$\tau : s_0 \to_x s_1 \to_x \cdots \to_x s_k.$$

A trajectory is accepting if some $s_i \in \mathcal{A}_x$.

Definition 6 (False trajectory)**.** *A trajectory ending in s_k is false if no continuation from s_k reaches an accepting state.*

Definition 7 (Cutoff procedure)**.** *A cutoff procedure is an algorithm $C(x,s) \in \{0,1\}$ such that if $C(x,s) = 1$, then no accepting state is reachable from s.*

Definition 8 (Divergence)**.** *A state system has polynomial divergence if there exists a polynomial q such that every false trajectory reaches some state s_i with $C(x,s_i) = 1$ for some $i \le q(|x|)$.*

Definition 9 (Polynomial frontier)**.** *A state system has polynomial frontier if there exists a polynomial r such that the number of states reachable from $s_0(x)$ within $q(|x|)$ steps and not cut off by C is at most $r(|x|)$.*

Definition 10 (Completion)**.** *A completion procedure is an algorithm $E(x,s)$ that, for any state s with $C(x,s) = 0$, either constructs an accepting state reachable from s or correctly reports that no accepting continuation exists. It is polynomial if its running time is polynomial in $|x|$.*

Definition 11 (Structural P-realization)**.** *A language L has a structural P-realization if for every input x there is a state system $(\mathcal{S}_x, s_0, \to_x, \mathcal{A}_x)$ with polynomially encoded states, polynomially computable transitions, polynomially enumerable frontier, a polynomial cutoff C, polynomial divergence, polynomial frontier, and a polynomial completion procedure E, such that*

$$x \in L \quad \Longleftrightarrow \quad \text{some trajectory from } s_0(x) \text{ reaches } \mathcal{A}_x.$$

Theorem 1 (Structural realization implies polynomial decidability)**.** *If a language L has a structural P-realization, then $L \in \mathrm{P}$.*

Proof. On input x, enumerate all non-cutoff states reachable from $s_0(x)$ up to depth $q(|x|)$. By the polynomial frontier condition, there are at most $r(|x|)$ such states. For each such state, run the completion procedure $E(x,s)$. If any call returns an accepting state, accept. If all calls correctly report that no accepting continuation exists, reject.

The number of states is polynomial, each state is polynomially encoded, transitions and cutoff tests are polynomially computable, and each completion call is polynomial. Hence the total computation is polynomial in $|x|$. □

Remark 2. *The theorem is intentionally simple: its interest is not that it proves a surprising inclusion, but that it isolates three structural causes of polynomial behavior: false trajectories die early, the live frontier remains polynomial, and live states can be completed or refuted efficiently.*

$$\text{divergence} \rightarrow \text{frontier} \rightarrow \text{completion}$$

Figure 1: Structural realization pipeline.

4 Geometry of State Spaces

The notion of structural realization introduced above can be interpreted geometrically. Instead of viewing computation as a sequence of individual trajectories, we consider the evolution of entire layers of states.

4.1 Layered state spaces

For a given input x, let S_x be the associated state space. We define a *layer* at depth t as the set

$$Y_t = \{s \in S_x \mid s \text{ is reachable from } s_0(x) \text{ in } t \text{ steps}\}.$$

A layered representation is a sequence $(Y_0, Y_1, \dots)$ such that

$$Y_{t+1} = \Phi(Y_t),$$

for some transition operator Φ induced by the transition relation.

The size of Y_t may be exponential in t, even when $t = O(|x|)$.

4.2 Navigability versus size

The central question is not the cardinality of Y_t, but whether it admits a representation and dynamics satisfying:

- Y_t can be represented in size $poly(|x|)$,
- $\Phi(Y_t)$ can be computed in $poly(|x|)$,
- the existence of an accepting state in Y_t can be decided in $poly(|x|)$.

This separates existence from computation:

- the layer Y_t may contain exponentially many states,
- but polynomial solvability requires that Y_t be navigable in polynomial time.

4.3 Aggregation and structural compression

In polynomial-time solvable problems, exponential families of trajectories are often aggregated into a single structured object:

- in Horn-SAT, a monotone closure replaces branching,
- in 2-SAT, implication graphs compress assignments,
- in bipartite matching, layered graphs represent sets of augmenting paths.

In each case, the effective frontier is not a set of individual states, but a structured object of polynomial size.

4.4 No free compression principle

A fundamental constraint emerges:

> Any compression of an exponentially large state space into a polynomial representation must either admit polynomial-time operations or implicitly encode super-polynomial computational power.

In other words, there is no free compression of computation: compression must be paid for either by the resource scale or by the strength of the allowed operations.

Lemma 1 (No free compression)**.** *Let Y_t be a layer with $|Y_t| = 2^{\Omega(n)}$. If there exists a representation $\mathcal{R}(Y_t)$ of size poly(n) such that all operations (transition, filtering, acceptance queries) are computable from $\mathcal{R}(Y_t)$ in poly(n), then these operations admit a polynomial simulation in a robust formalism.*

Proof. By assumption, $\mathcal{R}(Y_t)$ has polynomial size and all required operations are polynomially computable. Therefore they can be simulated in a standard model with polynomial overhead. Any failure of such a simulation implies that the operations implicitly encode super-polynomial computation, contradicting robustness. □

4.5 Implications for 3-SAT

The difficulty of 3-SAT can be reformulated geometrically:

whether there exists a representation of the solution space such that the induced layered dynamics satisfies polynomial divergence, polynomial frontier, and polynomially realizable transition and completion operators.

While local structures (implication graphs, learned clauses) provide partial compression, no known representation yields a globally polynomially navigable state space for all instances.

4.6 From layered dynamics to structural realizations

We now formalize the connection between layered representations and structural P-realizations.

Definition 12 (Polynomially navigable layered dynamics)**.** *A layered state space (Y_t) is polynomially navigable if there exist:*

- *a representation of each layer Y_t of size poly$(|x|)$,*
- *a transition operator Φ such that $Y_{t+1} = \Phi(Y_t)$ is computable in poly$(|x|)$,*
- *a procedure that, given t, decides whether Y_t contains an accepting state in poly$(|x|)$,*
- *a polynomial bound $q(|x|)$ such that all relevant layers occur for $t \leq q(|x|)$.*

Theorem 2 (Layered dynamics implies structural realization)**.** *If a language L admits a polynomially navigable layered dynamics, then L has a structural P-realization.*

Proof. Given a layered representation (Y_t), define the state space S_x as the set of states that appear in layers Y_t for $t \leq q(|x|)$.

The transition relation is induced by Φ: a state $s' \in Y_{t+1}$ is reachable from $s \in Y_t$ if it arises from the transition operator.

Define the cutoff procedure $C(x, s)$ implicitly through the layered dynamics: states that do not persist in layers leading toward acceptance are eventually excluded by the evolution of the layers.

Polynomial divergence follows from the bounded depth $t \leq q(|x|)$.

Polynomial frontier follows from the polynomial size of each layer representation.

For completion, given a state $s \in Y_t$, the layered dynamics allows us to determine whether there exists an accepting continuation from some layer $Y_{t'}$ with $t' \geq t$, and, when such a continuation exists, to construct a corresponding accepting trajectory.

All operations are polynomial by assumption. Hence the resulting state system satisfies the conditions of a structural P-realization. □

4.7 Collapse versus expansion

The existence of a polynomially navigable layered dynamics requires a strong global property: exponential branching must collapse into a polynomially bounded representation.

We formalize this as follows.

Definition 13 (Polynomial collapse)**.** *A layered system exhibits polynomial collapse if there exists a representation $\mathcal{R}(Y_t)$ such that:*

- $|\mathcal{R}(Y_t)| = poly(|x|)$,
- $\mathcal{R}(Y_{t+1})$ *can be computed from* $\mathcal{R}(Y_t)$ *in* $poly(|x|)$,
- *membership or acceptance queries over* Y_t *can be decided using* $\mathcal{R}(Y_t)$ *in* $poly(|x|)$.

Proposition 2 (Collapse is necessary for polynomial navigability)**.** *If a layered dynamics is polynomially navigable, then it admits polynomial collapse.*

Proof. Polynomial navigability requires that each layer be representable in polynomial size and that all operations on it be computable in polynomial time. This directly induces a representation $\mathcal{R}(Y_t)$ satisfying the above properties. □

4.8 Types of structural collapse

The examples above suggest that polynomial solvability may arise from different mechanisms of collapse. We distinguish three basic types.

Definition 14 (Types of collapse)**.** *Let (Y_t) be a layered dynamics.*

- ***Monotone collapse****: the live state space evolves by a monotone closure operator, so that branching is replaced by a least fixed point.*
- ***Quotient collapse****: exponentially many states are identified by an equivalence relation whose quotient has polynomial size.*
- ***Aggregation collapse****: exponentially many alternative trajectories are represented by one polynomially sized aggregate object.*

Proposition 3 (Collapse mechanisms in standard polynomial problems)**.** *Horn-SAT exhibits monotone collapse, 2-SAT exhibits quotient collapse, and bipartite matching exhibits aggregation collapse.*

Proof. For Horn-SAT, forward chaining computes a least fixed point of forced true variables, eliminating branching by monotone closure.

For 2-SAT, assignments are compressed through the strongly connected components of the implication graph; the relevant structure is the quotient DAG of components.

For bipartite matching, the set of shortest augmenting paths is not enumerated individually but represented by a layered alternating graph. This graph aggregates a family of possible improvements into one polynomially sized object. □

4.9 Failure of layered dynamics for 3-SAT

We now examine why the layered framework, while successful for several polynomial-time problems, does not currently yield a polynomial realization for 3-SAT.

4.9.1 Layer growth

Let Y_t denote the set of states reachable after t branching steps in a natural search formulation of 3-SAT.

Even under aggressive propagation (unit propagation, clause learning), the number of distinct partial assignments that are not immediately refuted can grow exponentially in t. That is, there exist families of instances such that

$$|Y_t| = 2^{\Omega(t)}.$$

Thus, any representation of Y_t must either:

- have exponential size, or
- compress exponentially many states into a structured representation.

4.9.2 Limits of structural compression

Local mechanisms such as implication graphs and clause learning provide partial compression of the search space. However, they do not yield a representation of Y_t satisfying all requirements of polynomial navigability.

In particular, known proof complexity results show that certain families of unsatisfiable formulas require exponentially large refutations in resolution and related proof systems.

This implies that no cutoff procedure based solely on such local structures can eliminate all false trajectories in polynomial time.

4.9.3 Failure of polynomial frontier

As a consequence, the frontier condition fails:

- either the representation of Y_t becomes exponential,
- or the transition operator Φ implicitly performs super-polynomial work,
- or the procedure for deciding whether Y_t contains an accepting state is not polynomial.

Thus, the layered dynamics for 3-SAT is not known to be polynomially navigable.

4.9.4 A near-structure: CDCL states

A natural candidate for a structured layer in 3-SAT is given by CDCL-style states consisting of a partial assignment A, an implication graph G, and a database of learned clauses $\mathcal{C}$.

Local strength. Unit propagation and conflict analysis provide strong local mechanisms: conflicts are detected efficiently, and learned clauses prune parts of the search space.

Breakdown of divergence. While conflicts are found quickly, there is no guarantee that all false trajectories are cut off within $poly(|x|)$ steps. Known lower bounds in proof complexity imply that some inconsistent regions require super-polynomial refutations.

Breakdown of frontier. The number of distinct partial assignments consistent with current clauses can remain exponential. Clause learning reduces redundancy but does not ensure a polynomial bound on the live frontier.

Breakdown of completion. Determining whether a given partial state admits an accepting extension is equivalent to solving SAT on a residual formula, and no polynomial procedure is known.

Conclusion. CDCL provides a powerful *local* compression, but it does not induce a globally polynomially navigable layered dynamics.

4.9.5 Reformulation of the P vs NP question

The difficulty of 3-SAT can therefore be restated as follows:

Does there exist a representation of the layered state space (Y_t) such that:

- each Y_t has polynomial-size representation,
- the transition operator Φ is computable in polynomial time,
- the existence of an accepting state in Y_t is decidable in polynomial time,
- and the number of relevant layers is polynomially bounded?

An affirmative answer would yield a polynomially navigable layered dynamics and thus, by the previous theorem, imply $P = NP$.

Core obstruction. The obstruction is therefore not the absence of local structure, but the absence of a polynomial collapse of the layered dynamics.

4.10 A toy model: local consistency without global compression

We include a simple toy model illustrating why local consistency is insufficient for polynomial navigability.

Let the state space be the Boolean hypercube

$$S_n = \{0,1\}^n.$$

At depth t, the layer Y_t consists of all partial assignments of length t that have not yet violated any local constraint:

$$Y_t \subseteq \{0,1,*\}^n.$$

Suppose that the constraints are arranged so that every partial assignment of depth $t < n$ is locally consistent, but only a small number of complete assignments at depth n are globally accepting.

Then:

$$|Y_t| = 2^t$$

for all $t < n$.

Thus, local consistency checks do not reduce the live frontier. The frontier remains exponential until the final layers, where global constraints become visible.

This shows that the existence of local rules and local cutoff tests does not imply polynomial navigability. A polynomial realization requires a global compression mechanism capable of aggregating the live frontier before it expands exponentially.

In this toy model, the failure is not the absence of local structure. The failure is the absence of a global structure that turns local consistency into polynomially bounded navigation.

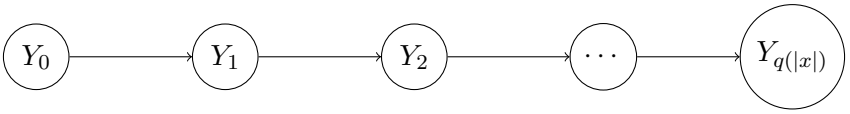

Figure 2: Layered dynamics of state spaces.

4.11 Connection to proof complexity

Structural realizations can be interpreted in proof-theoretic terms.

- The cutoff procedure corresponds to deriving a refutation of a subspace of assignments.
- The frontier corresponds to the number or complexity of simultaneously maintained subspaces.
- The completion procedure corresponds to constructing a satisfying assignment or a full refutation.

Thus, polynomial structural realizations correspond to proof systems in which all relevant subspaces admit polynomial-size refutations or certificates.

Known exponential lower bounds for proof systems such as resolution imply that such structural realizations are unlikely to exist for general 3-SAT instances.

4.12 Collapse and proof width

We now relate structural collapse to proof complexity more explicitly.

Definition 15 (Cutoff proof)**.** *Let s be a state representing a subspace of assignments. A cutoff proof for s is a derivation, in a fixed proof system $\mathcal{P}$, that no assignment in the subspace represented by s satisfies the formula.*

Definition 16 (Polynomial-width cutoff)**.** *A structural realization has polynomial-width cutoff with respect to $\mathcal{P}$ if every cutoff $C(x, s) = 1$ admits a proof in $\mathcal{P}$ of width at most $poly(|x|)$ and size at most $poly(|x|)$.*

Proposition 4 (Polynomial cutoff induces bounded proof complexity)**.** *If a structural realization of an unsatisfiable CNF formula uses a cutoff procedure whose positive answers are justified by proofs in a proof system $\mathcal{P}$ of polynomial size and polynomial width, then the overall rejection of the formula induces a polynomial-size $\mathcal{P}$-refutation.*

Proof. For an unsatisfiable input, every live trajectory must eventually be cut off. By polynomial divergence and polynomial frontier, only polynomially many non-cutoff states need be considered before all accepting continuations are eliminated. Each cutoff is justified by a polynomial-size proof in $\mathcal{P}$. Combining these polynomially many local refutations yields a polynomial-size global refutation in $\mathcal{P}$. □

Corollary 1. *If a family of unsatisfiable CNF formulas requires super-polynomial refutations in a proof system $\mathcal{P}$, then no structural realization whose cutoffs are all justified inside $\mathcal{P}$ with polynomial size and width can exist for that family.*

This does not rule out all possible structural realizations of 3-SAT. Rather, it identifies a precise obstruction: any proposed realization must either avoid the restricted proof system $\mathcal{P}$, or provide a stronger form of cutoff than the one available in $\mathcal{P}$.

5 Formalism Dependence of Complexity Classes

5.1 Computational Formalisms

We now analyze how the relation between P_F and NP_F depends on the choice of formalism $F = (M, R, D)$ introduced earlier.

We define the complexity classes relative to F:

- P_F consists of languages decidable using models in M with resource bounded by a polynomial in $|x|$ (measured in R),
- NP_F consists of languages verifiable with polynomial resource and certificates of polynomial size (measured in R and D).

5.2 Robust Class of Formalisms

We define a class F_{robust} of formalisms satisfying:

- any two models in M simulate each other with polynomial overhead in R,
- primitive operations do not compress exponential work into a single step,
- resources are measured as functions of input size $|x|$.

Simulatability condition. We make the robustness requirement explicit: every primitive operation in a formalism $F = (M, R, D)$ must admit a simulation in a standard model (e.g., Turing machine or RAM) with at most polynomial overhead in the resource scale R.

Formally, for every operation $\mathcal{O}$ in M, there exists a procedure $\mathcal{O}^*$ such that:

$$\text{cost}(\mathcal{O}^*) \leq poly(|x|).$$

Violations of robustness. Typical operations that violate this condition include:

- applying a transformation simultaneously to all elements of an exponentially large set,
- deciding existence of a satisfying assignment over a compressed representation in a single step,
- performing global filtering over all states of a layer without explicit polynomial simulation.

Such operations effectively encode super-polynomial computation into a single step and therefore belong to extended formalisms.

Theorem 3 (Invariance of P vs NP)**.** *For any $F_1, F_2 \in F_{\text{robust}}$,*

$$P_{F_1} = NP_{F_1} \iff P_{F_2} = NP_{F_2}.$$

This captures the standard robustness of classical computational models.

5.3 Extended Formalisms

We define a broader class $F_{\text{ext}} \supset F_{\text{robust}}$ where:

- global or aggregated operations are allowed,
- structured objects (e.g., layers, compressed state spaces) are admissible in D,
- alternative resource measures R' may be used.

Theorem 4 (Formalism Dependence)**.** *There exist formalisms $F \in F_{\text{robust}}$ and $F' \in F_{\text{ext}}$ such that*

$$P_F \neq NP_F, \qquad P_{F'} = NP_{F'}.$$

Thus, the statement $P = NP$ is inherently relative to the chosen formalism.

Example of an extended formalism. Consider a formalism F' in which the following operation is primitive:

- given a layer Y_t, decide whether it contains an accepting state in a single step.

In such a formalism, SAT can be solved in polynomially many steps by propagating layers and testing acceptance directly. Thus $P_{F'} = NP_{F'}$.

However, this operation does not admit a polynomial simulation in a robust formalism, and therefore F' lies outside F_{robust}.

5.4 Role of Global Structures S

Robust simulability. We make the admissibility of structured objects precise.

Definition 17 (Robustly simulable structured object)**.** *Let $F = (M, R, D)$ be a formalism in F_{robust}. A structured object S with operations $\mathcal{O} = \{\mathcal{O}_i\}$ is* robustly simulable *if:*

- *(**Size**) $|\text{enc}(S)| \leq poly(|x|)$,*
- *(**Simulation**) for every $\mathcal{O}_i \in \mathcal{O}$ there exists a procedure $\mathcal{O}_i^*$ in a standard model (e.g., TM/RAM) such that*
$$\text{cost}(\mathcal{O}_i^*) \leq poly(|x|),$$
- *(**Local realizability**) $\mathcal{O}_i^*$ is composed of primitive steps whose costs are bounded by R and do not hide super-polynomial work.*

Admissibility criterion. A structure S is admissible for classical P–analysis iff it is robustly simulable. Otherwise S belongs to an extended formalism $F' = (M', R', D')$.

Consider a structured representation S (e.g., a layer representing exponentially many states). Such a structure is meaningful only if its operations satisfy:

$$\text{operations on } S \text{ are simulable in } poly(|x|) \text{ within } F_{\text{robust}}.$$

Otherwise, S belongs to an extended formalism F' and does not imply results about classical P vs NP.

5.5 Description vs Computation

A key distinction emerges:

- Exponential objects may admit compact descriptions,
- but compact description does not imply efficient computation.

This explains why:

$$\text{compact representation} \not\Rightarrow \text{polynomial-time computation.}$$

5.6 Interpretation

We conclude:

- Global structure is not specified explicitly but emerges from local rules,
- Local structure does not guarantee a global polynomial navigation,
- Complexity is a property of the interaction between representation and allowed operations.

6 Examples

6.1 Horn-SAT as a structural realization

We now give a complete structural realization for Horn-SAT in terms of divergence, frontier, and completion.

Let φ be a Horn formula over variables $x_1, \ldots, x_n$.

State space. A state is a set $T \subseteq \{x_1, \ldots, x_n\}$ of variables assigned to true. The initial state is $T = \emptyset$.

Transition relation. For each Horn clause of the form

$$(a_1 \wedge \cdots \wedge a_k) \to b,$$

if $a_1, \ldots, a_k \in T$, then we add b to T.

Accepting states. A state is accepting if it does not derive a contradiction (i.e., no clause implies $\bot$).

Cutoff. The cutoff procedure $C(x, T)$ returns 1 if $\bot$ is derivable from T.

Divergence. Any false trajectory derives $\bot$ after at most n steps, since each step strictly increases $|T|$. Hence divergence is polynomial.

Frontier. The process is monotone: there is a unique maximal closure of T. Thus the number of reachable non-cutoff states is at most n. In fact, the effective frontier collapses to a single canonical state.

Completion. The closure procedure computes the least fixed point of the implications. If $\bot$ is not derived, this fixed point defines a satisfying assignment.

Conclusion. Horn-SAT admits a structural realization where:

- divergence is linear,
- the frontier collapses to one state,
- completion is polynomial.

This illustrates the strongest possible form of polynomial structure: exponential branching is completely eliminated by monotone closure.

6.2 2-SAT

For 2-SAT, the structural state is the implication graph. Each clause $(\ell_1 \vee \ell_2)$ yields implications $\neg\ell_1 \to \ell_2$ and $\neg\ell_2 \to \ell_1$. Satisfiability is characterized by the absence of a variable x such that x and $\neg x$ lie in the same strongly connected component [7], [8].

The cutoff procedure detects such a conflict. Completion is performed by assigning values according to the DAG of strongly connected components. The live frontier is not a tree of assignments but a compact global object: the SCC condensation graph.

In this case, exponential branching over assignments is replaced by global compression through implications.

6.3 Bipartite matching

For bipartite matching, the state is a current matching M together with the layered alternating graph built from free vertices. The relevant structural fact is Berge's theorem: a matching is maximum iff there is no augmenting path. The Hopcroft–Karp algorithm improves the matching by finding a maximal set of shortest vertex-disjoint augmenting paths in phases [9], [10].

The frontier is not the set of all alternating paths. It is one layered graph representing all shortest augmenting paths at once. Completion consists of augmenting along a maximal family of such paths, and termination is guaranteed in polynomial time.

This illustrates a different mechanism from Horn-SAT and 2-SAT: the alternatives are not eliminated individually; they are aggregated into a single polynomially sized structure.

6.4 3-SAT

For 3-SAT, a natural state resembles a CDCL solver state: a partial assignment, an implication graph, and a clause database including learned clauses [11], [12], [13]. Local conflicts are easy to detect, and clause learning can compress parts of the search space.

However, the strong cutoff required above is not merely the detection of a currently falsified clause. It must certify that no accepting continuation is reachable from a state. In proof-complexity terms, this amounts to producing sufficiently strong refutations for inconsistent subspaces. Exponential lower bounds are known for several proof systems, including resolution lower bounds for the pigeonhole principle [14], [15].

Thus 3-SAT is not hard because it lacks local structure. It has abundant local structure. The difficulty is that known local structures do not assemble into a globally polynomially navigable state space for all inputs.

7 Description versus computation

A recurring confusion is to identify compact description with efficient computation. An exponential object may have a compact recursive description, for example $f(0) = 1$ and $f(n+1) = 2f(n)$. But the complexity class of a function is not determined only by the length of its description. It is determined by the resource needed to compute or use the relevant value or property.

Similarly, a layer of 2^n possible states may be described compactly as a hypercube. This alone does not imply that one can decide the existence of an accepting state in that layer in polynomial time. The critical question is whether the operations on the representation of the layer—transition, filtering, cutoff, and completion—are themselves polynomially realizable.

This is the sense in which there is no free compression of computation: compression is either paid for by resource, encoded into the model, or justified by a polynomial simulation.

8 Formalism dependence and the role of global structures

The preceding definitions separate two possibilities for a global structure S.

First, S may be realizable inside a robust formalism. Then operations on S admit polynomial simulations using ordinary resources, and S yields a genuine algorithmic result. For 3-SAT, such a structure would imply $\mathrm{P} = \mathrm{NP}$.

Second, S may require new primitive operations or a different resource scale. Then it belongs to another formalism $F' = (M', R', D')$. It may still be mathematically meaningful and computationally useful, but it no longer answers the classical question. Instead, it defines a new complexity theory in which one should ask whether $\mathrm{P}_{F'} = \mathrm{NP}_{F'}$.

The distinction is not between true and false computation, but between different choices of admissible operations and resource measures.

9 Related Work

This work is related to several lines of research.

Implicit and symbolic representations. Symbolic model checking and BDD-based methods represent large state spaces compactly, but their efficiency depends on the existence of compact representations supporting efficient operations.

Succinct representations. Succinct graph representations and compressed data structures similarly separate description size from computational complexity.

Proof complexity. Lower bounds in proof complexity provide evidence that certain search spaces cannot be globally compressed into polynomial-size certificates.

Algebraic and circuit representations. Monotone circuits and algebraic models study when global structure enables efficient computation, closely related to the notion of structural collapse introduced here.

The present framework unifies these perspectives by expressing them in terms of layered dynamics and polynomial navigability.

10 Open Problems and Research Direction

The framework proposed in this note is intentionally preliminary. Its purpose is not to provide a completed structural complexity theory, but to isolate a family of concepts – layered dynamics, collapse, frontier, divergence, completion, and admissible compression – that may support a broader structural approach to computational complexity.

Several directions appear particularly important for further development.

- **Collapse mechanisms for SAT-like problems.**

Does 3-SAT admit a nontrivial polynomially collapsing layered structure? More generally, can one identify intermediate forms of collapse that explain the empirical effectiveness of modern SAT techniques without yielding full polynomial navigability?

- **Structural taxonomy of polynomial solvability.**

  Can polynomial-time problems be classified according to distinct mechanisms of structural collapse? The examples considered here suggest at least monotone, quotient, and aggregation collapse, but it is unclear whether these form a complete taxonomy or whether fundamentally different mechanisms exist.

- **Quantitative invariants of navigability.**

  The present framework is largely qualitative. A natural next step would be the introduction of quantitative invariants measuring the growth, compressibility, or navigability of layered state spaces. Such invariants could potentially play a role analogous to time, space, proof width, or circuit depth in classical complexity theory.

- **Hierarchies of structural realizations.**

  Is there a hierarchy of structural realizations analogous to standard complexity hierarchies? For example, one may ask whether bounded-frontier, bounded-collapse, or bounded-divergence systems define distinct classes of computational behavior.

- **Connection to proof complexity.**

  The relation between structural collapse and proof complexity remains largely unexplored. Can divergence or frontier growth be related quantitatively to proof width, proof size, or refutation depth? Conversely, can lower bounds in proof complexity be systematically interpreted as obstructions to polynomial navigability?

- **Intermediate representations between local and global structure.**

  Many successful algorithms appear to exploit partially global structures without achieving full polynomial collapse. Understanding such intermediate representations may clarify why certain hard problems admit strong heuristics despite lacking known polynomial algorithms.

- **Robust admissibility of compressed structures.**

  A central unresolved issue is the distinction between genuine polynomial compression and hidden computational power encoded into primitive operations. A more systematic theory of admissible structured objects and robust simulability may be required.

- **Toward a structural complexity theory.**

  The broader long-term question is whether computational complexity can be organized not only through asymptotic resource bounds, but also through the geometry and dynamics of state-space evolution. The notions introduced here may serve as preliminary components of such a theory, but substantial formal development remains necessary.

11 Conclusion

The perspective developed in this note is intentionally modest in scope. The goal has not been to resolve the classical P versus NP problem, nor to replace existing complexity theory, but rather to isolate and organize a structural viewpoint that repeatedly appears across known polynomial-time algorithms.

Many of the observations discussed here are, in isolation, already familiar: Horn-SAT relies on monotone closure, 2-SAT on implication structure, bipartite matching on layered aggregation,

and modern SAT solving on partial forms of global compression. Their role in the present text is therefore not primarily to introduce new algorithmic facts, but to expose a recurring structural pattern behind seemingly different forms of polynomial computation.

From this perspective, polynomial solvability appears less as the absence of search than as the existence of mechanisms that prevent uncontrolled expansion of the live computational frontier. The notions of divergence, frontier, completion, layered dynamics, and collapse are intended as preliminary components of a language for describing such mechanisms systematically.

The discussion of computational formalisms serves a similar clarifying role. Compact global structures are meaningful only when their induced operations remain polynomially realizable inside robust computational models. Otherwise, one has not compressed computation, but merely shifted complexity into stronger primitives or alternative resource assumptions.

The framework presented here should therefore be viewed primarily as exploratory and organizational. Its purpose is to formulate the problem in structural terms, identify possible obstructions to polynomial navigability, and outline directions for a more systematic theory of state-space complexity and structural collapse.

References

[1] S. A. Cook, "The complexity of theorem-proving procedures," *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pp. 151–158, 1971. DOI: 10.1145/800157.805047

[2] R. M. Karp, "Reducibility among combinatorial problems," *Complexity of Computer Computations*, pp. 85–103, 1972. DOI: 10.1007/978-1-4684-2001-2_9

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[4] M. Sipser, *Introduction to the Theory of Computation*, 3rd ed. Cengage Learning, 2013.

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

[6] W. F. Dowling and J. H. Gallier, "Linear-time algorithms for testing the satisfiability of propositional horn formulae," *The Journal of Logic Programming*, vol. 1, no. 3, pp. 267–284, 1984. DOI: 10.1016/0743-1066(84)90014-1

[7] B. Aspvall, M. F. Plass, and R. E. Tarjan, "A linear-time algorithm for testing the truth of certain quantified boolean formulas," *Information Processing Letters*, vol. 8, no. 3, pp. 121–123, 1979. DOI: 10.1016/0020-0190(79)90002-4

[8] R. E. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972. DOI: 10.1137/0201010

[9] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, 1973. DOI: 10.1137/0202019

[10] C. Berge, *Théorie des graphes et ses applications*. Dunod, 1958.

[11] M. Davis and H. Putnam, "A computing procedure for quantification theory," *Journal of the ACM*, vol. 7, no. 3, pp. 201–215, 1960. DOI: 10.1145/321033.321034

[12] M. Davis, G. Logemann, and D. Loveland, "A machine program for theorem-proving," *Communications of the ACM*, vol. 5, no. 7, pp. 394–397, 1962. DOI: 10.1145/368273.368557

[13] J. P. Marques-Silva and K. A. Sakallah, "Grasp: A search algorithm for propositional satisfiability," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 506–521, 1999. DOI: 10.1109/12.769433

[14] A. Haken, "The intractability of resolution," *Theoretical Computer Science*, vol. 39, pp. 297–308, 1985. DOI: 10.1016/0304-3975(85)90144-6

[15] P. Beame and T. Pitassi, "Propositional proof complexity: Past, present, and future," *Bulletin of the EATCS*, vol. 65, pp. 66–89, 1998.