Scheme-Theoretic Approach to Computational Complexity. I. The Separation of P and NP

Ali Çivril*

September 23, 2024

Abstract

We lay the foundations of a new theory for algorithms and computational complexity by parameterizing the instances of a computational problem as a moduli scheme. Considering the geometry of the scheme associated to 3-SAT, we separate P and NP. In particular, we show that no deterministic algorithm can solve 3-SAT in time less than 1.296839^n in the worst case.

1 Introduction

This paper introduces the rudiments of a new theory for algorithms and computational complexity via the Hilbert scheme. One of the most important consequences of the theory is the resolution of the conjecture $\mathsf{P} \neq \mathsf{NP}$.

An easily understood reason for the difficulty of the problem we consider is the superficial similarity between the problems in P and NP-complete problems. More concretely, one has not been able to find a metric somehow measuring the time complexity of a problem so that the difference between the values for 3-SAT and 2-SAT is large enough. Extracting this intrinsic property from a problem seems out of reach when it is treated by only combinatorial means.

From an elementary point of view, a computational problem is considered to be a *language* recognized by a *Turing machine*. Through a slightly refined lens, it is a *Boolean function* computed by a *circuit*. We recognize the existence of a much deeper perspective: A computational problem is a (moduli) *scheme* formed by its instances, and an algorithm is a *morphism* geometrically reducing it to a single point. This opens the possibility of understanding computational complexity using the language of category theory. In particular, we define a functor from the category of computational problems to the category of schemes parameterizing the instances of a computational problem, albeit currently restricted to k-SAT.

For concreteness, consider a satisfiable instance of 3-SAT represented by the formula ϕ with variables $x_1, \ldots, x_n$. We associate with this instance all the solutions that make ϕ satisfiable, which can be expressed as the zeros of a polynomial $\phi(x_1, \ldots, x_n)$ over $\mathbb{F}_2$. We then identify this information by considering the closed subscheme $\operatorname{Proj} \overline{\mathbb{F}}_2[x_0, x_1, \ldots, x_n]/(\phi(x_0, x_1, \ldots, x_n))$. The global scheme corresponding to the computational problem 3-SAT is the Hilbert scheme parameterizing these closed subschemes together with a set of others to ensure connectedness.

The next step is to unify the notion of a reduction and an algorithm in the new setting. Consider $\text{1-SAT} \in \mathsf{P}$. In order to separate P and NP, one needs to rule out a polynomial-time reduction f

*Istanbul Atlas University, Computer Engineering Department, Kagithane, Istanbul Turkey, e-mail: ali.civril@atlas.edu.tr, website: www.alicivril.com

satisfying $x \in \mathsf{3\text{-}SAT} \Leftrightarrow f(x) \in \mathsf{1\text{-}SAT}$. We extend this line of thinking by introducing the simplest object in the category of computational problems: the trivial problem defined via an instance with an empty set of variables, which may be represented by a single point. In our new language, solving a problem is nothing but reducing it to the trivial problem. One then needs to show that, in geometric terms we will later formalize, it is impossible to reduce the scheme of 3-SAT to a single point with polynomial number of unit operations.

2 Computational Problems and the Extended Amplifying Functor

2.1 Computational Problems

A computational problem $(\Pi, \overline{\Pi})$ consists of a set Π of *positive instances* and a set $\overline{\Pi}$ of *negative instances*, where each instance is represented by a finite string over a finite alphabet. Solving $(\Pi, \overline{\Pi})$ amounts to deciding, given an instance, whether it is a positive instance or a negative instance. By an *instance* is meant a positive instance in the rest of the paper, unless otherwise stated. We also briefly denote a given problem by its set of positive instances Π. With an abuse of notation, if Π consists of a single instance I, then I also denotes the computational problem Π.

We will specifically consider the computational problem k-SAT defined via the variable set $\{x_1, \ldots, x_n\}$. An instance of this problem is a satisfiable Boolean formula in CNF with each clause containing k literals. In this case we also consider the representation of an instance by a finite set of polynomial equations over $\mathbb{F}_2$, as will be exemplified below. We thus use a *polynomial system* as a synonym for an instance. The synonym for a single polynomial equation is a *clause*. A solution for an instance in this case is an assignment to the variables in $\mathbb{F}_2$ satisfying all the equations of the instance.

We give below examples of instances and negative instances of some computational problems, both in the classical logical form and in the algebraic form as polynomial systems over $\mathbb{F}_2$. The simplest problem is what we denote by TRIVIAL or T for short, defined via a single instance and a single negative instance, both with an empty set of variables. These instances are denoted by $True$ and $False$. Their algebraic form are $\{0 = 0\}$ and $\{1 = 0\}$, respectively.

Problem: TRIVIAL or T
Logical form: $True$, $False$.
Algebraic form: $\{0 = 0\}$, $\{1 = 0\}$.

Problem: 1-SAT
Logical form: $\{x\}$, $\{x_1 \wedge \overline{x_2}\}$, $\{x_1 \wedge \overline{x_1}\}$.
Algebraic form: $\{1 - x = 0\}$, $\{1 - x_1 = 0, x_2 = 0\}$, $\{1 - x_1 = 0, x_1 = 0\}$

Problem: 3-SAT
Logical form: $\{(x_1 \vee \overline{x_3} \vee x_4) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_5})\}$.
Algebraic form: $\{(1 - x_1)x_3(1 - x_4) = 0, x_2(1 - x_3)x_5 = 0\}$.

2.2 Representability of the Hilbert Functor

Let S be a scheme, and let $X \subseteq \mathbb{P}^n_S$ be a closed subscheme. Define

$$H(X/S) := \{Z \subseteq X \text{ is a closed subscheme}, Z \to S \text{ is flat}\}.$$

The Hilbert functor $\mathcal{H}_{X/S}$ is the functor $T \mapsto H(X \times_S T/T)$ for any S-scheme T. We set $S = \text{Spec } \overline{\mathbb{F}}_2$, and denote $\mathcal{H}_{X/\overline{\mathbb{F}}_2}$ briefly as $\mathcal{H}_X$.

Let X be a projective scheme over $\overline{\mathbb{F}}_2$, and let $Z \subseteq X$ be a closed subscheme. Let $\mathscr{F}$ be a coherent sheaf on Z. The *Hilbert polynomial of Z with respect to $\mathscr{F}$* is $P(Z, \mathscr{F})(m) := \chi(Z, \mathscr{F}(m))$, where $\mathscr{F}(m)$ is the twisting of $\mathscr{F}$ by m, and $\chi(Z, \mathscr{F})$ denotes the *Euler characteristic* of $\mathscr{F}$ given by

$$\chi(Z, \mathscr{F}) := \sum_{i=0}^{\dim Z} (-1)^i \dim_{\overline{\mathbb{F}}_2} H^i(Z, \mathscr{F}). \tag{1}$$

The *Hilbert polynomial of Z* is

$$P(Z)(m) := \chi(Z, \mathscr{O}_Z(m)) \tag{2}$$

where $\mathscr{O}_Z$ is the structure sheaf of Z. Let $\mathcal{H}_X^P$ denote the subfunctor of $\mathcal{H}_X$ induced by the closed subschemes of X with a fixed Hilbert polynomial $P \in \mathbb{Q}[x]$. By the following result stated in our context, the Hilbert functor is representable by a projective scheme over $\overline{\mathbb{F}}_2$.

Theorem 2.1 ([1])**.** *Let X be a projective scheme over $\overline{\mathbb{F}}_2$. Then for every polynomial $P \in \mathbb{Q}[x]$, there exists a projective scheme* $\text{Hilb}^P(X)$ *over $\overline{\mathbb{F}}_2$, which represents the functor $\mathcal{H}_X^P$. Furthermore, the Hilbert functor $\mathcal{H}_X$ is represented by the* Hilbert scheme

$$\text{Hilb}(X) := \coprod_{P \in \mathbb{Q}[x]} \text{Hilb}^P(X).$$

We set $\Pi := k\text{-SAT}$. Recall that an instance of this problem consists of a satisfiable Boolean formula in CNF with each clause containing k literals, and by our assumptions its corresponding polynomial system over $\mathbb{F}_2$. Given a homogenized polynomial ϕ, one might consider the closed subscheme

$$\text{Proj } \overline{\mathbb{F}}_2[x_0, x_1, \ldots, x_n]/(\phi(x_0, x_1, \ldots, x_n)),$$

so that each polynomial equation and hence a polynomial system of Π (upon homogenization) identifies a closed subscheme of $\mathbb{P}^n_{\overline{\mathbb{F}}_2}$ via the corresponding ideal. We thus set $X = \mathbb{P}^n_{\overline{\mathbb{F}}_2}$ in the theorem above, and refer to the Hilbert polynomial of an instance.

2.3 Reductions and Prime Homogeneous Simple Sub-problems

Let $(\mathsf{A}, \overline{\mathsf{A}})$ and $(\mathsf{B}, \overline{\mathsf{B}})$ be computational problems, and $f : (\mathsf{A}, \overline{\mathsf{A}}) \to (\mathsf{B}, \overline{\mathsf{B}})$ be a set-theoretic map such that $f(\mathsf{A}) \subseteq \mathsf{B}$ and $f(\overline{\mathsf{A}}) \subseteq \overline{\mathsf{B}}$.

Definition 2.2. A computational procedure $\alpha_f : (\mathsf{A}, \overline{\mathsf{A}}) \to (\mathsf{B}, \overline{\mathsf{B}})$ realizing f, possibly with an advice string (thus simulating circuits), is called a *reduction*. We assume that this procedure is executed by a Turing machine, which starts with an element of $(\mathsf{A}, \overline{\mathsf{A}})$ on its tape, and halts with an element of $(\mathsf{B}, \overline{\mathsf{B}})$. In this case we disregard the action of the Turing machine on negative instances and briefly denote α_f by $\alpha_f : \mathsf{A} \to \mathsf{B}$.

Definition 2.3. The number of *deterministic* computational steps performed by a reduction α_f is called the *complexity* of α_f, denoted by $\tau(\alpha_f)$.

Definition 2.4. $\tau(f) := \tau(\mathsf{A}, \mathsf{B}) := \min_{\alpha_f} \tau(\alpha_f)$ is called the complexity of f.

Definition 2.5. $\tau(\mathsf{A}) := \tau(\mathsf{A}, \mathsf{T})$ is called the complexity of *solving* A. In this case a reduction $\alpha_f : \mathsf{A} \to \mathsf{T}$ realizing the unique set-theoretic map $f : (\mathsf{A}, \overline{\mathsf{A}}) \to (\{True\}, \{False\})$ such that $f(\mathsf{A}) = True$ and $f(\overline{\mathsf{A}}) = False$ is said to *solve* A.

Definition 2.6. Given an instance I of Π, a reduction $\alpha : I \to \mathsf{T}$ is called a *unit reduction*.

Example 2.1. Suppose I is $\{x = 0\}$. A unit reduction may write 0 on x, outputting $\{0 = 0\}$, which is equivalent to $True$.

Definition 2.7. Given two instances I_1 and I_2 of Π, a reduction $\alpha : I_1 \to I_2$ is called a *unit instance reduction*.

Example 2.2. Suppose I_1 is $\{x_1 = 0, 1 - x_2 = 0\}$. A unit instance reduction may replace x_1 with $1 - x_1$ and x_2 with $1 - x_2$, resulting in another instance I_2, which is $\{1 - x_1 = 0, x_2 = 0\}$.

Definition 2.8. A reduction is called a *unit operation* if it is either a unit reduction or a unit instance reduction.

Definition 2.9. A computational problem defined via a non-empty subset of the instances of Π is called a *sub-problem* of Π.

Definition 2.10. A sub-problem Λ of Π is called a *simple sub-problem* if the instances of Λ have the same Hilbert polynomial.

Definition 2.11. Instances I_1 and I_2 of Π are said to be *distinct* if they satisfy the following:

1. They have distinct solution sets over $\overline{\mathbb{F}}_2$.
2. $I_1 \setminus I_2 \neq \emptyset$.
3. $I_2 \setminus I_1 \neq \emptyset$.

In this case we also say that I_1 is *distinct from* I_2.

Definition 2.12. A sub-problem Λ of Π whose instances are defined via the variable set $S = \{x_1, \ldots, x_n\}$, is said to be *homogeneous* if all the variables in S appear in each instance of Λ and the instances of Λ are pair-wise distinct.

Definition 2.13. Let I_1 and I_3 be instances of Π. Let I_2 and I_4 be instances of Π or T. Unit operations $\alpha : I_1 \to I_2$ and $\beta : I_3 \to I_4$ are said to be *distinct* if they satisfy the following:

1. $(I_1 \triangle I_2) \setminus (I_3 \triangle I_4) \neq \emptyset$.
2. $(I_3 \triangle I_4) \setminus (I_1 \triangle I_2) \neq \emptyset$.

In this case we also say that α is *distinct from* β.

Remark 2.14. By definition, if α and β are distinct unit operations, then α performs a computational step that does not exist in β, and β performs a computational step that does not exist in α.

Example 2.3. Consider Example 2.2 with $I_1 : \{x_1 = 0, 1 - x_2 = 0\}$. The unit instance reduction permuting the variables x_1 and x_2 is not distinct from the unit instance reduction in Example 2.2, as it results in the same instance I_2. In particular, $(I_1 \triangle I_2) \setminus (I_1 \triangle I_2) = \emptyset$.

Definition 2.15. Given a sub-problem Λ of Π, let T_1 be the set of all unit reductions defined via the instances of Λ, and let T_2 be the set of all unit instance reductions defined between pairs of distinct instances of Λ. Let $T = T_1 \cup T_2$. The sub-problem Λ is said to be *prime* if the elements of T are pair-wise distinct.

Example 2.4. Let Λ be defined via the instances

$$I_1 : \{x_1 = 0, x_2 = 0\},$$
$$I_2 : \{x_1 = 0, 1 - x_2 = 0\},$$
$$I_3 : \{1 - x_1 = 0, 1 - x_2 = 0\}.$$

Then Λ is not prime since the unit instance reduction from I_1 to I_3 contains the unit instance reductions from I_1 to I_2 and I_2 to I_3. In particular, $(I_1 \triangle I_2) \setminus (I_1 \triangle I_3) = \emptyset$.

2.4 The Extended Amplifying Functor

Let Λ be a prime homogeneous simple sub-problem of Π consisting of a set of polynomial systems $\{P_i\}_{i=1}^{\ell}$ defined via the variables $x_1, \ldots, x_n$. Let ϕ_{ij} be the homogenized j-th polynomial in the polynomial system P_i:

$$\phi_{ij} := \phi_{ij}(x_0, x_1, \ldots, x_n),$$

for $j = 1, \ldots, |P_i|$. Define

$$X_i := \text{Proj } \overline{\mathbb{F}}_2[x_0, x_1, \ldots, x_n]/(\phi_{i1}, \ldots, \phi_{i|P_i|}), \tag{3}$$

for $i = 1, \ldots, \ell$. Let $X_\Lambda := \bigcup_{i=1}^{\ell} X_i$. In words, X_Λ contains all the closed subschemes identified by the instances of Λ. Define the *amplifying functor* $\mathcal{A}_\Lambda$ on Λ as

$$T \mapsto \{Y \times_{\overline{\mathbb{F}}_2} T | Y \in X_\Lambda, Y \times_{\overline{\mathbb{F}}_2} T \to T \text{ is flat}\},$$

for any scheme T over $\overline{\mathbb{F}}_2$. It is clear by definition that $\mathcal{A}_\Lambda$ is a subfunctor of the Hilbert functor, as it does nothing but considers only a subset of the set of all closed subschemes in defining the Hilbert functor. Define $\text{Hilb}(\Lambda) := \text{Hilb}^{P(\Lambda)}(\mathbb{P}^n_{\overline{\mathbb{F}}_2})$, where $P(\Lambda)$ is the Hilbert polynomial associated to Λ. For a fixed Hilbert polynomial P, $\text{Hilb}^P(\mathbb{P}^n_{\overline{\mathbb{F}}_2})$ is connected by a result of Hartshorne [2]. Thus, $\text{Hilb}(\Lambda)$ is connected.

Our strategy is via an extension of the amplifying functor from the category of computational problems to the category of schemes, which we define implicitly via its representation. We call it the *extended amplifying functor*. The objects of the source category are certain sub-problems, and the morphisms are reductions between sub-problems. In particular, the extended amplifying functor maps a certain sub-problem Λ to a geometric object $B(\Lambda)$ whose connectivity is crucial.

Recall that in order prove a separation result, one needs to establish a lower bound for *any* computational procedure, and a computational procedure might produce *any* set of instances during its execution. Nevertheless, we are only interested in the complexity of a specific computational problem, which encodes all the necessary information for our purpose. This leads us to the following strategy: We consider the representation of the sub-problem of interest in full detail with the aid of the Hilbert scheme. Any other instance that might appear during computation however, is mapped to an object with an irrelevant structure. This is enough to establish the main result. A more general functor is needed for a full theory of course, which we do not attempt for the time being.

Objects of the source category: Let Λ be a prime homogeneous simple sub-problem of Π. Over all such sub-problems Λ of Π, let $\kappa(\Pi)$ denote the maximum value of $b(\Lambda)$, the number of instances of Λ. From this point on, fix a single $\Lambda = \{I_1, \ldots, I_r\}$ with $\kappa(\Pi) = b(\Lambda) = r$. Let $\Lambda' = \{I_1, \ldots, I_{r-1}\}$ for $r \geq 2$, and $\Lambda' = \mathsf{T}$ for $r = 1$. The objects we consider are Λ, Λ', I_r, Γ, $\Lambda \cup \Gamma$, $\Lambda' \cup \Gamma$, $\{I_r\} \cup \Gamma$, and T, where Γ is a computational problem consisting of *any* non-empty set of instances with $\Lambda \cap \Gamma = \emptyset$.

Morphisms of the source category: Let A and B be computational problems. If $\mathsf{B} \subseteq \mathsf{A}$, and $f : \mathsf{A} \to \mathsf{B}$ is a set-theoretic map, we always consider the extension $f' : \mathsf{A} \to \mathsf{A}$. The morphisms we consider are all the reductions realizing the following maps, which cover all computational procedures solving Λ, Λ', and I_r:

$$\begin{array}{ccc}
\Lambda \to \Lambda & \Lambda \to \Gamma & \Lambda \to \mathsf{T} \\
\Lambda' \to \Lambda' & \Lambda' \to \Gamma & \Lambda' \to \mathsf{T} \\
I_r \to I_r & I_r \to \Gamma & I_r \to \mathsf{T} \\
\Gamma \to \Gamma & \Gamma \to \Lambda \cup \Gamma & \Gamma \to \mathsf{T} \\
\Lambda \cup \Gamma \to \Lambda \cup \Gamma & \Lambda \to \Lambda \cup \Gamma & \Lambda \cup \Gamma \to \mathsf{T} \\
\Lambda' \cup \Gamma \to \Lambda' \cup \Gamma & \Lambda' \to \Lambda' \cup \Gamma & \Lambda' \cup \Gamma \to \mathsf{T} \\
\{I_r\} \cup \Gamma \to \{I_r\} \cup \Gamma & \{I_r\} \to \{I_r\} \cup \Gamma & \{I_r\} \cup \Gamma \to \mathsf{T}
\end{array}$$

Objects in the image of the extended amplifying functor: We define $B(\Lambda) = \mathrm{Hilb}(\Lambda)$. Let p_r be the point of $B(\Lambda)$ representing the instance $I_r \in \Lambda$. With an abuse of notation, we also denote the scheme induced by this closed point by p_r, and set $B(I_r) = p_r$. We define $B(\Lambda')$ to be the scheme induced by the set of points of $B(\Lambda)$ excluding p_r. For the other objects defined in the source category as above, $B(\Gamma)$, $B(\Lambda \cup \Gamma)$, $B(\Lambda' \cup \Gamma)$, and $B(\{I_r\} \cup \Gamma)$ are all defined to be $\mathrm{Hilb}(\Lambda) \times \mathrm{Hilb}(\Lambda)$, where the product is over $\overline{\mathbb{F}}_2$. We define $B(\mathsf{T}) = \mathrm{Spec}\ \overline{\mathbb{F}}_2$.

Morphisms in the image of the extended amplifying functor: Recall that given a set-theoretic map $f : \mathsf{A} \to \mathsf{B}$, we consider *all* the reductions realizing f. The functor thus maps all such reductions to a single algebro-geometric morphism between the schemes representing A and B. In particular, $B(\Lambda \to \Lambda)$ is defined to be the identity morphism $B(\Lambda) \to B(\Lambda)$, which maps each point of $B(\Lambda)$ to itself, $B(\Lambda' \to \Lambda')$ is the morphism $B(\Lambda') \to B(\Lambda')$ induced by $B(\Lambda')$, and $B(I_r \to I_r)$ is the morphism $p_r \to p_r$ induced by the closed point p_r. The other morphisms are defined as follows. Given a morphism $\mathsf{A} \to \mathsf{B}$ in the source category, if $\mathsf{B} = \mathsf{T}$, then $B(\mathsf{A} \to \mathsf{B})$ is the structure morphism $B(\mathsf{A}) \to \mathrm{Spec}\ \overline{\mathbb{F}}_2$. If both A and B contain Γ, then $B(\mathsf{A} \to \mathsf{B})$ is the identity morphism $\mathrm{Hilb}(\Lambda) \times \mathrm{Hilb}(\Lambda) \to \mathrm{Hilb}(\Lambda) \times \mathrm{Hilb}(\Lambda)$. If B contains Γ and $\mathsf{A} = \Lambda$, then $B(\mathsf{A} \to \mathsf{B})$ is the diagonal morphism $\mathrm{Hilb}(\Lambda) \to \mathrm{Hilb}(\Lambda) \times \mathrm{Hilb}(\Lambda)$. If B contains Γ and $\mathsf{A} = \Lambda'$ or $\mathsf{A} = I_r$, then $B(\mathsf{A} \to \mathsf{B})$ is the diagonal morphism induced by Λ' or p_r, respectively. It is clear that these morphisms define a functor.

3 The Fundamental Lemma of Lower Bounds

Lemma 3.1 (Fundamental Lemma)**.**

$$\tau(\Pi) \geq \kappa(\Pi).$$

Proof. Let Λ and Λ' be the sub-problems defined in the previous section. Since $\tau(\Pi) \geq \tau(\Lambda)$, it suffices to show $\tau(\Lambda) \geq r$. We argue by induction on r. For $r = 1$, we clearly have $\tau(\Lambda) \geq 1$, since the complexity of solving a problem other than T is non-zero. For $r \geq 2$, assume $\tau(\Lambda') \geq r - 1$. We want to relate the complexity of the map $\Lambda \to \mathsf{T}$ to the complexity of the map $\Lambda' \to \mathsf{T}$. To this aim, consider a factorization of the morphism $f : B(\Lambda) \to \mathrm{Spec}\ \overline{\mathbb{F}}_2$ in the image of the extended amplifying functor as

$$B(\Lambda) \xrightarrow{h} X \to \mathrm{Spec}\ \overline{\mathbb{F}}_2,$$

such that $B(\Lambda') \subseteq X$ and $h[B(\Lambda')] = B(\Lambda')$. In this case since $B(\Lambda)$ is connected, $h[B(\Lambda)]$ must be connected. This implies by the requirements of the factorization that $h[B(\Lambda)] = B(\Lambda)$. In words, while reducing $B(\Lambda)$ to Spec $\overline{\mathbb{F}}_2$ via a morphism, which fixes $B(\Lambda')$, one has to "go through" $B(\Lambda)$ itself. This restriction, provided by the connectedness of $B(\Lambda)$, is crucial for our argument:

$$B(\Lambda) \xrightarrow{h} B(\Lambda) \to \text{Spec } \overline{\mathbb{F}}_2.$$

By our assumption, we also have

$$B(\Lambda') \xrightarrow{h} B(\Lambda') \to \text{Spec } \overline{\mathbb{F}}_2,$$

where h is uniquely defined in the image of the extended amplifying functor.

A pre-image of the first factorization above might have the following two reduction sequences applied to I_r.

$$\begin{array}{rccccc} & \Lambda & \to & \Lambda & \to & \mathsf{T} \\ \alpha_1: & I_r & \mapsto & I_r & \overset{\alpha_3}{\longmapsto} & \mathsf{T} \\ \alpha_2: & I_r & \overset{\alpha_4}{\longmapsto} & I_j & \mapsto & \mathsf{T} \end{array}$$

where $j \in \{1, \ldots, r-1\}$. We might also have the following two reduction sequences in a pre-image of the second factorization.

$$\begin{array}{rccccc} & \Lambda' & \to & \Lambda' & \to & \mathsf{T} \\ \beta_1: & I_j & \mapsto & I_j & \overset{\beta_3}{\longmapsto} & \mathsf{T} \\ \beta_2: & I_j & \overset{\beta_4}{\longmapsto} & I_k & \mapsto & \mathsf{T} \end{array}$$

where $k \in \{1, \ldots, r-1\} \setminus \{j\}$. Note that α_3 and β_3 are unit reductions, whereas α_4 and β_4 are unit instance reductions.

Consider $\alpha_1 \cup \beta : \Lambda \to \mathsf{T}$, where β contains, for each j, either β_1 or β_2. Note that this reduction contains a unit reduction α_3. Since Λ is prime, α_3 is distinct from all β_3 and all β_4 in the diagram above. This implies by Remark 2.14 that a reduction containing α_1 performs a computational step that does not exist in β. We thus obtain

$$\tau(\alpha_1 \cup \beta) \geq \tau(\beta) + 1. \tag{4}$$

Consider next $\alpha_2 \cup \beta : \Lambda \to \mathsf{T}$, where β contains, for each j, either β_1 or β_2. This reduction contains a unit instance reduction α_4. Since Λ is prime, α_4 is distinct from all β_3 and all β_4 in the diagram above. This implies by Remark 2.14 that a reduction containing α_2 performs a computational step that does not exist in β. We then have

$$\tau(\alpha_2 \cup \beta) \geq \tau(\beta) + 1. \tag{5}$$

The inequalities (4) and (5) together imply $\tau(\Lambda) \geq \tau(\Lambda') + 1 \geq (r-1) + 1 = r$, which completes the induction and the proof. □

4 3-SAT: The Separation of P and NP

Denote by k-SAT(n, m) the problem k-SAT with n variables and m clauses.

Theorem 4.1. *For any constant $\epsilon > 0$, there exist infinitely many $n \in \mathbb{Z}^+$ such that*

$$\kappa(\text{3-SAT}(n, 2n)) \geq 2^{\left(\frac{3}{8} - \epsilon\right)n}.$$

Proof. We construct a prime homogeneous simple sub-problem of 3-SAT with $\binom{r}{r/2} \cdot 2^{r/2}$ instances, each having $4r$ variables and $8r$ clauses, for $r \geq 1$.

The Initial Construction: A Homogeneous Simple Sub-problem Each instance consists of r *blocks*. For $r = 1$, a block of an instance is initially defined via 4 variables x_1, x_2, x_3, x_4, and 8 clauses. We first construct 3 instances with the solution sets over $\mathbb{F}_2$ consisting of the following points:

Instance 1	Instance 2	Instance 3
$(0,0,1,0)$	$(0,0,1,0)$	$(0,1,0,0)$
$(1,0,0,0)$	$(0,1,0,0)$	$(0,1,1,0)$
$(1,1,0,0)$	$(1,0,1,0)$	$(1,0,0,0)$

These instances consisting of a single block are shown in Table 1. A block for each instance can be described by a procedure using the truth table of the variables. Each of the 8 clauses is introduced one by one to rule out certain assignments over $\mathbb{F}_2$ in the tables. We enumerate the rows of the tables for each instance by an indexing of these clauses in Table 2, Table 3, and Table 4. The solution sets over $\mathbb{F}_2$ are the entries left out by the introduced clauses. We will show below that the corresponding schemes over $\overline{\mathbb{F}}_2$ have isomorphic cohomology groups with respect to any coherent sheaf, so that by (1) and (2) the Hilbert polynomials of the instances are the same. In particular, we show that they are the disjoint union of a closed point and a linear subspace, which implies the claim.

The first 5 clauses of the instances are common. Clause 1 forces at least one of x_1, x_2, and x_3

Clause	Instance 1	Instance 2	Instance 3
1	$x_1 \vee x_2 \vee x_3$	$x_1 \vee x_2 \vee x_3$	$x_1 \vee x_2 \vee x_3$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$x_2 \vee \overline{x_3} \vee \overline{x_4}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_2} \vee x_3 \vee \overline{x_4}$
4	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$
5	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\overline{x_1} \vee x_2 \vee \overline{x_4}$
6	$x_1 \vee x_3 \vee x_4$	$\overline{x_1} \vee x_3 \vee x_4$	$\overline{x_1} \vee \overline{x_2} \vee x_4$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$x_2 \vee x_3 \vee x_4$	$\overline{x_1} \vee \overline{x_3} \vee x_4$
8	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$x_2 \vee \overline{x_3} \vee x_4$

Table 1: The clauses of the 3 instances satisfying Table 2, Table 3, and Table 4

Clause	x_1	x_2	x_3	x_4	Clause	x_1	x_2	x_3	x_4
1	0	0	0	0		1	0	0	0
1	0	0	0	1	5	1	0	0	1
	0	0	1	0	7	1	0	1	0
2	0	0	1	1	2	1	0	1	1
6	0	1	0	0		1	1	0	0
3	0	1	0	1	3	1	1	0	1
8	0	1	1	0	8	1	1	1	0
4	0	1	1	1	4	1	1	1	1

Table 2: The truth table of a block of Instance 1 with clause-indexing

Clause	x_1	x_2	x_3	x_4	Clause	x_1	x_2	x_3	x_4
1	0	0	0	0	7	1	0	0	0
1	0	0	0	1	5	1	0	0	1
	0	0	1	0		1	0	1	0
2	0	0	1	1	2	1	0	1	1
	0	1	0	0	6	1	1	0	0
3	0	1	0	1	3	1	1	0	1
8	0	1	1	0	8	1	1	1	0
4	0	1	1	1	4	1	1	1	1

Table 3: The truth table of a block of Instance 2 with clause-indexing

Clause	x_1	x_2	x_3	x_4	Clause	x_1	x_2	x_3	x_4
1	0	0	0	0		1	0	0	0
1	0	0	0	1	5	1	0	0	1
8	0	0	1	0	8	1	0	1	0
2	0	0	1	1	2	1	0	1	1
	0	1	0	0	6	1	1	0	0
3	0	1	0	1	3	1	1	0	1
	0	1	1	0	7	1	1	1	0
4	0	1	1	1	4	1	1	1	1

Table 4: The truth table of a block of Instance 3 with clause-indexing

to be 1, as it corresponds to

$$(1-x_1)(1-x_2)(1-x_3)=0.$$

Given this, the following 4 clauses make $x_4=0$, since $x_4\neq 0$ implies $x_1=x_2=x_3=0$ by these clauses. In other words, $x_i=1$ for any $i\in\{1,2,3\}$ implies a contradiction in the following system:

$$\begin{gathered}(1-x_2)x_3=0.\\ x_2(1-x_3)=0.\\ x_2x_3=0.\\ x_1(1-x_2)=0.\end{gathered}$$

Given that $x_4=0$ (or more generally $x_4\neq 1$), we now examine the last 3 clauses of the instances.

1. Instance 1:

$$\begin{gathered}(1-x_1)(1-x_3)=0.\\ x_1x_3=0.\\ x_2x_3=0.\end{gathered}$$

$x_1=1\Rightarrow x_3=0, x_2\in\overline{\mathbb{F}}_2.$

$x_2=1\Rightarrow x_1=1, x_3=0.$

$x_3=1\Rightarrow x_1=0, x_2=0.$

Thus, the solution set for (x_1,x_2,x_3) is $\{(0,0,1)\}\cup\{(1,\alpha,0)\}$, where $\alpha\in\overline{\mathbb{F}}_2$.

2. Instance 2:

$$x_1(1-x_3)=0.$$
$$(1-x_2)(1-x_3)=0.$$
$$x_2x_3=0.$$

$x_1=1\Rightarrow x_2=0, x_3=1.$

$x_2=1\Rightarrow x_1=0, x_3=0.$

$x_3=1\Rightarrow x_2=0, x_1\in\overline{\mathbb{F}}_2.$

Thus, the solution set for (x_1,x_2,x_3) is $\{(0,1,0)\}\cup\{(\alpha,0,1)\}$, where $\alpha\in\overline{\mathbb{F}}_2$.

3. Instance 3:

$$x_1x_2=0.$$
$$x_1x_3=0.$$
$$(1-x_2)x_3=0.$$

$x_1=1\Rightarrow x_2=0, x_3=0.$

$x_2=1\Rightarrow x_1=0, x_3\in\overline{\mathbb{F}}_2.$

$x_3=1\Rightarrow x_1=0, x_2=1.$

Thus, the solution set for (x_1,x_2,x_3) is $\{(1,0,0)\}\cup\{(0,1,\alpha)\}$, where $\alpha\in\overline{\mathbb{F}}_2$.

Note that all the 4 variables appear in all the instances. Since the instances are also distinct, they form a homogeneous simple sub-problem. Assume now the induction hypothesis that there exists a homogeneous simple sub-problem of size 3^r, for some $r\geq 1$. In the inductive step, we introduce 4 new variables $x_{4r+1}, x_{4r+2}, x_{4r+3}, x_{4r+4}$, and 3 new blocks on these variables each consisting of 8 clauses with the exact form as in Table 1. Appending these blocks to each of the 3^r instances of the induction hypothesis, we obtain 3^{r+1} instances. The constructed sub-problem is a homogeneous simple sub-problem. We now describe a procedure to make it into a prime homogeneous simple sub-problem.

Mixing the Blocks: A Prime Homogeneous Simple Sub-problem For simplicity and the purpose of providing examples, we describe the procedure for $r=2$. The construction is easily extended to the general case. Suppose that the first block is defined via Instance 1. We perform the following operation: Replace the literals of Clause 4 except $\overline{x_4}$ with appropriate literals of variables belonging to the second block, depending on which instance it is defined via. If the second block is defined via Instance 1, then Clause 4 becomes $(x_5\vee x_7\vee\overline{x_4})$. If it is defined via Instance 2, it becomes $(\overline{x_6}\vee\overline{x_7}\vee\overline{x_4})$. If it is defined via Instance 3, it becomes $(\overline{x_5}\vee\overline{x_6}\vee\overline{x_4})$. In extending this to the general case, the second block is generalized as the next block to the current one, and the variables used for replacement are the ones with the first three indices of the next block in increasing order, respectively corresponding to x_5, x_6, and x_7.

If the first block is defined via Instance 2, the same operations are performed, this time considering Clause 5. If the first block is defined via Instance 3, we consider Clause 2. All possible cases are illustrated in Table 5-Table 10, where the interchanged literals are shown in bold. In the general case, the described operation is also performed for the last block indexed r for which the next block is defined to be the first block, completing a cycle.

The constructed sub-problem is prime: In mixing the blocks, we force one specific clause of a block depending on its type to contain variables belonging to the next block in a way distinctive

Clause	Instance 1	Instance 1
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$x_6 \vee \overline{x_7} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\mathbf{x_5} \vee \mathbf{x_7} \vee \overline{x_4}$	$\mathbf{x_1} \vee \mathbf{x_3} \vee \overline{x_8}$
5	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\overline{x_5} \vee x_6 \vee \overline{x_8}$
6	$x_1 \vee x_3 \vee x_4$	$x_5 \vee x_7 \vee x_8$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$\overline{x_5} \vee \overline{x_7} \vee x_8$
8	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$\overline{x_6} \vee \overline{x_7} \vee x_8$

Table 5: Modification to form a prime sub-problem on Instance 1 and Instance 1 blocks

Clause	Instance 1	Instance 2
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$x_6 \vee \overline{x_7} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\overline{\mathbf{x_6}} \vee \overline{\mathbf{x_7}} \vee \overline{x_4}$	$\overline{x_6} \vee \overline{x_7} \vee \overline{x_8}$
5	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\mathbf{x_1} \vee \mathbf{x_3} \vee \overline{x_8}$
6	$x_1 \vee x_3 \vee x_4$	$\overline{x_5} \vee x_7 \vee x_8$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$x_6 \vee x_7 \vee x_8$
8	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$\overline{x_6} \vee \overline{x_7} \vee x_8$

Table 6: Modification to form a prime sub-problem on Instance 1 and Instance 2 blocks

Clause	Instance 1	Instance 3
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$\mathbf{x_1} \vee \mathbf{x_3} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\overline{\mathbf{x_5}} \vee \overline{\mathbf{x_6}} \vee \overline{x_4}$	$\overline{x_6} \vee \overline{x_7} \vee \overline{x_8}$
5	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\overline{x_5} \vee x_6 \vee \overline{x_8}$
6	$x_1 \vee x_3 \vee x_4$	$x_5 \vee x_7 \vee x_8$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$\overline{x_5} \vee \overline{x_7} \vee x_8$
8	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$\overline{x_6} \vee \overline{x_7} \vee x_8$

Table 7: Modification to form a prime sub-problem on Instance 1 and Instance 3 blocks

to the type of the next block. In particular, suppose we represent an instance as a sequence of blocks numbered according to their types. Then any unit instance reduction from the instance 22 to the instance 23 is distinct from a unit instance reduction from the instance 32 to the instance 33. The first operation can in fact be labeled as one from $(2,2)(2,2)$ to $(2,3)(3,2)$, since a block is distinguished by itself together with the next block. The second operation is from $(3,2)(2,3)$ to $(3,3)(3,3)$, which better indicates that it is distinct from the first one. The same applies to the general case, where there are arbitrarily many blocks. It is also clear that the unit reductions via

Clause	Instance 2	Instance 2
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$x_6 \vee \overline{x_7} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$	$\overline{x_6} \vee \overline{x_7} \vee \overline{x_8}$
5	$\overline{\mathbf{x_6}} \vee \overline{\mathbf{x_7}} \vee \overline{x_4}$	$\overline{\mathbf{x_2}} \vee \overline{\mathbf{x_3}} \vee \overline{x_8}$
6	$\overline{x_1} \vee x_3 \vee x_4$	$\overline{x_5} \vee \overline{x_6} \vee x_8$
7	$x_2 \vee x_3 \vee x_4$	$\overline{x_5} \vee \overline{x_7} \vee x_8$
8	$\overline{x_2} \vee \overline{x_3} \vee x_4$	$x_6 \vee \overline{x_7} \vee x_8$

Table 8: Modification to form a prime sub-problem on Instance 2 and Instance 2 blocks

Clause	Instance 2	Instance 3
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$x_2 \vee \overline{x_3} \vee \overline{x_4}$	$\overline{\mathbf{x_2}} \vee \overline{\mathbf{x_3}} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$	$\overline{x_6} \vee \overline{x_7} \vee \overline{x_8}$
5	$\overline{\mathbf{x_5}} \vee \overline{\mathbf{x_6}} \vee \overline{x_4}$	$\overline{x_5} \vee x_6 \vee \overline{x_8}$
6	$\overline{x_1} \vee \overline{x_2} \vee x_4$	$x_5 \vee x_7 \vee x_8$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$\overline{x_5} \vee \overline{x_7} \vee x_8$
8	$x_2 \vee \overline{x_3} \vee x_4$	$\overline{x_6} \vee \overline{x_7} \vee x_8$

Table 9: Modification to form a prime sub-problem on Instance 2 and Instance 3 blocks

Clause	Instance 3	Instance 3
1	$x_1 \vee x_2 \vee x_3$	$x_5 \vee x_6 \vee x_7$
2	$\overline{\mathbf{x_6}} \vee \overline{\mathbf{x_7}} \vee \overline{x_4}$	$\overline{\mathbf{x_2}} \vee \overline{\mathbf{x_3}} \vee \overline{x_8}$
3	$\overline{x_2} \vee x_3 \vee \overline{x_4}$	$\overline{x_6} \vee x_7 \vee \overline{x_8}$
4	$\overline{x_2} \vee \overline{x_3} \vee \overline{x_4}$	$\overline{x_6} \vee \overline{x_7} \vee \overline{x_8}$
5	$\overline{x_1} \vee x_2 \vee \overline{x_4}$	$\overline{x_5} \vee x_6 \vee \overline{x_8}$
6	$\overline{x_1} \vee \overline{x_2} \vee x_4$	$x_5 \vee x_7 \vee x_8$
7	$\overline{x_1} \vee \overline{x_3} \vee x_4$	$\overline{x_5} \vee \overline{x_7} \vee x_8$
8	$x_2 \vee \overline{x_3} \vee x_4$	$\overline{x_6} \vee \overline{x_7} \vee x_8$

Table 10: Modification to form a prime sub-problem on Instance 3 and Instance 3 blocks

the instances are all distinct from the aforementioned unit instance reductions, ensuring that we have a prime sub-problem.

Selecting a simple sub-problem: We first observe the following for the first block, which also holds for all the other blocks by the construction. Assume $x_4 \neq 0$ and $x_4 \neq 1$. We will show that this leads to a contradiction, so that $x_4 \neq 0$ implies $x_4 = 1$. Recall that Equation 1 forces at least one of x_1, x_2, and x_3 to be 1, which we repeatedly use below. Consider the case in which the

first block is defined via Instance 1. By the equations numbered 2, 3, and 5 of the first block:

$$(1-x_2)x_3 = 0.$$
$$x_2(1-x_3) = 0.$$
$$x_1(1-x_2) = 0.$$

$x_1 = 1 \Rightarrow x_2 = 1, x_3 = 1.$
$x_2 = 1 \Rightarrow x_3 = 1, x_1 \in \overline{\mathbb{F}}_2.$
$x_3 = 1 \Rightarrow x_2 = 1, x_1 \in \overline{\mathbb{F}}_2.$
Thus, the solution set to these equations is $\{(\alpha, 1, 1)\}$, which contradicts the solution set implied by the last 3 equations of the first block for $x_4 \neq 1$: $\{(0,0,1)\} \cup \{(1,\alpha,0)\}$.

Suppose now that the first block is defined via Instance 2. By looking at the equations numbered 2, 3, and 4 of the first block:

$$(1-x_2)x_3 = 0.$$
$$x_2(1-x_3) = 0.$$
$$x_2x_3 = 0.$$

$x_2 = 1 \Rightarrow x_3 = 0, x_3 = 1$, contradiction.
$x_3 = 1 \Rightarrow x_2 = 0, x_2 = 1$, contradiction.
$x_2 \neq 0 \Rightarrow x_3 = 0, x_3 = 1$, contradiction.
$x_3 \neq 0 \Rightarrow x_2 = 0, x_2 = 1$, contradiction.
Thus, the solution set to these equations is $\{(1,0,0)\}$, which contradicts the solution set implied by the last 3 equations of Instance 2 for $x_4 \neq 1$: $\{(0,1,0)\} \cup \{(\alpha,0,1)\}$.

Finally, suppose that the first block is defined via Instance 3. By looking at the equations numbered 3, 4, and 5 of the first block, we obtain

$$x_2(1-x_3) = 0.$$
$$x_2x_3 = 0.$$
$$x_1(1-x_2) = 0.$$

$x_1 = 1 \Rightarrow x_2 = 1, x_3 = 0, x_2 = 0$, contradiction.
$x_2 = 1 \Rightarrow x_3 = 1, x_2 = 0$, contradiction.
$x_3 = 1 \Rightarrow x_2 = 0, x_1 = 0.$
Thus, the solution set to these equations is $\{(0,0,1)\}$, which contradicts the solution set implied by the last 3 equations of Instance 3 for $x_4 \neq 1$: $\{(1,0,0)\} \cup \{(0,1,\alpha)\}$. All these imply that either $x_4 = 0$ or $x_4 = 1$.

We next observe that the replaced clauses in each block are satisfiable. Assume $x_4 \neq 0$. If the second block is defined via Instance 1, $x_5 \vee x_7$ does not contradict any of the terms of the solution set of Instance 1 for (x_5, x_6, x_7), which is $\{(0,0,1)\} \cup \{(1,\alpha,0)\} \cup \{(\alpha,1,1)\}$. Similarly, if the second block is defined via Instance 2, $\overline{x_6} \vee \overline{x_7}$ does not contradict any of the terms of the solution set of Instance 2, which is $\{(1,0,0)\} \cup \{(0,1,0)\} \cup \{(\alpha,0,1)\}$. If the second block is defined via Instance 3, $\overline{x_5} \vee \overline{x_6}$ does not contradict any of the terms of the solution set of Instance 3, which is $\{(0,0,1)\} \cup \{(1,0,0)\} \cup \{(0,1,\alpha)\}$.

We have already shown that for $x_4 = 0$, the solution sets associated to three different types of blocks have the same cohomology. Notice that for $x_4 = 1$, the solutions associated to these blocks are the ones computed in the discussion above. For Instance 1, it is $(\alpha, 1, 1, 1)$. For Instance 2, it is $(1, 0, 0, 1)$. For Instance 3, it is $(0, 0, 1, 1)$. Thus, the Hilbert polynomials associated to Instance 2 and Instance 3 are the same, whereas Instance 1 differs from them. We consider the following set of instances with uniform Hilbert polynomial. Select out of all instances having $r/2$ blocks defined via Instance 1 and $r/2$ blocks defined via either Instance 2 or Instance 3, where we assume r is

even. The number of such instances is $\binom{r}{r/2} \cdot 2^{r/2}$. Using the Stirling approximation, we have for all $\epsilon > 0$

$$\binom{r}{r/2} \cdot 2^{r/2} > 2^{\left(\frac{3}{2}-\epsilon\right)r},$$

as r tends to infinity. Since $r = n/4$, the proof is completed. □

By Theorem 4.1, Lemma 3.1, and the NP-completeness of 3-SAT [5]:

Corollary 4.2. $\mathsf{P} \neq \mathsf{NP}$.

The definition of τ also implies

Corollary 4.3. $\mathsf{NP} \not\subseteq \mathsf{P/poly}$.

Furthermore, by the specific lower bound derived for 3-SAT:

Corollary 4.4. *The exponential time hypothesis* [3] *is true against deterministic algorithms.*

Finally, this exponential lower bound implies the following by [4].

Corollary 4.5. $\mathsf{BPP} = \mathsf{P}$.

5 Final Remarks: Why 2-SAT is Easy

We first note that the base of the exponential function in Theorem 4.1 is $2^{3/8} \approx 1.296839$. In contrast, the best deterministic algorithm for 3-SAT runs in time $O(1.32793^n)$ [6]. We next show that the strategy developed in the previous section cannot establish a strong lower bound for 2-SAT. This partially explains, from a technical standpoint, why 3-SAT is hard but 2-SAT is easy. In brief, the strategy was as follows:

1. Define 3 instances on 4 variables, each via a single block, and forming a homogeneous simple sub-problem.
2. Introduce n blocks, each with a new set of variables, to attain an exponential number of instances forming a homogeneous simple sub-problem.
3. Mix the consecutive blocks in a distinctive way depending on their types, so that we have a prime homogeneous sub-problem. Select a further sub-problem, which is simple.

Let us now try to imitate our strategy for 2-SAT. Consider the two instances given in Table 11. The first 3 clauses imply that at least one of x_1 and x_2 is 1, and x_3 is 0. These are analogous to the first 5 clauses of the blocks we have defined for 3-SAT. Suppose we want to fix $x_1 = 0$ in the first instance so that the last clause is $\overline{x_1} \vee x_3$. The solution set of this instance over $\overline{\mathbb{F}}_2$ consists of the

Clause	Instance 1	Instance 2
1	$x_1 \vee x_2$	$x_1 \vee x_2$
2	$\overline{x_1} \vee \overline{x_3}$	$\overline{x_1} \vee \overline{x_3}$
3	$\overline{x_2} \vee \overline{x_3}$	$\overline{x_2} \vee \overline{x_3}$
4	$\overline{x_1} \vee x_3$	$\overline{x_2} \vee x_3$

Table 11: Two instances of 2-SAT forming a homogenous sub-problem

single closed point $(0, 1, 0)$, with the Hilbert polynomial 1. For the second instance, we analogously use $\overline{x_2} \vee x_3$ as the last clause, which results in the solution set $\{(1, 0, 0)\}$. Observe that we can now relate consecutive blocks via the second and the third clauses to create a prime sub-problem. In particular, there are two cases for Instance 1 in a prime sub-problem, the first including the clauses numbered 1, 2, and 4, the second including clauses numbered 1, 3, and 4. The first one corresponds to the following system:

$$(1 - x_1)(1 - x_2) = 0.$$
$$x_1 x_3 = 0.$$
$$x_1(1 - x_3) = 0.$$

The solution set of this system is $\{(0, 1, \alpha)\}$, where $\alpha \in \overline{\mathbb{F}}_2$. The second one corresponds to the following system:

$$(1 - x_1)(1 - x_2) = 0.$$
$$x_2 x_3 = 0.$$
$$x_1(1 - x_3) = 0.$$

The solution set of this system is $\{(1, 0, 1), (0, 1, 0)\}$. Recall the step of the analysis in the previous section, which ensures that the replaced clauses are satisfiable. In particular to 2-SAT, we should further check that, for $x_3 \neq 0$, there is an assignment to the replaced literal (which is one of $x_4, \overline{x_4}, x_5$, or $\overline{x_5}$) that satisfies the clause. In the first case we have the solution $(0, 1)$ for (x_4, x_5), and in the second case we have the solution $(1, 0)$. By the nature of the construction employed, one of these cases is used for Instance 1, and the other is used for Instance 2, considered for the current block. Recall also that if the next block is defined via Instance 1, the solution set for (x_4, x_5) might be $\{(0, 1)\}$, and if the next block is defined via Instance 2, the solution set might be $\{(1, 0)\}$. Thus, the replaced literal must satisfy both elements of the set $\{(0, 1), (1, 0)\}$, which is clearly not possible: In an attempt to construct a prime sub-problem with this approach, no matter which case we consider and no matter which literal we use for replacement, the satisfiability of the replaced clause is not guaranteed.

A clause of 2-SAT puts a more stringent requirement on the variables than 3-SAT, resulting in only one clause that is not common between the instances. Furthermore, there is not enough "room" in a clause of 2-SAT that would let us consider different variations so as to ensure a prime sub-problem. In contrast, the freedom of having 4 variables and 3 non-common clauses between instances in the case of 3-SAT allows us to consider many more combinations, and we were able to show that one of them leads to a sub-problem that is both homogeneous, prime, and simple.

Acknowledgment We would like to thank Sinan Ünver for informing us about schemes and morphisms.

References

[1] A. Grothendieck. *Fondements de la Géométrie Algébrique [Extraits du Séminaire Bourbaki 1957-1962]*, chapter Techniques de construction et théorèmes d'existence en géométrie algébrique. IV. Les schémas de Hilbert. Secr. Math., 1962.

[2] R. Hartshorne. Connectedness of the Hilbert scheme. *Publications Mathématiques de l'IHÉS*, 29:5–48, 1966.

[3] R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[4] R. Impagliazzo and A. Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 220–229. ACM, 1997.

[5] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[6] S. Liu. Chain, generalization of covering code, and deterministic algorithm for k-SAT. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107, pages 88:1–88:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.