



CATEGORICAL COMPLEXITY

SAUGATA BASU¹ and UMUT ISIK²

¹ Department of Mathematics, Purdue University, West Lafayette, IN 47906, USA;
email: sbasu@math.purdue.edu

² Department of Mathematics, University of California, Irvine, Irvine, CA 92697, USA;
email: isik@math.uci.edu

Received 14 January 2018; accepted 21 March 2020

Abstract

We introduce a notion of complexity of diagrams (and, in particular, of objects and morphisms) in an arbitrary category, as well as a notion of complexity of functors between categories equipped with complexity functions. We discuss several examples of this new definition in categories of wide common interest such as finite sets, Boolean functions, topological spaces, vector spaces, semilinear and semialgebraic sets, graded algebras, affine and projective varieties and schemes, and modules over polynomial rings. We show that on one hand categorical complexity recovers in several settings classical notions of nonuniform computational complexity (such as circuit complexity), while on the other hand it has features that make it mathematically more natural. We also postulate that studying functor complexity is the categorical analog of classical questions in complexity theory about separating different complexity classes.

2010 Mathematics Subject Classification: 18A10, 68Q15 (primary); 14Q20 (secondary)

1. Introduction

It is usual to associate some measure of complexity to mathematical objects. For example, the complexity of a polynomial is often measured by its degree, or alternatively by the volume of its Newton polytope, or the number the monomials appearing in it with nonzero coefficients, or the least number of operations needed for an algorithm to evaluate the polynomial at a given point. Once a notion of complexity is fixed, one can make quantitative statements about properties of the objects in terms of their complexity. In the case of polynomials for example, there are many results on upper bounds on the topological invariants of the variety that

© The Author(s) 2020. This is an Open Access article, distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives licence (<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is unaltered and is properly cited. The written permission of Cambridge University Press must be obtained for commercial re-use or in order to create a derivative work.

the polynomial defines (see for example, the survey [BPR05]), the number of steps needed to desingularize the variety [BGMW11], and many other functions defined on the space of polynomials, in terms of the chosen complexity measure.

The notion of complexity also arose in theoretical computer science as a means of studying efficiency of algorithms and also to measure the intrinsic hardness of certain algorithmic problems. The latter led to the development of structural complexity theory and in particular to the famous P versus NP questions for discrete complexity classes, which remain unresolved until today. Even though these arose first in the context of decision problems and Boolean functions, there have been subsequent attempts to generalize the scope of computational complexity to other classes of mathematical objects—for example, the Blum–Shub–Smale (B–S–S) theory for computations over reals as well as complex numbers [BCSS98], over more general structures [Poi95], for polynomials [Val79a, Val79b, vzG87], and for constructible sheaves and functions [Bas15]. Some of these generalizations are motivated by costs of computations in certain models of computations, while others by the desire to have a sound internal notion of complexity for mathematical objects. Remarkably, there exist analogs of the P versus NP question in all the generalizations mentioned above. Thus, it seems that there should be a more fundamental way of looking at questions arising in computational complexity theory, which unifies these various viewpoints.

The goal of the current paper is to develop this general theory of complexity via a categorical approach that reconciles the intuitive notion of complexity of mathematical objects with the different notions of computational and circuit complexities used in theoretical computer science.

We start by defining a categorical notion called a *diagram computation*. In an arbitrary category \mathcal{C} with a chosen set of morphisms called *basic morphisms*, diagram computations can be used to construct diagrams and, in particular, objects and morphisms in \mathcal{C} as follows. At the first level, one starts with a diagram consisting entirely of basic morphisms and then successively adds limits and colimits of arbitrary subdiagrams, along with the accompanying morphisms from/to those subdiagrams, to construct more and more complex diagrams. Any diagram isomorphic to a subdiagram of the final resulting diagram is said to be computed by the diagram computation. This allows us to associate, to each object, arrow, or diagram in \mathcal{C} , a complexity by counting the number of limits, colimits, and basic morphisms used in its most efficient computation. Diagram computations come in three kinds, the full version described above, called a *mixed computation*, and two more restricted ones where one is either allowed to use only the so-called constructive limits or only constructive colimits. This leads to the notions of *mixed complexity*, *limit complexity*, and *colimit complexity* of objects, arrows, or any diagrams in \mathcal{C} .

Although our notion of complexity bears some similarity with a more classical view of complexity coming from logic, namely descriptive complexity [Imm95], there is one important respect in which our notion of complexity differs significantly from all classical notions. In our categorical world, isomorphic objects should have identical complexity—which is indeed the case with our definition. Thus, we are able to define a good notion of complexity in the category, say, of affine or projective schemes, which is independent of embeddings. This is very natural from the mathematical point of view—making our theory completely geometric in those settings—but is sometimes at odds with ordinary complexity theory, which deals with embedded objects (like subsets of the Boolean hypercube or subvarieties of \mathbb{C}^n). Nevertheless, we will show that, with the appropriate choice of category and basic morphisms, even noncategorical notions of complexity can be meaningfully embedded in categorical ones.

A fundamentally new point of view emerges when one thinks about the categorical analogs of classical complexity questions. For every functor between two categories for which complexity is defined, one can define a natural notion of complexity of the functor. Unlike, the complexity of diagrams, which are numbers, the complexity of a functor is a function $f : \mathbb{N} \rightarrow \mathbb{N}$, and one can ask whether the complexity of such a functor is bounded from above by a polynomial. In this way, classical questions about separation of complexity classes become, in the categorical world, questions about polynomial boundedness of the complexity of certain natural functors. With this shift of viewpoint, one can pose many questions about complexities of functors, which have no direct analogs in the world of computational complexity. Well-studied properties of functors such as preservation of limits and colimits, adjointness, and so on are important from this point of view.

The importance of functor complexity was already suggested in [Bas15], where the complexities of adjoint pairs of functors between the categories of semialgebraically constructible sheaves on finite-dimensional real affine spaces were posited as generalizing the P versus NP question in the real B–S–S model.

We remark here that connections between computability and logic on one hand and category theory and topos theory on the other hand have a long history (see for example, [LS88, MLM94]). A more recent work on computability and complexity in categorical structures using the notion of Kan extensions appears in [Yan15]. However, our goal is different, and it is to develop a completely general notion of complexity, based on category theory, that is useful in studying basic objects in algebra and geometry from a quantitative point of view. To the best of our knowledge, this task has not been undertaken before.

We now give a brief summary of our results. After the basic definitions in Section 3, we look at several key examples. For sets, the colimit complexity of

a set S is $\text{card}(S) + 1$ (Proposition 3.15). Infinite sets are ‘noncomputable’ in this theory. For topological spaces, colimit computations starting from simplices and face maps define a simplicial complexity for topological spaces. Similarly, mixed computations starting from points and intervals give rise to cubical complexity. These measure how hard it is to make a space from simplices and cubes, respectively. Another important example is one where we recover monotone Boolean complexity from the categorical complexity in the lattice of subsets of a finite set.

In order to relate the new notion of categorical complexity, with pre-existing notions of (nonuniform) complexity, such as circuit complexity, or lengths of straight-line programs (we refer the reader to the books [BCS97, Bür00] for these notions), we prove certain comparison theorems. The first set of such theorems is about affine varieties, affine schemes, and algebras over a field. We show that the affine zero-set of a polynomial with low arithmetic circuit complexity has low limit complexity (Theorem 4.2); on the other hand, if X is a variety with low limit complexity, then it is *isomorphic* to the zero-set of a polynomial with low arithmetic circuit complexity (Theorem 4.5). The same results hold for affine schemes and algebras. For projective schemes in \mathbb{P}^n : by building affine pieces with limits and then gluing them using colimits, we show that the mixed complexity of a projective scheme is bounded above by a constant multiple of n^2N , where N is the arithmetic circuit complexity of its defining equations.

The categorical complexities of isomorphic varieties are equal by definition, while circuit complexity, being a nongeometric attribute, does not share this property. In order to reconcile these two notions, we consider in Section 5 two additional categories where circuit complexity of polynomials is, in a sense, embedded into categorical complexity. The first of these is the category of pairs of graded algebras, constructed specifically to make this embedding possible. Still, it remains to be seen how complexity in this category compares to arithmetic circuit complexity of polynomials, or to the complexity of projective varieties discussed in [Isi19]. The second category considered here is the category of modules over polynomial rings, where we prove that the colimit complexities of a sequence of morphism diagram $(k[x_1, \dots, x_n] \xrightarrow{1 \mapsto f_n} k[x_1, \dots, x_n])_{n>0}$ are bounded by a quasipolynomial function of n , if and only if the arithmetic circuit complexities of the sequence $(f_n)_{n>0}$ are also bounded by a quasipolynomial function of n (cf. Remark 5.8).

In Section 6, we discuss the behavior of categorical complexity under the action of functors. Limit and colimit computations are preserved under right and left adjoints, respectively. We define the complexity of a functor $F : \mathcal{C} \rightarrow \mathcal{D}$ as a function $C(F)(n)$ of n , where $C(F)(n)$ is the supremum of the complexity of $F(D)$, where D runs over all diagrams in \mathcal{C} whose complexity is less than or

equal to n . We posit that the question of whether the complexity of the ‘image functor’ (see Definition 6.8) on the morphism category $\mathcal{C}^{\bullet \rightarrow \bullet}$ is polynomially bounded, is the categorical analog of the P versus NP problem for the category \mathcal{C} (cf. discussion in the beginning of Section 6.4). We investigate this question in the context of limit complexity in the categories of semilinear and semialgebraic sets and answer it in the negative. Our final result is an analysis of the colimit complexity of the image functor for the category of modules over polynomial rings.

Finally, in Section 7, we list several open problems and future research direction in the area of categorical complexity.

We assume no prior knowledge of category theory in this paper and have included all relevant definitions. For background in category theory, we refer the reader to the books [Avo10, ML98, Sim11]. For background in complexity theory, we refer the reader to the books [BCS97, Bür00]. Finally, we make use of certain basic functors from algebraic geometry, and we refer the reader to the book [MO15] as an accessible source for these.

2. Categories and functors

In this section, we recall some basic definitions from category theory and introduce some notation that will be useful in what follows.

DEFINITION 2.1 (Categories). A category \mathcal{C} consists of

- (1) a class $\text{Ob}(\mathcal{C})$ whose elements are the ‘objects of the category \mathcal{C} ’;
- (2) for every pair A, B of objects of \mathcal{C} , a set $\mathcal{C}(A, B)$ of ‘morphisms’ or ‘arrows’ from A to B ;
- (3) for every triple A, B, C of objects of \mathcal{C} , a composition law

$$\mathcal{C}(A, B) \times \mathcal{C}(B, C) \rightarrow \mathcal{C}(A, C),$$

which will be denoted by $(f, g) \mapsto g \circ f$;

- (4) for every object A of \mathcal{C} , a morphism $1_A \in \mathcal{C}(A, A)$ called the identity morphism on A .

The above data are subject to the following two axioms.

- (a) (Associativity). Given morphisms $f \in \mathcal{C}(A, B)$, $g \in \mathcal{C}(B, C)$, $h \in \mathcal{C}(C, D)$,

$$h \circ (g \circ f) = (h \circ g) \circ f.$$

- (b) (Identity). Given morphisms $f \in \mathcal{C}(A, B)$, $g \in \mathcal{C}(B, C)$, the following equalities hold.

$$\begin{aligned} 1_B \circ f &= f, \\ g \circ 1_B &= g. \end{aligned}$$

We say that a category \mathcal{C} is a small category if its class of objects is a set.

For any category \mathcal{C} , we will denote by \mathcal{C}^{opp} the category whose morphisms are defined by

$$\mathcal{C}^{\text{opp}}(A, B) = \mathcal{C}(B, A),$$

for every pair of objects A, B of \mathcal{C} .

NOTATION 2.2. We will denote a morphism $f \in \mathcal{C}(A, B)$ often as $f : A \rightarrow B$, and also denote the source A by $\text{dom}(f)$, and the target B by $\text{codom}(f)$.

NOTATION 2.3. The following categories will appear later in the paper.

- (A) The category **Set**, whose objects are sets and whose morphisms are maps between sets.
- (B) The category **Vect_k**, where k is a field, and whose objects are k -vector spaces and whose morphisms are linear maps.
- (C) The category **Grp** of groups and homomorphisms.
- (D) The category **SL** (respectively, **SA**) of embedded semilinear (respectively, semialgebraic) sets and affine (respectively, polynomial) maps. More precisely, each object of **SL** (respectively, **SA**) is a semilinear (respectively, semialgebraic) subset $A \subset \mathbb{R}^n$ for some $n \geq 0$, and a morphism $(A \subset \mathbb{R}^n) \rightarrow (B \subset \mathbb{R}^m)$ is a map $f : A \rightarrow B$ such that there exists a commutative square

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow & & \downarrow \\ \mathbb{R}^n & \xrightarrow{g} & \mathbb{R}^m \end{array}$$

where $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is an affine (respectively, polynomial) map. In other words, f is the restriction to A of an affine (respectively, polynomial) map from \mathbb{R}^n to \mathbb{R}^m .

- (E) The category **Alg_k** of k -algebras over a field k .

- (F) The category **AffVar** $_k$ of affine varieties and the category **AffSch** $_k$ of affine k -schemes for a field k .
- (G) The category **R-Mod**, where R is a polynomial ring in finitely many variables.
- (H) The category **Top** of topological spaces.

DEFINITION 2.4 (Functors). A (covariant) functor F from a category \mathcal{A} to \mathcal{B} consists of the following:

- (1) a mapping $\text{Ob}(\mathcal{A}) \rightarrow \text{Ob}(\mathcal{B})$ (the image of A will be written as $F(A)$);
- (2) for every pair of objects A, A' of \mathcal{A} , a mapping $\mathcal{A}(A, A') \rightarrow \mathcal{B}(F(A), F(A'))$ (the image of $f \in \mathcal{A}(A, A')$ is written as $F(f)$).

The above data is subject to the following axioms.

- (a) for every pair of morphisms $f \in \mathcal{A}(A, A')$, $g \in \mathcal{A}(A', A'')$,

$$F(g \circ f) = F(g) \circ F(f);$$

- (b) for every object A of \mathcal{A} ,

$$F(1_A) = 1_{F(A)}.$$

A contravariant functor F from a category \mathcal{A} to \mathcal{B} is a covariant functor from \mathcal{A}^{opp} to \mathcal{B} .

DEFINITION 2.5 (Natural transformations between functors). Let F, G be two (covariant) functors from a category \mathcal{C} to \mathcal{D} . A natural transformation $\theta : F \rightarrow G$ is a family of arrows $(\theta_C : F(C) \rightarrow G(C))_C$ in \mathcal{D} , indexed by objects C of \mathcal{C} , such that for each pair of objects C, C' of \mathcal{C} and $f \in \mathcal{C}(C, C')$, the following diagram commutes:

$$\begin{array}{ccc} F(C) & \xrightarrow{\theta_C} & G(C) \\ \downarrow F(f) & & \downarrow G(f) \\ F(C') & \xrightarrow{\theta_{C'}} & G(C'). \end{array}$$

We will denote the class of all natural transformations between two functors F, G by $\text{Nat}(F, G)$. A natural transformation $\theta \in \text{Nat}(F, G)$ is called an isomorphism if it admits an inverse.

DEFINITION 2.6. Two functors $F : \mathcal{C} \rightarrow \mathcal{D}$ and $U : \mathcal{D} \rightarrow \mathcal{C}$ are said to be an *adjoint pair* (with F left-adjoint to U and U right-adjoint to F), if there exist for each object C of \mathcal{C} and D of \mathcal{D} , bijective maps

$$\begin{aligned} \mathcal{C}(C, U(D)) &\rightarrow \mathcal{D}(F(C), D), & f &\mapsto f^\sharp, \\ \mathcal{D}(F(C), D) &\rightarrow \mathcal{C}(C, U(D)), & g &\mapsto g_\flat, \end{aligned}$$

which are inverses to each other and are moreover natural in C and D .

More precisely, for every $k \in \mathcal{C}(A, C)$, $\ell \in \mathcal{D}(D, B)$,

$$\begin{aligned} (U(\ell) \circ f \circ k)^\sharp &= \ell \circ f^\sharp \circ F(k), \\ U(\ell) \circ g_\flat \circ k &= (\ell \circ g \circ F(k))_\flat, \end{aligned}$$

both hold.

REMARK 2.7. In Definition 2.6, the functors F, U induce functors \mathcal{F}, \mathcal{U} from the product category $\mathcal{C}^{\text{opp}} \times \mathcal{D}$ to **Set** defined by

$$\begin{aligned} \mathcal{F}(C, D) &= \mathcal{D}(F(C), D), \\ \mathcal{U}(C, D) &= \mathcal{C}(C, U(D)), \end{aligned}$$

with $\mathcal{F}(f, g), \mathcal{G}(f, g)$ defined in the obvious manner for f, g arrows in the categories \mathcal{C}^{opp} and \mathcal{D} , respectively. Then, the naturality condition in Definition 2.6 translates into the fact that the functors \mathcal{F} and \mathcal{U} are natural transformations.

EXAMPLE 2.8. The functor $F : \mathbf{Set} \rightarrow \mathbf{Grp}$, which takes a set to the free group generated by the set, and the forgetful functor $U : \mathbf{Grp} \rightarrow \mathbf{Set}$ form an adjoint pair (with F left-adjoint to U).

We also need the notion of universal elements.

DEFINITION 2.9 (Representable functors, Yoneda's lemma, and universal elements). Let \mathcal{C} be a category and F a functor from \mathcal{C} to **Set**. Then for each object A of \mathcal{C} , $\mathcal{C}(A, \cdot)$ is a functor from \mathcal{C} to **Set**. The map

$$\lambda_A : \text{Nat}(\mathcal{C}(A, \cdot), F) \rightarrow F(A)$$

defined by

$$\lambda_A(\phi) = \phi_A(\text{id}_A)$$

is bijective. This statement is referred to as Yoneda's lemma. Now, if $\phi \in \text{Nat}(\mathcal{C}(A, \cdot), F)$ is an isomorphism, then we say that F is representable (by A), and $u = \lambda_A(\phi)$ is called a *universal element* of F . The element u has the property, that for any object B of \mathcal{C} and $f \in \mathcal{C}(B, A)$, there is a unique element $t \in F(B)$, such that $u = F(f)(t)$.

If F is a contravariant functor from \mathcal{C} to (Set) , then it is representable by an object A of \mathcal{C} , if and only if the corresponding covariant functor from \mathcal{C}^{opp} to sets is representable, and in this case a universal element of this covariant functor will be called a universal element of F .

DEFINITION 2.10 (Graphs and diagram categories). A *directed graph* I is a quadruple (V, E, s, t) , where V, E are sets (referred to as the sets of vertices and edges of I) and $s, t : E \rightarrow V$ are maps. (For $e \in E$, we will sometimes refer to $s(e)$ as the source and $t(e)$ as the target of the edge e .) A homomorphism $\phi = (\phi_1, \phi_2) : I \rightarrow I'$ of directed graphs $I = (V, E, s, t), I' = (V', E', s', t')$ is a pair of maps $\phi_1 : V \rightarrow V', \phi_2 : E \rightarrow E'$ such that the two diagrams

$$\begin{array}{ccc} E & \xrightarrow{\phi_2} & E' \\ \downarrow s & & \downarrow s' \\ V & \xrightarrow{\phi_1} & V' \end{array} \quad , \quad \begin{array}{ccc} E & \xrightarrow{\phi_2} & E' \\ \downarrow t & & \downarrow t' \\ V & \xrightarrow{\phi_1} & V' \end{array}$$

commute.

For a small category \mathcal{C} , we will denote by $U(\mathcal{C})$ the directed graph whose set of vertices is the set of objects of \mathcal{C} and whose set of edges is the set of all morphisms of \mathcal{C} , along with the maps s, t taking a morphism to its codomain and domain, respectively. Even if the category \mathcal{C} is not small, we will continue to use the notation $U(\mathcal{C})$ to denote its underlying graph, keeping in mind that this will always restrict our attention to subgraphs of $U(\mathcal{C})$ with finite sets of vertices.

Let \mathcal{C} be a category and let $U(\mathcal{C})$ be the underlying directed graph. Let $I = (V, E, s, t)$ be any directed graph. By a *diagram in \mathcal{C}* , we mean a directed graph homomorphism $D : I \rightarrow U(\mathcal{C})$. The graph I will be called the *shape* of D , and we will denote by $v(D)$ (respectively, $e(D), s(D), t(D)$) the set of vertices (respectively, edges, sources, and targets of edges) of the graph I .

We say that the diagram D is *discrete* if the edge set E of I is empty.

By a *subdiagram* of a diagram $D : I \rightarrow U(\mathcal{C})$, with $I = (V, E)$, we mean the restriction $D_J : J \rightarrow U(\mathcal{C})$, with $J = (V', E')$ a full subgraph of I , that is, $V' \subset V$, and $E' = \{e \in E \mid s(e) \in V', t(e) \in V'\}$. The restrictions to not necessarily full subgraphs will be specified as *not necessarily full subdiagrams*.

For $I = (V, E, s, t)$ and two diagrams, $D_1 : I \rightarrow U(\mathcal{C}), D_2 : I \rightarrow U(\mathcal{C})$, a morphism between D_1 and D_2 is a collection of morphisms $\varphi = (\varphi_v : D_1(v) \rightarrow D_2(v))_{v \in V}$ such that for all $e \in E, D_2(e) \circ \varphi_{D_1(s(e))} = \varphi_{D_1(t(e))} \circ D_1(e)$. This defines the *category of diagrams, \mathcal{C}^I , of the category \mathcal{C} with shape I* .

REMARK 2.11. Note that in Definition 2.10, I is just a directed graph and does not have a composition operation on it. As such, there is no *a priori* assumption of functoriality/commutativity for diagrams.

It is possible to associate a category, $\mathbf{Pth}(I)$, to each directed graph I , called the path category of I , and define (path) diagrams as actual functors from $\mathbf{Pth}(I)$ to a category \mathcal{C} [Sim11]. The diagrams we consider in this paper are not functors in this sense.

Also, note that our notion of a full subdiagram is not the same as the full subfunctor of the diagram functor from the path category of I . So if I is the directed graph $1 \rightarrow 2 \rightarrow 3$ and D a diagram of a category \mathcal{C} of shape I , then the restriction of D to the full subgraph corresponding to the vertex set $J = \{1, 3\}$ is the diagram whose image is the subgraph of $U(\mathcal{C})$ consisting of two vertices $D(1), D(3)$ and an empty set of edges.

DEFINITION 2.12 (Cones and limits). Given an object W of \mathcal{C} , a *constant diagram with value W and shape $I = (V, E, s, t)$* is the diagram $D : I \rightarrow U(\mathcal{C})$, with $D(v) = W$ for all $v \in V$ and $D(e) = 1_W$ for all $e \in E$.

Given a diagram $D : I \rightarrow U(\mathcal{C})$ and an object W of \mathcal{C} , a *cone over D with vertex W* is a morphism $D' \rightarrow D$, where D' is a constant diagram of shape I with value W . Note that if D' is the constant diagram of shape I and value W , then a cone $\varphi : D' \rightarrow D$ is determined by morphisms $\varphi(v) : W \rightarrow D(v)$ such that the following diagram commutes for each $e \in E$:

$$\begin{array}{ccc}
 & W & \\
 \varphi(s(e)) \swarrow & & \searrow \varphi(t(e)) \\
 D(s(e)) & \xrightarrow{D(e)} & D(t(e)).
 \end{array} \tag{1}$$

Note that the cone over $D : I \rightarrow \mathcal{C}$ with vertex W can also be thought of as a new diagram, whose associated directed graph is obtained from the graph of I by adding one extra vertex v_0 and an edge, e_v , for each vertex v of I , with source v_0 and target v (cf. Figure 1). This diagram of course has the extra property that all subdiagrams of the form (1) commute.

Given a diagram $D : I \rightarrow U(\mathcal{C})$, with $I = (V, E, s, t)$, the map $\mathcal{C}^I(\cdot, D)$, that associates to each object W of \mathcal{C} the set of cones over D with vertex W defines a contravariant functor from \mathcal{C} to **Set**.

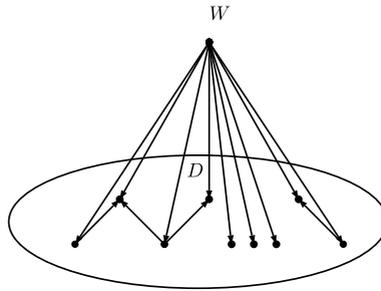


Figure 1. Cone over D with vertex W .

A universal element of this functor is called the *limit of D* (denoted by $\lim D$). In other words, $\lim D$ is a cone $\lim D : D' \rightarrow D$, where D' is a constant diagram of shape I (say with value L) such that for any cone $\varphi : D'' \rightarrow D$, where D'' is a constant diagram of shape I with value M , there exists a unique morphism $\varphi' : M \rightarrow L$ such that the following diagram commutes (denoting by φ' also the induced morphism in $\mathcal{C}^I(D'', D)$):

$$\begin{array}{ccc}
 D'' & \xrightarrow{\varphi'} & D' \\
 \searrow \varphi & & \swarrow \lim D \\
 & D &
 \end{array}$$

We will denote by $I(\lim D)$ the directed graph (V', E', s', t') , where

$$\begin{aligned}
 V' &= V \cup \{v_0\}, & v_0 &\notin V, \\
 E' &= E \cup \bigcup_{v \in V} \{e_v\},
 \end{aligned}$$

and s', t' are defined by

$$\left. \begin{aligned}
 s'(e) &= s(e), \\
 t'(e) &= t(e),
 \end{aligned} \right\} \text{ if } e \in E,$$

$$\left. \begin{aligned}
 s'(e) &= v_0, \\
 t'(e) &= v.
 \end{aligned} \right\} \text{ if } e = e_v, v \in V.$$

We will also sometimes abuse notation and refer to the value L of the limit also by $\lim D$.

EXAMPLE 2.13 (Products, pullbacks, equalizers, and composition). The limit of a discrete diagram is called a *product*. Limits of diagrams of shapes

$$\bullet \longrightarrow \bullet \longleftarrow \bullet, \quad \bullet \rightrightarrows \bullet$$

are usually referred to as pullbacks and equalizers, respectively. Note also that taking the composition of two morphisms is a particular case of taking limits of diagrams of shape

$$\bullet \longrightarrow \bullet \longrightarrow \bullet.$$

DEFINITION 2.14 (Cocones and colimits). By dualizing (that is, reversing the direction of the arrow) in Definition 2.12, we obtain the notion of cocones and colimits of diagrams. Given a diagram $D : I \rightarrow U(\mathcal{C})$ and an object W of \mathcal{C} , a *cocone over D with vertex W* is a morphism $D \rightarrow D'$, where D' is a constant diagram of shape I with value W . Note that if D' is the constant diagram of shape I and value W , then a cone $\varphi : D \rightarrow D'$ is determined by morphisms $\varphi(v) : D(v) \rightarrow W$ such that the following diagram commutes for each $e \in E$:

$$\begin{array}{ccc} & W & \\ \varphi(s(e)) \nearrow & & \nwarrow \varphi(t(e)) \\ D(s(e)) & \xrightarrow{D(e)} & D(t(e)) \end{array}.$$

Given a diagram $D : I \rightarrow U(\mathcal{C})$, with $I = (V, E, s, t)$, the map $\mathcal{C}^I(D, \cdot)$, that associates to each object W of \mathcal{C} the set of cocones over D with vertex W defines a covariant functor from \mathcal{C} to **Set**. A universal element of this functor is called the *colimit of D* (denoted by $\text{colim } D$). In other words, $\text{colim } D$ is a cocone $\text{colim } D : D \rightarrow D'$, where D' is a constant diagram of shape I (say with value C) such that for any cocone $\varphi : D \rightarrow D''$, where D'' is a constant diagram of shape I with value M , there exists a unique morphism $\varphi' : C \rightarrow M$ such that the following diagram commutes (denoting by φ' also the induced morphism in $\mathcal{C}^I(D'', D)$):

$$\begin{array}{ccc} D' & \xrightarrow{\varphi'} & D'' \\ \text{colim } D \swarrow & & \searrow \varphi \\ & D & \end{array}.$$

We will denote by $I(\text{colim } D)$ the directed graph (V', E', s', t') , where

$$\begin{aligned} V' &= V \cup \{v_0\}, & v_0 &\notin V, \\ E' &= E \cup \bigcup_{v \in V} \{e_v\}, \end{aligned}$$

and s', t' are defined by

$$\left. \begin{aligned} s'(e) &= s(e), \\ t'(e) &= t(e), \end{aligned} \right\} \text{ if } e \in E,$$

$$\left. \begin{aligned} s'(e) &= v, \\ t'(e) &= v_0. \end{aligned} \right\} \text{ if } e = e_v, v \in V.$$

We will sometimes abuse notation and refer to the value C of the colimit also by $\text{colim } D$.

EXAMPLE 2.15 (Coproducts, pushforwards, and coequalizers). By reversing the direction of the arrows of the diagrams in Example 2.13 and taking colimits, we obtain the definitions of coproducts, pushforwards, and coequalizers (respectively, from the definitions of products, pullbacks, and equalizers).

3. Definition of diagram computations

In this section, we define *diagram computations* and *categorical complexity*. As explained in Section 1 (Introduction), diagram computations come in three different flavors—namely, limit, colimit, and mixed limit–colimit computations. The associated notions of complexities of diagrams will be called limit, colimit, and mixed complexity, respectively.

We fix a category \mathcal{C} for the rest of this section and also fix a set \mathcal{A} of morphisms in \mathcal{C} . The morphisms in \mathcal{A} will be referred to as the *basic morphisms* in \mathcal{C} .

3.1. Limit and colimit computations. We define a notion of computation in \mathcal{C} , called a *limit computation*, by starting with these basic morphisms and adding a finite limit at each step; similarly, in a *colimit computation*, we build objects by adding colimits of subdiagrams.

DEFINITION 3.1. A *limit computation* (respectively, a *colimit computation*) in \mathcal{C} is a finite sequence of diagrams (D_0, \dots, D_s) , with $D_j : I_j = (V_j, E_j, s_j, t_j) \rightarrow U(\mathcal{C})$, $0 \leq j \leq s$, where we have the following:

- (i) $D_0(e) \in \mathcal{A}$ for each edge e of I_0 .
- (ii) For each $i = 1, \dots, s$, D_i is obtained from D_{i-1} by adding a limit or colimit cone of a subdiagram. More precisely, there is a subdiagram $D_{i-1}|_{J_i}$ of D_{i-1} with

$$J_i = (V'_{i-1}, E'_{i-1}, s'_i = s_i|_{E'_{i-1}}, t'_i = t_i|_{E'_{i-1}}), \quad V'_{i-1} \subset V_{i-1}, E'_{i-1} \subset E_{i-1}$$

such that denoting $I(\lim D_{i-1}|_{J_i}) = (V', E', s', t')$ (respectively, $I(\operatorname{colim} D_{i-1}|_{J_i}) = (V', E', s', t')$),

(1) $V_i = V_{i-1} \cup \{v_0\}$, where v_0 is the unique vertex of $V' \setminus V_{i-1}$;

(2) $E_i = E_{i-1} \cup \bigcup_{v \in V'_{i-1}} \{e_v\}$;

(3) for $e \in E_i$,

$$\left. \begin{aligned} s_i(e) &= s_{i-1}(e), \\ t_i(e) &= t_{i-1}(e), \end{aligned} \right\} \text{ for } e \in E_{i-1},$$

$$\left. \begin{aligned} s_i(e) &= v_0 \text{ (respectively, } v), \\ t_i(e) &= v \text{ (respectively, } v_0), \end{aligned} \right\} \text{ for } e = e_v, v \in V'_{i-1};$$

(4) for $e \in E_i$,

$$D_i(e) = D_{i-1}(e) \quad \text{for } e \in E_{i-1},$$

$$D_i(e) = (\lim D_{i-1}|_{J_i})_v \text{ (respectively, } (\operatorname{colim} D_{i-1}|_{J_i})_v),$$

$$\text{for } e = e_v, v \in V'_{i-1},$$

(cf. Definitions 2.10, 2.12, and 2.14).

(iii) (*Constructivity*) For each $i, 0 < i \leq s$, if the unique vertex $v_0 \in V_i \setminus V_{i-1}$ belongs to J_j for some $j, i < j \leq s$, then J_i is a subgraph of J_j . (In other words, if a limit $L_i = \lim D_{i-1}|_{J_i}$ (respectively, colimit C_i) produced in the i th step of the computation is used again in the subdiagram $D_{j-1}|_{J_j}$ used at the j th step of the computation, then J_i is a subgraph of J_j , that is, the subdiagram that produced L_i (respectively, C_i) must be a subdiagram of $D_{j-1}|_{J_j}$.)

The computation (D_0, \dots, D_s) is said to *compute a diagram* D if D is isomorphic to a (not necessarily full) subdiagram of D_s . In particular, an object in \mathcal{C} is computed by (D_0, \dots, D_s) if an object isomorphic to it appears in D_s .

REMARK 3.2. Note that in Definition 3.1, we are not assuming that all limits or colimits of finite diagrams exist but in each particular computation (D_0, \dots, D_s) , for each $i, 0 \leq i < s$, the limit or colimit of the subdiagram $D_i|_{J_i}$ of D_i is assumed to exist. Thus, the notion of a limit/colimit computation is still well defined even if the category \mathcal{C} does not admit limits or colimits of all finite diagrams. See also Remark 3.4.

REMARK 3.3. One could take Parts (i) and (ii) as the definition of limit (respectively, colimit) computation. However, in order that our notion of categorical complexity is closer to the classical notions—such as circuit

complexity in certain relevant categories (see Section 4), we also consider the constructivity condition. This roughly means that the limit (or colimit) computation does not forget how an object was constructed. It also prevents objects of exponential rank/size from being constructed; cf. Example 3.16.

REMARK 3.4. Of course, one may not be able to obtain every object, morphism, or diagram from a given set of basic morphisms \mathcal{A} in a category \mathcal{C} . We will think of such objects/morphisms/diagrams as noncomputable in \mathcal{C} with respect to \mathcal{A} .

We now describe a basic syntax for writing down the limit or colimit computations. The computation is described by a set expressions, each expression in a line. The first kind of expression is of the form

i. source, f, target

and describes objects and/or morphisms that are added to D_0 . Here, *i* is an identifier that can be any string. In subsequent lines, the identifier ‘*i*’ is used to refer to the source, and ‘*i*’ is used to refer to the target of the basic morphism $f \in \mathcal{A}$ that is added to D_0 by this expression. *source* and *target* are the identifiers of the vertices that are the intended source and target of the new morphism being attached to D_0 .

If the source is a new vertex that did not exist in the diagram before, then we write *i. i, f, target*, or *i. _, f, target* for it. If only the target is new, we write *i. source, f, i'* or *i. source, f, _*; we write *i. i, f, i'* or *i. _, f, _* if both are new, distinct vertices, and *i. _, f, i* if both are new and are the same vertex.

There is no need to list all the morphisms in D_0 at the beginning, so we will have these steps as intermediate steps as well; as long as the morphisms attach only to other vertices in D_0 , they be can be considered as part of D_0 .

The second kind of expression is that of the form

i. lim(a,b,...)

which describes steps where a limit is added to the subdiagram. The identifiers *a, b, ...* describe the vertices in the subdiagram whose limit is being taken. In subsequent steps, *i* is used to refer to the limit added. Similarly, we write *i. colim(a,b,...)* for describing colimit computations. We may use the notation *i*->*a* to refer to morphisms created during the computation.

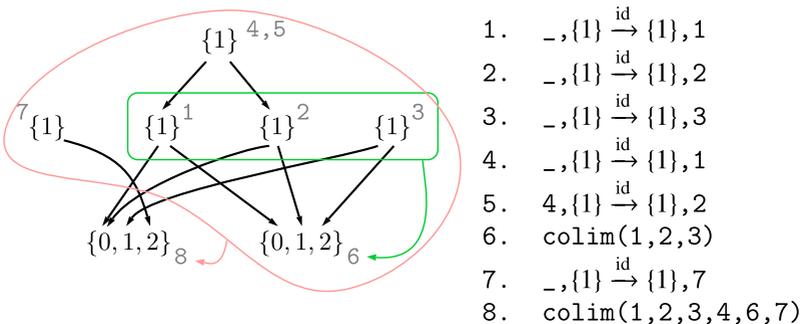
We start with two basic examples about constructions in the category of sets.

EXAMPLE 3.5. Let \mathcal{C} be the category of sets and let \mathcal{A} consist of a single morphism $\text{id} : \{1\} \rightarrow \{1\}$. Consider the colimit computation described by

1. $_, \{1\} \xrightarrow{\text{id}} \{1\}, 1$
2. $_, \{1\} \xrightarrow{\text{id}} \{1\}, 2$
- ...
- n. $_, \{1\} \xrightarrow{\text{id}} \{1\}, n$
- n+1. $\text{colim}(1, 2, \dots, n)$

For each $k \leq n$, the step $k. _, \{1\} \xrightarrow{\text{id}} \{1\}, k$ is adding a new copy of $\{1\}$ to the diagram (that is, the vertex labeled k with a self-loop corresponding to id). In the end, $n+1$ is the set with n elements.

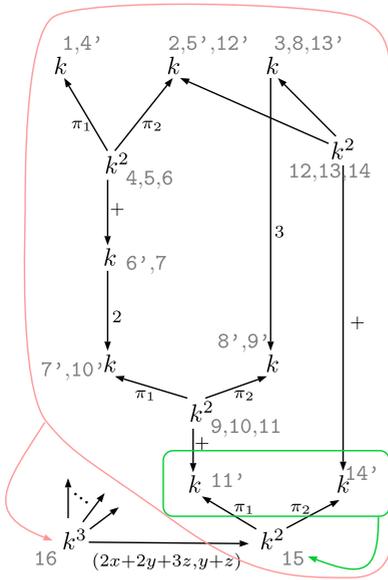
EXAMPLE 3.6. Continuing with the previous example, we now make a colimit computation that produces the morphism $\{0, 1, 2\} \xrightarrow{f} \{0, 1, 2\}$ in the category of sets, where $f(0) = 0, f(1) = 0, f(2) = 1$.



The morphism $6 \rightarrow 8$ is f , in the sense that the full subdiagram containing 6 and 8 is isomorphic to $\{0, 1, 2\} \xrightarrow{f} \{0, 1, 2\}$.

We will come back to sets later. We now discuss a more detailed example where we annotate each step in the computation.

EXAMPLE 3.7. Consider the category \mathbf{Vect}_k of vector spaces over a field k . Let \mathcal{A} consist of the scalar multiplication morphisms $k \xrightarrow{c} k$ for each $c \in k$, the addition morphism $k^2 \xrightarrow{+} k$, the two projections $\pi_1, \pi_2 : k^2 \rightarrow k$, and morphisms $0 \rightarrow k, k \rightarrow 0$. Say, the characteristic of k is 0 and we wish to compute the morphism $f : k^3 \rightarrow k^2, f(x, y, z) = (2x + 2y + 3z, y + z)$. We describe the computation as follows.



1. $_, k \xrightarrow{1} k, 1$
2. $_, k \xrightarrow{1} k, 2$
3. $_, k \xrightarrow{1} k, 3$
4. $_, k \times k \xrightarrow{\pi_1} k, 1$
5. $4, k \times k \xrightarrow{\pi_2} k, 2$
6. $4, k \times k \xrightarrow{+} k, 6'$
7. $6', k \xrightarrow{2} k, 7'$
8. $3, k \xrightarrow{3} k, 8'$
9. $_, k \times k \xrightarrow{\pi_2} k, 8'$
10. $9, k \times k \xrightarrow{\pi_1} k, 7'$
11. $9, k \times k \xrightarrow{+} k, 11'$
12. $_, k \times k \xrightarrow{\pi_1} k, 2$
13. $12, k \times k \xrightarrow{\pi_2} k, 3$
14. $12, k \times k \xrightarrow{+} k, 14'$
15. $\lim(11', 14')$
16. $\lim(1, 1', 2, 2', \dots, 14, 14', 15)$

REMARK 3.8. Two facts about limits and colimits are useful in thinking about the above example and other computations.

The first (as already noted in Example 2.13) is that if $f : X \rightarrow Y$ and $g : Y \rightarrow Z$ are morphisms, then the limit of the diagram $X \xrightarrow{f} Y \xrightarrow{g} Z$ is (isomorphic to) X , and the induced morphism $X \rightarrow Z$ is equal to the composition $g \circ f$. So, compositions are obtained using limits. The second fact is the following. Let $D : I = (V, E, s, t) \rightarrow \mathcal{C}$ be a diagram, and $L = \lim D$, and let $\varphi : C \rightarrow D$ be a cone over D , where C is a constant diagram of shape I and value X . Let $I' = (V', E', s', t')$ be the graph, with $V' = V \cup \{v_0, v_1\}$, $E' = E \cup \bigcup_{v \in V} \{e_v, e'_v\}$,

$$\left. \begin{aligned} s'(e) &= s(e), \\ t'(e) &= t(e), \end{aligned} \right\} \text{ for } e \in E,$$

$$\left. \begin{aligned} s'(e) &= v_0, \\ t'(e) &= v, \end{aligned} \right\} \text{ for } e = e_v, \quad \left. \begin{aligned} s'(e) &= v_1, \\ t'(e) &= v, \end{aligned} \right\} \text{ for } e = e'_v, \quad \left. \right\} \text{ for } v \in V,$$

and $D' : I' \rightarrow \mathcal{C}$ be the diagram defined by,

$$\begin{aligned} D'(v) &= D(v), & \text{for } v \in V, \\ D'(v_0) &= L, \\ D'(v_1) &= X, \end{aligned}$$

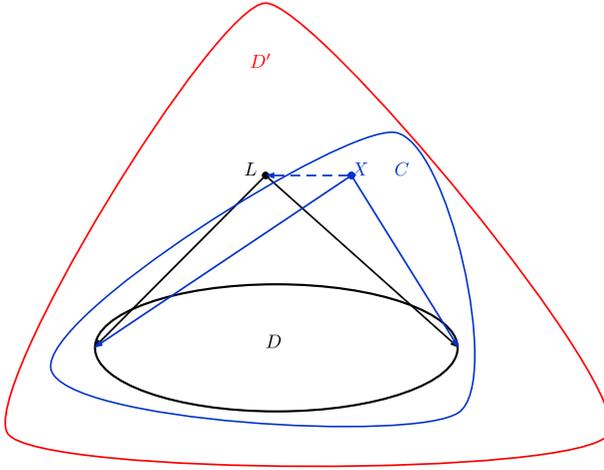


Figure 2. One extra limit.

$$\begin{aligned} D'(e) &= D(e), & \text{for } e \in E, \\ D'(e) &= (\lim D)_v, & \text{for } e = e_v, v \in V, \\ D'(e) &= \varphi_v, & \text{for } e = e'_v, v \in V. \end{aligned}$$

Then, $\lim D'$ is isomorphic to X , and morphism $X \rightarrow L$ in this corresponding limit cone is the unique morphism coming from the universal property of the limit L . Thus, in a limit computation, once the cones corresponding to L and X are computed, in order to obtain the morphism $X \rightarrow L$, which is implied by the universal property of limits, one needs to take just one additional limit.

A similar fact holds for colimits (with the arrows reversed). In other words, let $D : I = (V, E, s, t) \rightarrow \mathcal{C}$ be a diagram, and $M = \text{colim } D$, and let $\varphi : D \rightarrow C$ be a cocone over D , where C is a constant diagram of shape I and value Y . Then, in a colimit computation, once the cocones corresponding to M and Y are computed, in order to obtain the morphism $M \rightarrow Y$, which is implied by the universal property of colimits, one needs to take just one additional colimit.

3.2. Mixed limit–colimit computations. We now discuss computations where we can use limits and colimits together. We call these *mixed computations*.

DEFINITION 3.9. A *mixed computation* is a finite sequence (D_0, \dots, D_s) of diagrams with $D_i : I_i \rightarrow \mathcal{U}(\mathcal{C})$, where D_0 consists only of morphisms in \mathcal{A} , and for each $i = 1, \dots, s$, D_i is obtained from D_{i-1} by adding either the limit of a subdiagram with the corresponding cone morphisms or colimit of a subdiagram with the corresponding cocone morphisms.

Note that there is no constructivity assumption for mixed computations. To include it would have been too restrictive and would have prevented natural applications like gluing geometric objects already constructed.

EXAMPLE 3.10 (Monotone Boolean circuits). Let \mathcal{B}_n be the lattice of subsets of $\{0, 1\}^n$, which is a category whose objects are the subsets of $\{0, 1\}^n$, and with $\text{Hom}_{\mathcal{B}_n}(A, B) = \{\iota\}$ if $A \subset B$, where $\iota : A \rightarrow B$ is the inclusion, and $\text{Hom}_{\mathcal{B}_n}(A, B) = \emptyset$ otherwise. Let $Z_i = \{(x_1, \dots, x_n) \in \{0, 1\}^n \mid x_i = 1\}$. Let \mathcal{A}_n be the set of basic morphisms $\{\text{id}_{Z_i} \mid i = 1, \dots, n\}$. Let $\mathcal{B} = \coprod_{n=1}^{\infty} \mathcal{B}_n$ be the disjoint union of these categories and $\mathcal{A} = \coprod_{n=1}^{\infty} \mathcal{A}_n$.

We show that there is a correspondence between multi-output monotone Boolean circuits with n inputs and mixed computations in \mathcal{B}_n . Given a monotone Boolean circuit, consider the corresponding straight-line program with Boolean operations. Start a mixed computation in \mathcal{B}_n with a copy of each of the subsets Z_i . These correspond to the input variables z_1, \dots, z_n of the straight-line program. Subsequent entries z_{n+1}, z_{n+2}, \dots will correspond to newly constructed objects in the mixed computation. For each operation in the straight-line program of the form $z_i = z_j \wedge z_k$, take the limit of the objects corresponding to z_j and z_k ; similarly, take the colimit for $z_i = z_j \vee z_k$. To make a straight-line program from a mixed computation, start with the input variables z_1, \dots, z_n and add $k - 1$ new \wedge operations for each limit of k objects and $k - 1$ new \vee operations for each colimit of k objects (ignoring the arrows does not change the limit/colimit).

Thus, mixed computations in \mathcal{B}_n are in direct correspondence with monotone straight-line programs or, equivalently, monotone Boolean circuits.

EXAMPLE 3.11. Consider the category **Top** of topological spaces. Let $I = [0, 1]$ be the unit interval, and let the basic morphisms consist of $I \xrightarrow{\text{id}} I, I \rightarrow pt$, $pt \xrightarrow{0} I$, and $pt \xrightarrow{1} I$. We can build all cubes using limits and then can glue these using colimits to construct many topological spaces.

We will reconsider mixed computations when we look at the complexity of projective schemes.

3.3. Cost and complexity. Let $c_0 : \mathcal{A} \rightarrow \mathbb{Z}_{\geq 0}$ be any function, considered as the *cost* of the basic morphisms.

DEFINITION 3.12. The *cost* of the computation (D_0, \dots, D_s) is the number of steps plus the cost of the initial diagram D_0 consisting of basic morphisms, that is,

$$c(D_0, \dots, D_s) = s + \sum_{f \in \text{edges}(I_0)} c_0(D_0(f)).$$

If c_0 is not specified, then we consider it to be the constant function 1, so every basic morphism will have unit cost. This will be the case in almost every example we consider.

DEFINITION 3.13. The *limit* (respectively, *colimit*; respectively, *mixed*) complexity, $c_{\mathcal{C}, \mathcal{A}, c_0}^{\text{lim}}(D)$ (respectively, $c_{\mathcal{C}, \mathcal{A}, c_0}^{\text{colim}}(D)$; respectively, $c_{\mathcal{C}, \mathcal{A}, c_0}^{\text{mixed}}(D)$), of a diagram D in a category \mathcal{C} is the cost of the limit (respectively, colimit; respectively, mixed) computation using basic morphisms \mathcal{A} , which has the smallest cost among all such computations that compute D . When c_0 is the constant function 1, we will omit it from the subscript and just write $c_{\mathcal{C}, \mathcal{A}}^{\text{lim}}(D)$, $c_{\mathcal{C}, \mathcal{A}}^{\text{colim}}(D)$, or $c_{\mathcal{C}, \mathcal{A}}^{\text{mixed}}(D)$.

For a morphism $f : X \rightarrow Y$ in \mathcal{C} , the complexity $c(f)$ of f is the complexity of the corresponding diagram mapping two objects and a single morphism $X \xrightarrow{f} Y$. For an object X in \mathcal{C} , the complexity $c(X)$ of X is the complexity of the diagram with one object, X .

EXAMPLE 3.14 (Gluing simplices). Let $\mathcal{C} = \mathbf{Top}$ be the category of topological spaces and let \mathcal{A} be the set of all face embeddings $\Delta_n \hookrightarrow \Delta_m$ corresponding to each strictly increasing map $[n] \rightarrow [m]$, where Δ_n is the standard n -simplex. Colimit computations correspond to gluing operations between simplices. The colimit complexity then measures how many simplices are needed to construct a given topological space by gluing.

We now go back to considering $\mathcal{C} = \mathbf{Set}$ with the basic morphisms \mathcal{A} consisting of a single morphism $\text{id} : \{1\} \rightarrow \{1\}$.

PROPOSITION 3.15 (Colimit complexity of sets). *In the category \mathbf{Set} , let*

$$\mathcal{A} = \{\text{id} : \{1\} \rightarrow \{1\}\}, c_0(\text{id}) = 1.$$

Then, for any set finite set S ,

$$c_{\mathbf{Set}, \mathcal{A}}^{\text{colim}}(S) = \text{card}(S) + 1.$$

Proof. Since finite sets of equal size are isomorphic, a computation will compute S if and only if it computes any set of cardinality equal to $\text{card}(S)$. As in Example 3.5, starting with $\text{card}(S)$ copies of $\{1\}$ and taking their colimit, we get a set of cardinality $\text{card}(S)$. So, the complexity is bounded from above by $\text{card}(S) + 1$. To see that this is the most efficient way of producing a set with $\text{card}(S)$ elements, we use Lemma 3.19, which states that if we only care about building a single object, then a colimit computation can be replaced by a single

colimit on D_0 consisting of basic morphisms. Since the identity on $\{1\}$ is the only basic morphism in this case, taking the colimit of $\text{card}(S)$ copies of $\{1\}$ is the most efficient way to obtain an object isomorphic to S . \square

EXAMPLE 3.16 (Nonconstructive colimit complexity). The following example shows the difference between colimit computations and nonconstructive computations. Consider the computation

$$\begin{aligned}
 1. & \quad _, \{1\} \xrightarrow{\text{id}} \{1\}, 1 \\
 2. & \quad _, \{1\} \xrightarrow{\text{id}} \{1\}, 2 \\
 3. & \quad \text{colim}(1, 2) \\
 4. & \quad \text{colim}(1, 2) \\
 5. & \quad \text{colim}(3, 4) \\
 6. & \quad \text{colim}(3, 4) \\
 & \quad \vdots \\
 & \quad \vdots \\
 2a-1. & \quad \text{colim}(2a-3, 2a-2) \\
 2a. & \quad \text{colim}(2a-3, 2a-2) \\
 2a+1. & \quad \text{colim}(2a-1, 2a)
 \end{aligned}$$

This computation produces a set of size 2^a . Observe that the steps 5 to $2a+1$ are not constructive. To make them constructive, one would need to include $1, \dots, 2i-2$ in the colimit in step $2i-1$, $3 \leq i \leq a+1$, and include $1, \dots, 2i-2$ in the colimit in step $2i$, $3 \leq i \leq a$. This would produce in step $2a+1$ a set with just two elements.

EXAMPLE 3.17 (Nonconstructive mixed complexity). Using mixed computations, one can construct a set of doubly exponential complexity. Consider for example the following mixed computation.

$$\begin{aligned}
 1. & \quad _, \{1\} \xrightarrow{\text{id}} \{1\}, 1 \\
 2. & \quad _, \{1\} \xrightarrow{\text{id}} \{1\}, 2 \\
 3. & \quad \text{colim}(1, 2) \\
 4. & \quad \text{colim}(1, 2) \\
 5. & \quad \text{lim}(3, 4) \\
 6. & \quad \text{lim}(3, 4) \\
 & \quad \vdots \\
 & \quad \vdots \\
 2a-1. & \quad \text{lim}(2a-3, 2a-2) \\
 2a. & \quad \text{lim}(2a-3, 2a-2) \\
 2a+1. & \quad \text{lim}(2a-1, 2a)
 \end{aligned}$$

It is easy to check that it produces a set of cardinality 2^{2^a} . So the mixed complexity of a finite set $\text{card}(S)$ is $O(\log \log \text{card}(S))$.

One important feature of categorical complexity is that it allows one to define complexity of not just single objects or even morphisms of a given category (equipped with a set of basic morphisms), but one has a notion of complexity of arbitrary (finite) diagrams of the category as well. This last notion has no analog in classical theory of computational complexity.

We illustrate this feature in the following simple example.

EXAMPLE 3.18. We will consider the colimit complexity (in the category \mathbf{Vect}_k , where k is a field) of two diagrams consisting of the inclusion morphisms of n^2 different subspaces of dimension two in a $2n$ -dimensional k -vector space.

In the first diagram, the subspaces are assumed to be generic (Case (a) below), and we prove that the colimit complexity of the diagram is in $O(n^3)$ (which agrees with the intuition that to specify n^2 inclusions of two-dimensional subspaces in a $2n$ -dimensional vector space, we need to specify a matrix in $k^{2n \times 2n^2}$).

In the second diagram, the subspaces are in a special position (Case (b) below), and we prove that the colimit complexity of the diagram is in $O(n^2)$ (intuitively, only $2n$ distinct columns appear in the $2n \times 2n^2$ corresponding to the inclusion if the basis vectors are chosen properly). Categorical complexity helps in quantifying the distinction in the complexity of the two diagrams having the same shape. There is no analog in classical computational complexity, which deals mainly with membership questions (and thus from the point of view of category theory complexities of objects rather than general diagrams) of this kind of distinction.

We consider the category \mathbf{Vect}_k of vector spaces over a field k and let \mathcal{A} consist of the scalar multiplication morphisms $k \xrightarrow{c} k$ for each $c \in k$, the addition morphism $k^2 \xrightarrow{+} k$, the two projections $\pi_1, \pi_2 : k^2 \rightarrow k$, and morphisms $0 \rightarrow k$, $k \rightarrow 0$ (as in Example 3.7).

- (a) Let $n \geq 2$, $S_1, \dots, S_{n^2} \subset V$ be subspaces of a finite-dimensional k -vector space V , and suppose that $\dim V = 2n$, $\dim S_i = 2$, $1 \leq i \leq n^2$, and $\dim(S_i \cap S_j) = 0$, $1 \leq i < j \leq n^2$.

Consider the diagram shown in Figure 3, where the ϕ_i , $1 \leq i \leq n^2$, are the inclusion homomorphisms.

We now show how to produce the above diagram in Figure 3 using a colimit computation.

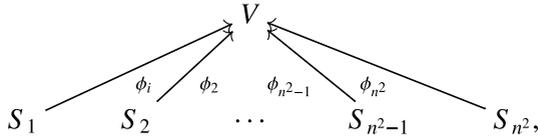


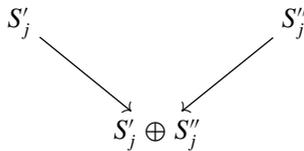
Figure 3. Subspace diagram.

Let for each $i, 1 \leq i \leq n^2, S'_i, S''_i$ be one-dimensional subspaces of S_i such that $S_i = S'_i \oplus S''_i$, and let ψ'_i, ψ''_i denote the inclusions of S'_i, S''_i into V , respectively.

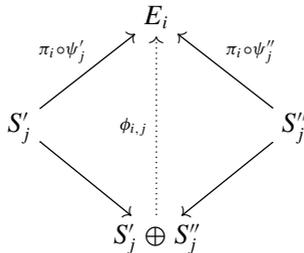
Similarly, let $E_1, \dots, E_{2n} \subset V$ be one-dimensional subspaces of V such that $V = E_1 \oplus \dots \oplus E_{2n}$. Then, V is canonically isomorphic to the direct product $E_1 \times \dots \times E_{2n}$ and let $\pi_i : V \rightarrow E_i$ denote the canonical projections.

For each $1 \leq i \leq 2n, 1 \leq j \leq n^2$, we first use colimit computation to obtain a diagram computing the $2n^3$ linear morphisms, $\phi_{i,j} = \pi_i \circ (\psi'_j \oplus \psi''_j) : S'_j \oplus S''_j \rightarrow E_i$, to obtain the diagram shown in Figure 4.

The colimit computation to obtain the diagram shown in Figure 4 is as follows. For each $j, 1 \leq j \leq n$, first take the colimit of S'_j and S''_j to obtain the diagram



and then successively for each $i, 1 \leq i \leq n$, take colimits of the diagrams



and observe (cf. Remark 3.8) that we obtain the morphisms $\phi_{i,j}$. Note that the morphisms $\pi_i \circ \psi'_j, \pi_i \circ \psi''_j$ being morphisms between one-dimensional k -vector spaces are basic. It is easy to check that the total cost of the colimit computation described above is

$$2n^3 + n^2 + 2n^3 = 4n^3 + n^2.$$

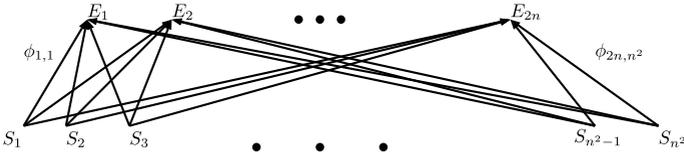


Figure 4. Subspace diagram before taking the last colimit.

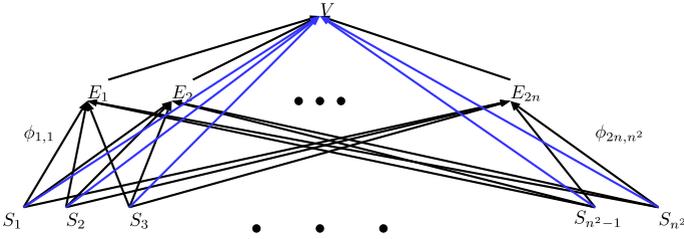


Figure 5. Diagram after taking the last colimit.

We then take the colimit of the whole diagram computed till now (as shown in Figure 5), and obtain the desired diagram as a subdiagram (whose morphisms are shown in blue in Figure 5).

The total cost is $4n^3 + n^2 + 1 = O(n^3)$.

- (b) We now consider subspaces S_1, \dots, S_{n^2} in a special position, and prove that colimit complexity of the corresponding diagram can be much smaller.

Let $V = V' \oplus V''$, where $\dim V' = \dim V'' = n$. Let $L'_i \subset V', L''_i \subset V'', 1 \leq i \leq n$ be subspaces with $\dim L'_i = \dim L''_i = 1, 1 \leq i \leq n$, and suppose that S_1, \dots, S_{n^2} are the subspaces $L'_i \oplus L''_j \subset V = V' \oplus V'', 1 \leq i, j \leq n$.

Let V' (respectively, V'') be the direct sum of one-dimensional subspaces E'_1, \dots, E'_n (respectively, E''_1, \dots, E''_n). Then $V' = E'_1 \oplus \dots \oplus E'_n, V'' = E''_1 \oplus \dots \oplus E''_n$, and moreover, V' (respectively, V'') is also canonically isomorphic to the direct product $E'_1 \times \dots \times E'_n$ (respectively, $E''_1 \times \dots \times E''_n$). Let for $1 \leq i \leq n, \pi'_i : V' \rightarrow E'_i$ (respectively, $\pi''_i : V'' \rightarrow E''_i$) denote the canonical projections. Also, for $1 \leq j \leq n$, let $\psi'_j : L'_j \rightarrow V'$ (respectively, $\psi''_j : L''_j \rightarrow V''$) denote the inclusion morphisms, and denote $\phi'_{i,j} = \pi'_i \circ \psi'_j$ (respectively, $\phi''_{i,j} = \pi''_i \circ \psi''_j$).

In this case, we can construct the diagram in Figure 3 in the following way.

First construct the diagram containing the morphisms $\phi'_{i,j}$ and $\phi''_{i,j}$ as shown in Figure 6. This costs $2n^2$.

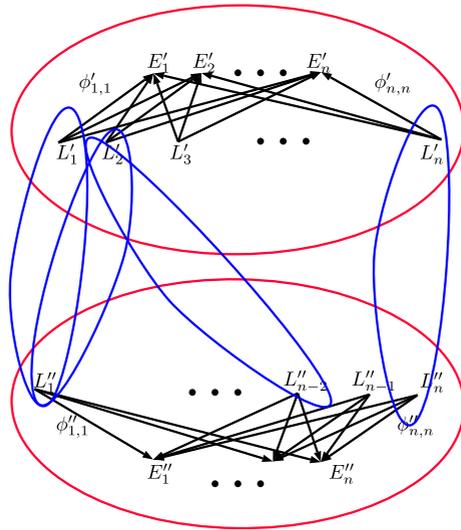


Figure 6. Colimit computation of arrangement of subspaces in a special position before taking any colimits.

Next take colimits of the n^2 subdiagrams circled in blue (corresponding to pairs of vertices $L_i, L'_j, 1 \leq i, j \leq n$) to obtain the diagram shown in Figure 7. Finally, take a colimit of the entire diagram constructed so far to obtain the diagram shown in Figure 8. It is an easy exercise to check that the required diagram occurs as a subdiagram (whose morphisms are shown using blue broken arrows in Figure 8) of the diagram so obtained. The total cost is $3n^2 + 1 = O(n^2)$.

Thus, while both arrangements of linear subspaces $S_1, \dots, S_{n^2} \subset V$ in Parts (a) and (b) contain the same number of subspaces, and would require the same number of polynomials to define (say, using a first-order formula in the language of reals), the second one is ‘simpler’ than the first in terms of its categorical colimit complexity.

3.4. Useful facts about limit and colimit computations. We now collect a few facts that will be useful for proving statements about objects and morphisms computed by limit and colimit computations.

The following lemma, which was already used in the proof of Proposition 3.15, shows that if the aim is to produce a specific object, intermediate steps in a

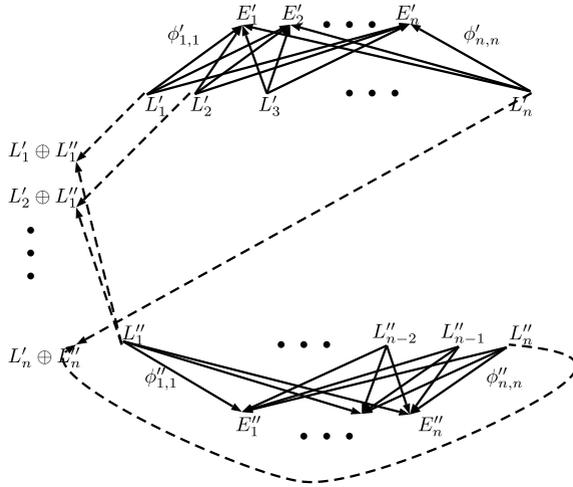


Figure 7. Colimit computation of arrangement of subspaces in a special position after taking n^2 colimits.

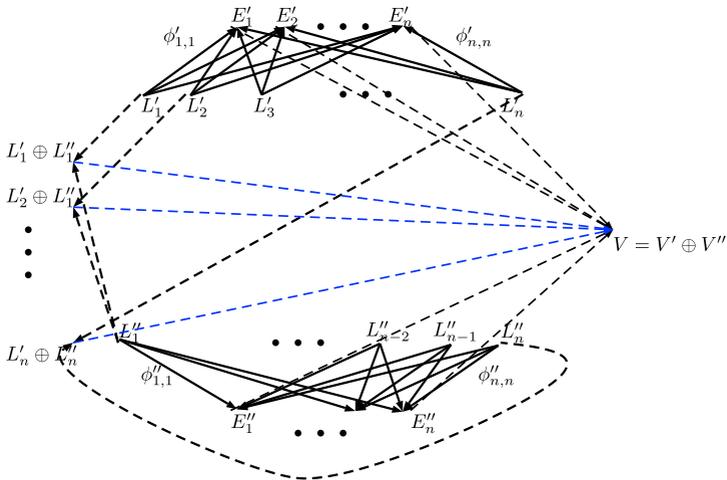


Figure 8. Colimit computation of arrangement of subspaces in a special position after taking the final colimit.

limit or colimit computation are unnecessary. The key here is the constructivity assumption.

LEMMA 3.19. Assume \mathcal{C} has finite limits (respectively, colimits). An object produced in a limit computation (respectively, colimit computation) is a limit (respectively, colimit) of a diagram consisting only of basic morphisms.

Proof. Let (D_0, \dots, D_s) be a limit computation and let X be an object appearing in D_s . The point of the statement is that constructivity ensures that the information that would be added in intermediate limits is also included in the final limit that would produce X .

More precisely, let $L_i = \lim D_{i-1}|_{J_i}$ be the limit added to the diagram at the i th step. Let $J'_i = I_0 \cap J_i$. So we have that $D_{i-1}|_{J'_i}$ is the portion of the subdiagram of $D_{i-1}|_{J_i}$, which is also in D_0 . We claim that $L_i \cong \lim D_{i-1}|_{J'_i}$. Indeed, the universal property of limits and constructivity imply that cones from any object Z to $D_{i-1}|_{J'_i}$ can be uniquely extended to cones from Z to $D_{i-1}|_{J_i}$, and therefore $\lim D_{i-1}|_{J'_i}$ satisfies the same universal property as L_i .

The analogous proof holds for colimits. \square

The following remarks are very useful for working with objects and morphisms produced in limit computations.

REMARK 3.20. If the category \mathcal{C} has finite products and equalizers, then we can write any limit L as an equalizer; see for example, [Awo10, 5.4]. More precisely, suppose that $J = (V, E, s, t)$ is a finite graph and $D : J \rightarrow U(\mathcal{C})$ a diagram. Consider the following diagram:

$$\prod_{v \in V} D(v) \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \prod_{e \in E} D(t(e)), \quad (2)$$

where ϕ is the canonically defined morphism from $\prod_{v \in V} D(v)$ to the product $\prod_{e \in E} D(t(e))$ induced from the morphisms

$$\left(\text{pr}_{t(e)} : \prod_{v \in V} D(v) \rightarrow t(e) \right)_{e \in E},$$

and ψ is the canonically defined morphism from $\prod_{v \in V} D(v)$ to the product $\prod_{e \in E} D(t(e))$ induced from the morphisms

$$\left(D(e) \circ \text{pr}_{s(e)} : \prod_{v \in V} D(v) \rightarrow t(e) \right)_{e \in E},$$

with both maps being defined by the universal property of products.

Let $L \xrightarrow{f} \prod_{v \in V} D(v)$ in the following diagram be the equalizer of diagram (2):

$$L \xrightarrow{f} \prod_{v \in V} D(v) \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \prod_{e \in E} D(t(e)) .$$

Then the object L along with the morphisms

$$(\text{pr}_{D(v)} \circ f : L \rightarrow D(v))_{v \in V}$$

is isomorphic to the limit, $\lim D$, of D .

REMARK 3.21. If $X \xrightarrow{f} Y$ is a morphism computed by a limit computation, and neither of X and Y is in D_0 , then X must have been computed as a limit of a diagram that contains Y . By constructivity, the diagram whose limit is X must contain the diagram that produced Y as a subdiagram. Therefore, if $X = \lim D$, where $D : (V, E) \rightarrow U(\mathcal{C})$ is a diagram of basic morphisms, then $Y = \lim D'$, where D' is the subdiagram corresponding to a full subgraph $(V', E') \subset (V, E)$. We then have a commuting diagram

$$\begin{array}{ccccc} X & \longrightarrow & \prod_{v \in V} D(v) & \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} & \prod_{e \in E} D(t(e)) . \\ \downarrow f & & \downarrow \pi & & \downarrow \pi' \\ Y & \longrightarrow & \prod_{v \in V'} D(v) & \begin{array}{c} \xrightarrow{\psi'} \\ \xrightarrow{\phi'} \end{array} & \prod_{e \in E'} D(t(e)) \end{array}$$

In particular, f is induced by the projection π .

We have corresponding facts for colimits as well.

REMARK 3.22. If the category \mathcal{C} has finite coproducts and coequalizers, then we can write any colimit M as a coequalizer. More precisely, suppose that $J = (V, E, s, t)$ is a finite graph and $D : J \rightarrow U(\mathcal{C})$ is a diagram. Consider the following diagram:

$$\coprod_{e \in E} D(s(e)) \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \coprod_{v \in V} D(v) , \tag{3}$$

where ϕ is the canonically defined morphism from $\coprod_{e \in E} D(s(e))$ to $\coprod_{v \in V} D(v)$ induced from the morphisms

$$\left(j_{s(e)} : D(s(e)) \rightarrow \coprod_{v \in V} D(v) \right)_{e \in E} ,$$

and ψ is the canonically defined morphism from $\coprod_{e \in E} D(s(e))$ to $\coprod_{v \in V} D(v)$ induced from the morphisms

$$\left(j_{r(e)} \circ D(e) : D(s(e)) \rightarrow \coprod_{v \in V} D(v) \right)_{e \in E},$$

with both maps being defined by the universal property of coproducts.

Let $\coprod_{v \in V} D(v) \xrightarrow{f} M$ in the following diagram be the coequalizer of diagram (3):

$$\coprod_{e \in E} D(s(e)) \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \coprod_{v \in V} D(v) \xrightarrow{f} M.$$

Then the object M with the morphisms

$$(f \circ j_{D(v)} : D(v) \rightarrow M)_{v \in V}$$

is isomorphic to the colimit, $\text{colim } D$, of D .

REMARK 3.23. If $Y \xrightarrow{f} X$ is a morphism computed by a colimit computation, and neither of X and Y is in D_0 , then X must have been computed as a colimit of a diagram that contains Y . By constructivity, the diagram whose colimit is X must contain the diagram that produced Y as a subdiagram. Therefore, if $X = \text{colim } D$, where $D : (V, E) \rightarrow U(\mathcal{C})$ is a diagram of basic morphisms, then $Y = \text{colim } D'$, where D' is the subdiagram corresponding to a full subgraph $(V', E') \subset (V, E)$. We then have a commuting diagram

$$\begin{array}{ccccc} \coprod_{e \in E} D(s(e)) & \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} & \coprod_{v \in V} D(v) & \longrightarrow & X \\ \uparrow j' & & \uparrow j & & \uparrow f \\ \coprod_{e \in E'} D(s(e)) & \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} & \coprod_{v \in V'} D(v) & \longrightarrow & Y \end{array}$$

where j, j' are the morphisms induced by the universal property of coproducts. In particular, f is induced by the morphism j .

4. Diagram computations, circuits, and algebraic varieties

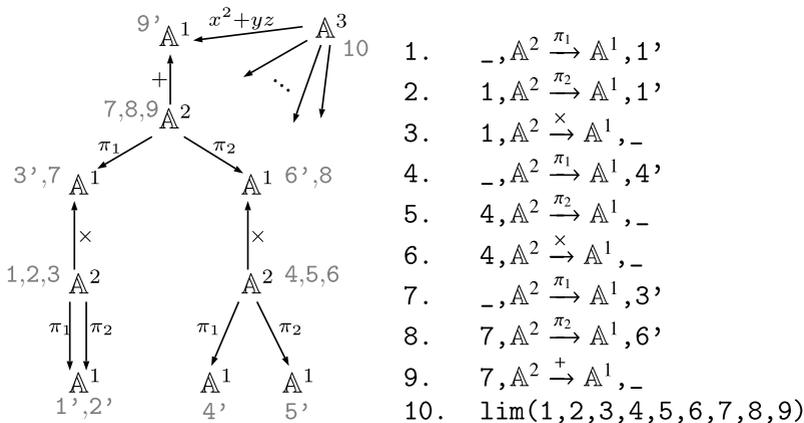
We now make a comparison between the arithmetic circuit complexity of polynomials and the limit complexity of the varieties, which are their zero-sets. For simplicity, we will start with the category of affine algebraic varieties, but what we describe will make sense in other settings like affine schemes and algebras.

4.1. Limit computations in affine varieties, schemes, and k -algebras. Let \mathcal{C} be the category \mathbf{AffVar}_k of affine algebraic varieties over a field k . Let \mathcal{A} consist of the following basic morphisms.

$$\begin{aligned}
 \mathbb{A}^1 &\xrightarrow{c} \mathbb{A}^1, \quad \text{for each } c \in k, \\
 \mathbb{A}^1 \times \mathbb{A}^1 &\xrightarrow{+} \mathbb{A}^1, \\
 \mathbb{A}^1 \times \mathbb{A}^1 &\xrightarrow{\times} \mathbb{A}^1, \\
 \mathbb{A}^1 \times \mathbb{A}^1 &\xrightarrow{\pi_1, \pi_2} \mathbb{A}^1, \\
 \mathbb{A}^1 &\rightarrow \mathbb{A}^0, \quad \text{and } \mathbb{A}^0 \xrightarrow{c} \mathbb{A}^1, \quad \text{for each } c \in k.
 \end{aligned}
 \tag{4}$$

Each of these morphisms is considered to have unit cost.

EXAMPLE 4.1. As an example, let us make a limit computation of the morphism $\mathbb{A}^3 \xrightarrow{x^2+yz} \mathbb{A}^1$ using these basic morphisms.



The following shows that a similar computation can be done to compute any polynomial map.

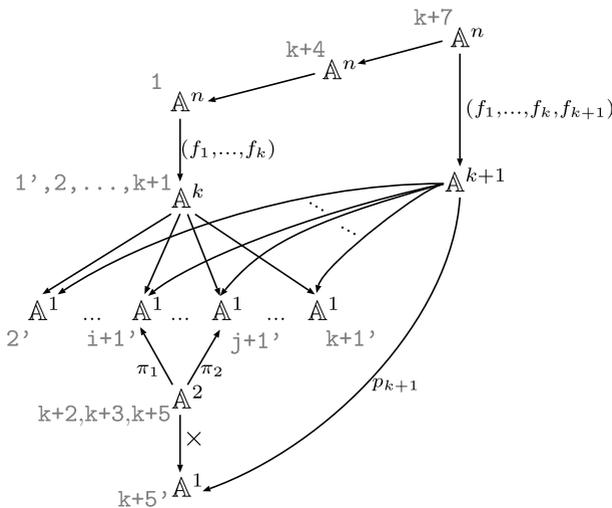
THEOREM 4.2. Let $f \in k[x_1, \dots, x_n]$ be a polynomial of degree d . Assume that f is computed by a straight-line program (cf. [Bür00, Definition 2.1]) Γ of length N . Then, the limit complexity (and therefore the categorical complexity) of the zero-set $X \subset \mathbb{A}^n$ of f is in $O(N)$.

Proof. Using the operations in the straight-line program Γ , we will construct a morphism $\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1$. X is then the limit $\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1 \xleftarrow{0} \mathbb{A}^0$.

Associated to each straight-line program of length s , there is a morphism $(f_1, \dots, f_s) : \mathbb{A}^n \rightarrow \mathbb{A}^s$, where f_1, \dots, f_s are the polynomials computed in each line of the diagram. Let $\Gamma = (\Gamma_1, \dots, \Gamma_N)$, where Γ_i is the i th instruction in the straight-line program Γ . For the first n instructions of Γ , which introduce the variables x_1, \dots, x_n as polynomials, we have the map $\mathbb{A}^n \xrightarrow{\text{id}} \mathbb{A}^n$. The space \mathbb{A}^n is constructed in a limit computation by taking the limit of n disjoint copies of the diagram consisting of \mathbb{A}^1 mapping to itself by 1. To get the map $\mathbb{A}^n \xrightarrow{\text{id}} \mathbb{A}^n$, we take the limit of the whole diagram obtained so far which costs $n + 2$. Note that our diagram also contains all the projections from the source and target of $\text{id}_{\mathbb{A}^n}$ to the n components. This is the base case of the following inductive construction.

Assume that, for $n \leq k \leq N - 1$, we have produced, in a limit computation, a morphism $(f_1, \dots, f_k) : \mathbb{A}^n \rightarrow \mathbb{A}^k$ corresponding to the polynomials computed by each line of $(\Gamma_1, \dots, \Gamma_k)$, together with the projections p_1, \dots, p_k from \mathbb{A}^k to k copies of \mathbb{A}^1 , all of which are in D_0 .

Assume that Γ_{k+1} is the multiplication of the i th and j th lines of the program, that is, $f_{k+1} = f_i f_j$. We can then perform the following steps of the computation. For convenience, we have added, as if they were basic morphisms, the portion of the diagram used for the next steps as the first $k + 1$ steps in this description:



1. $_, \mathbb{A}^n \xrightarrow{(f_1, \dots, f_k)} \mathbb{A}^k, 1'$
2. $1', \mathbb{A}^k \xrightarrow{p_1} \mathbb{A}^1, 2'$
3. $1', \mathbb{A}^k \xrightarrow{p_2} \mathbb{A}^1, 3'$
- ...
- k+1. $1', \mathbb{A}^k \xrightarrow{p_k} \mathbb{A}^1, (k+1)'$
- k+2. $_, \mathbb{A}^2 \xrightarrow{\pi_1} \mathbb{A}^1, (i+1)'$
- k+3. $k+2, \mathbb{A}^2 \xrightarrow{\pi_2} \mathbb{A}^1, (j+1)'$
- k+4. $\lim((k+2), (i+1)', (j+1)', 1', 1)$
- k+5. $k+2, \mathbb{A}^2 \xrightarrow{x} \mathbb{A}^1, _$
- k+6. $\lim(2', 3', \dots, (k+1)', (k+4)')$
- k+7. $\lim((k+4), 1, 1', 2', 3', \dots, (k+1)', k+2, (k+5)', k+6)$

In the end, the map $(k+7) \rightarrow (k+6)$ is the map $\mathbb{A}^n \xrightarrow{(f_1, \dots, f_{k+1})} \mathbb{A}^{k+1}$, and the projection maps for the next step are the maps

$$(k+6) \rightarrow 2', (k+6) \rightarrow 3', \dots, (k+6) \rightarrow (k+1)', (k+6) \rightarrow (k+5)'.$$

The process for addition steps is similar, with the step k+5 modified. For scalar multiplication, it is similar with two steps less. For constants appearing in the computation, we add a new variable and take the fiber product with $\mathbb{A}^0 \xrightarrow{c} \mathbb{A}^1$ to fix the variable to the value c . Repeating this process until $k = N$, we see that (f_1, \dots, f_N) is produced in $O(N)$ steps. We can then compose with the projection to the last coordinate, by taking a limit, to produce $\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1$. To obtain the zero-set of f , we add $\mathbb{A}^0 \xrightarrow{0} \mathbb{A}^1$ to this last \mathbb{A}^1 and take the limit.

It should also be noted that the way we made a limit computation for $\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1$ was not the most efficient, but this way gives the cleanest inductive argument. For efficiency, the intermediate limit steps can be removed; cf. Lemma 3.19. \square

REMARK 4.3. The proof above shows that we can, by starting from a number of \mathbb{A}^1 added as basic objects, construct any polynomial functions $f_1 : \mathbb{A}^n \rightarrow \mathbb{A}^1$. We will use this fact later.

We now consider a converse for Theorem 4.2 and show that, given a limit computation with low cost, an object X computed by it is *isomorphic to* the zero-set of a polynomial whose arithmetic circuit complexity is low. Since diagram computations produce objects up to isomorphism, categorical complexity does not reflect the complexity of every polynomial that might be used to cut out X in a larger space. For example, X could be the graph of a polynomial map $f : \mathbb{A}^n \rightarrow \mathbb{A}^1$ with very high arithmetic circuit complexity, but since X would be isomorphic to \mathbb{A}^n , its limit complexity would be very small, which does not say

anything about the arithmetic circuit complexity of f . This is discussed in more detail in Section 5.1.

We first recall the definition of an arithmetic circuit over a field k and that of the arithmetic circuit complexity of a polynomial map.

DEFINITION 4.4 (Arithmetic circuit complexity). An arithmetic circuit C over a field k and variables X_1, \dots, X_n is a finite directed acyclic graph such that

- (i) every vertex of the directed graph with indegree 0 is labeled either by a variable or an element of k ;
- (ii) every other vertex is labeled by either $+$ or \times .

The *size* of an arithmetic circuit is the number of vertices in the associated directed graph. Each vertex of C is associated to a polynomial in $k[X_1, \dots, X_n]$ computed at that vertex (in the obvious way). We say that a polynomial is computed by an arithmetic circuit C if it appears as the polynomial associated to some vertex of the circuit. For a tuple of polynomials $f = (f_1, \dots, f_m) \in k[X_1, \dots, X_n]$, we say that the *arithmetic circuit complexity* of the induced polynomial map $f : k^n \rightarrow k^m$ is the size of the arithmetic circuit of the smallest size that computes each of the polynomials f_1, \dots, f_m .

THEOREM 4.5. Let (D_0, \dots, D_s) be a limit computation in \mathbf{AffVar}_k , whose cost equals $C = c(D_0, \dots, D_s)$. Then we have the following:

- (i) If X is an object computed by (D_0, \dots, D_s) , then X is isomorphic to the zero-set of a polynomial morphism $\mathbb{A}^{m_1} \rightarrow \mathbb{A}^{m_2}$ whose components are polynomials of degree at most 2. The total arithmetic circuit complexity of the map is bounded above by $4C$.
- (ii) Every morphism $f : X \rightarrow Y$ in D_s is the restriction of a projection $\mathbb{A}^{m_1} \rightarrow \mathbb{A}^{m_2}$, where \mathbb{A}^{m_1} and \mathbb{A}^{m_2} are the spaces from Part (i), where X and Y are embedded respectively.

Proof. If $s = 0$, then the statement is true since the basic morphisms are $\mathbb{A}^1 \xrightarrow{c} \mathbb{A}^1$, $\mathbb{A}^1 \times \mathbb{A}^1 \xrightarrow{+, \times} \mathbb{A}^1$, $\mathbb{A}^1 \times \mathbb{A}^1 \xrightarrow{\pi_1, \pi_2} \mathbb{A}^1$, $\mathbb{A}^1 \rightarrow \mathbb{A}^0$, and $\mathbb{A}^0 \xrightarrow{c} \mathbb{A}^1$.

For the general case, constructivity implies that, as far as the isomorphism class of an object X in D_s is concerned, the intermediate limits in a limit computation can be removed (Lemma 3.19). Using the notation of the proof of Lemma 3.19, if $X = L_i$ is produced as a limit in the diagram computation, then $X \cong \lim D_{i-1}|_{J_i}$, where $D_{i-1}|_{J_i}$ is a subdiagram of D_0 .

As discussed in Remark 3.20, we can write $X = L_i$ as an equalizer. Let $J'_i = (V, E)$, $L_i \cong \lim D_{i-1}|_{J'_i}$; writing $D = D_{i-1}|_{J'_i}$, L_i is isomorphic to the limit of

$$\prod_{v \in V} D(v) \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \prod_{e \in E} D(t(e)),$$

where $\phi_{D(t(e))} = \pi_{D(t(e))}$ and $\psi_{D(t(e))} = D(e) \circ \pi_{D(s(e))}$.

Since $J'_i \subset I_0$, we have that each $D(v)$ or $D(t(e))$ is \mathbb{A}^1 , \mathbb{A}^2 , or a point; and $D(e)$ are addition, multiplication, constant, projection, or multiplication by a constant. So the above equalizer is of the form

$$\mathbb{A}^{m_1} \begin{array}{c} \xrightarrow{\psi} \\ \xrightarrow{\phi} \end{array} \mathbb{A}^{m_2},$$

and L_i is isomorphic to the zero-locus of $\mathbb{A}^{m_1} \xrightarrow{\phi - \psi} \mathbb{A}^{m_2}$. For the complexity of $\phi - \psi$, observe that $m_1, m_2 \leq 2C$ and that each component of $\phi - \psi$ is a very simple polynomial, which can be produced in two steps.

We now consider the second assertion. This follows directly from the discussion in Remark 3.21. We observe that any morphism $f : X \rightarrow Y$ appearing in D_s but not in D_0 must be a cone map from a limit $X = L_i = \lim D_{i-1}|_{J_i}$ to an object Y appearing in $D_{i-1}|_{J_i}$. By the above construction, we know that X and Y are zero-loci of morphisms whose sources are products of objects in subdiagrams of D_0 . Moreover, constructivity implies that the subdiagram for Y is contained in the subdiagram for X . The morphism f is then the restriction of the projection from the subdiagram for X to the subdiagram for Y . □

The above arguments can also be considered for the category of affine schemes instead of varieties. Let k be a field and let \mathbf{AffSch}_k be the category of affine schemes. Consider the same set \mathcal{A} of morphisms as in the case of affine varieties: $\mathbb{A}^1 \xrightarrow{c} \mathbb{A}^1$, for each $c \in k$, $\mathbb{A}^1 \times \mathbb{A}^1 \xrightarrow{+} \mathbb{A}^1$, $\mathbb{A}^1 \times \mathbb{A}^1 \xrightarrow{\times} \mathbb{A}^1$, $\mathbb{A}^1 \times \mathbb{A}^1 \xrightarrow{\pi_1, \pi_2} \mathbb{A}^1$, $\mathbb{A}^1 \rightarrow \mathbb{A}^0$, and $\mathbb{A}^0 \xrightarrow{c} \mathbb{A}^1$.

One can also consider the category \mathbf{Alg}_k of k -algebras with basic morphisms

$$\begin{aligned} k[x] &\xrightarrow{c} k[x], \quad \text{for each } c \in k, \\ k[x] &\xrightarrow{x \otimes 1 + 1 \otimes x} k[x] \otimes k[x], \\ k[x] &\xrightarrow{x \otimes x} k[x] \otimes k[x], \\ k[x] &\xrightarrow{\text{id} \otimes 1, 1 \otimes \text{id}} k[x] \otimes k[x], \\ k &\xrightarrow{1} k[x], \quad \text{and} \\ k[x] &\xrightarrow{x \mapsto c} k, \quad \text{for each } c \in k. \end{aligned}$$

These correspond to the basic morphisms considered for \mathbf{AffVar}_k under the adjoint equivalence

$$\mathbf{AffSch}_k \begin{array}{c} \xrightarrow{k[\cdot]} \\ \xleftarrow{\text{Spec}} \end{array} \mathbf{Alg}_k^{\text{op}}.$$

Without any modification to the proofs, Theorems 4.2 and 4.5 hold when \mathbf{AffVar}_k is replaced by \mathbf{AffSch}_k or \mathbf{Alg}_k .

4.2. Mixed computations in projective schemes. Let \mathcal{C} be the category \mathbf{Sch}_k of all schemes over a field k . Letting \mathcal{A} consist of the morphisms above as in the affine scheme case, we get a definition of complexity in the category of schemes.

The morphisms in \mathcal{A} are actually enough to produce all projective schemes using mixed computations. For example, in order to make \mathbb{P}^1 , we can produce the following diagram as a subdiagram of a computation and take its colimit:

$$\begin{array}{ccc} \mathbb{A}^1 & & \mathbb{A}^1 \\ & \swarrow \pi_1 & \searrow \pi_2 \\ & Z(xy - 1) \subset \mathbb{A}^2 & \end{array}$$

Note that $Z(xy - 1)$ is isomorphic to $\mathbb{A}^1 - \{0\}$.

PROPOSITION 4.6. *Let $X \subset \mathbb{P}_k^n$ be the zero-scheme of homogeneous polynomials $f_1, \dots, f_m \in k[x_0, \dots, x_n]$. Assume that, for each i , there is an arithmetic circuit of size c_i computing f_i . Then, the categorical complexity of X is in $\mathcal{O}(n^2(c_1 + \dots + c_s))$.*

Proof. We first show how to construct \mathbb{P}^n by a mixed computation and then modify the construction to make the zero-scheme X of f_1, \dots, f_m .

The last step in making \mathbb{P}^n will be to take the colimit of the diagram

$$\begin{array}{ccccccc} A_0 = \mathbb{A}^n & & A_1 = \mathbb{A}^n & & A_2 = \mathbb{A}^n & \dots & A_n = \mathbb{A}^n \\ & \swarrow & \nearrow & \swarrow & \nearrow & & \nearrow \\ & Z_{0,1} & Z_{0,2} & Z_{1,2} & \dots & Z_{i,j} & \dots & Z_{n-1,n} \end{array}$$

There are $n + 1$ copies of \mathbb{A}^n in the first row, which correspond to the standard covering U_0, \dots, U_n of \mathbb{P}^n by affine opens. In the second row, there is an object $Z_{i,j}$ for each pair (i, j) with $i < j$; with each $Z_{i,j}$ corresponding to the

intersections $U_i \cap U_j$ of the affine opens in the chart, each is therefore isomorphic to the complement of a hyperplane in \mathbb{A}^n .

We will construct each Z_{ij} as a subscheme of $\mathbb{A}^n \times \mathbb{A}^n = A_i \times A_j$, considered with coordinates $x_1, \dots, x_n, y_1, \dots, y_n$, defined by the equation

$$y_i x_j - 1 = 0, \tag{5}$$

and, for each $l \in \{1, \dots, i - 1, i + 1, \dots, n\}$, the equation

$$y_i x_l - y_l = 0. \tag{6}$$

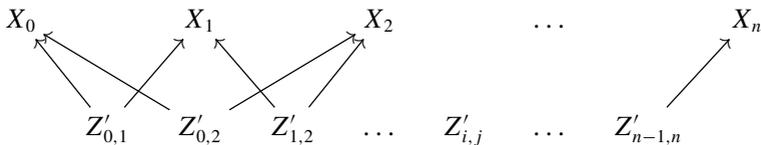
These equations describe the graph of the transition maps

$$(x_1, \dots, x_n) \mapsto \left(\frac{x_1}{x_j}, \frac{x_2}{x_j}, \dots, \frac{x_{i-1}}{x_j}, \frac{1}{x_j}, \frac{x_{i+1}}{x_j}, \dots, \frac{x_{j-1}}{x_j}, \frac{x_{j+1}}{x_j}, \dots, \frac{x_n}{x_j} \right)$$

between the affine opens in the standard covering of \mathbb{P}^n . Here, each affine open U_i , corresponding to points in homogeneous coordinates $[\frac{a_0}{a_i} : \dots : \frac{a_{i-1}}{a_i} : 1 : \frac{a_{i+1}}{a_i} : \dots : \frac{a_n}{a_i}]$ is parameterized by simply omitting the i th variable. The maps $Z_{i,j} \rightarrow A_i$ and $Z_{i,j} \rightarrow A_j$ are the restrictions of the projection maps from $A_i \times A_j$.

To carry out the computation, start with $(n + 1)$ sets of n copies of \mathbb{A}^1 . Make A_i the limit of the i th set of n \mathbb{A}^1 's. Using the procedure described in the proof of Theorem 4.2 (cf. Remark 4.3), construct each $Z_{i,j}$ as the zero-set of equations (5) and (6). Making the projections to the A_i (cf. second part of Remark 3.8), we get the diagram above. At this point, the colimit of this diagram can be taken to produce \mathbb{P}^n .

To make the zero-scheme X of f_1, \dots, f_m in \mathbb{P}^n , we continue in order to make the following diagram.



Here the X_i are isomorphic to $X \cap U_i$ and the $Z'_{i,j}$ are isomorphic to $X \cap U_i \cap U_j$. To make the X_i , de-homogenize f_1, \dots, f_m for each A_i so that we get the equations for the subscheme of A_i that corresponds to $X \cap U_i$. Using the procedure in Theorem 4.2, take the zero-scheme of these de-homogenized equations in each A_i . To make the $Z'_{i,j}$, make the map $X_i \times X_j \rightarrow A_i \times A_j$ and pull back $Z_{i,j}$. Finally, take the colimit of the diagram above to get a scheme isomorphic to X . \square

5. Categorical complexity of morphisms versus circuit complexity of polynomials

Although the motivation behind studying categorical complexity in various categories is a natural problem in its own right, it is still interesting to compare this new notion with pre-existing notions of computational complexity in certain special categories. For a pre-existing notion of complexity, the closest in spirit (in the categories we consider below) is that of arithmetic circuit complexity of polynomials. The goal of this section is to see how closely we can recover arithmetic circuit complexity (defined in Definition 4.4) (or equivalently, straight-line program complexity). We remark that arithmetic circuit complexity is one of the most widely studied frameworks for nonuniform computational complexity theory (see for example, [Bür00]).

We discuss three categories where we can compare the arithmetic circuit complexity of a polynomial $f \in k[x_1, \dots, x_n]$ with the categorical complexity of a morphism diagram. In the first one, we look at the morphism diagram $D_f = (\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1)$ in \mathbf{AffVar}_k and see that its complexity can be very different from the arithmetic circuit complexity of f . The second category, the category of graded pairs of algebras, is an attempt at removing this discrepancy. The third category is the category of modules over $k[x_1, \dots, x_n]$, where we prove concrete comparison results between the arithmetic circuit complexity of f and the categorical complexity of the morphism diagram $k[x_1, \dots, x_n] \xrightarrow{1 \mapsto f} k[x_1, \dots, x_n]$.

5.1. Complexity of polynomial morphisms in \mathbf{AffVar}_k . We consider the category \mathbf{AffVar}_k with the basic morphisms discussed above (4). Given a polynomial

$$f \in k[x_1, \dots, x_n],$$

what is the categorical complexity of the diagram $D_f = (\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1)$ in \mathbf{AffVar}_k ?

First, observe that since we have $\mathbb{A}^0 \xrightarrow{0} \mathbb{A}^1$ as a basic morphism, we can use the pullback of the diagram

$$\mathbb{A}^n \xrightarrow{f} \mathbb{A}^1 \xleftarrow{0} \mathbb{A}^0$$

to obtain the inclusion morphism $V \hookrightarrow \mathbb{A}^n$, where $V = Z(f)$ is the affine subvariety of \mathbb{A}^n defined by f . Thus,

$$c_{\mathbf{AffVar}_k, \mathcal{A}}^{\lim}(Z(f)) \leq c_{\mathbf{AffVar}_k, \mathcal{A}}^{\lim}(D_f) + 2,$$

where \mathcal{A} is the set of basic morphisms given in (4).

Categorical computations produce diagrams up to isomorphism, and categorical complexity is defined for isomorphism classes of diagrams. So, the complexity of f is equal to the complexity of any other $\mathbb{A}^n \xrightarrow{g} \mathbb{A}^1$ where there is an automorphism $\phi : \mathbb{A}^n \rightarrow \mathbb{A}^n$ such that

$$\begin{array}{ccc} \mathbb{A}^n & \xrightarrow{f} & \mathbb{A}^1 \\ \phi \downarrow & & \downarrow \text{id} \\ \mathbb{A}^n & \xrightarrow{g} & \mathbb{A}^1 \end{array}$$

commutes. Therefore, the complexity of D_f in \mathbf{AffVar}_k is invariant under polynomial automorphisms of \mathbb{A}^n (and the complexity of the variety $Z(f)$ is equal to that of $\phi(Z(f))$ for any polynomial automorphism ϕ of \mathbb{A}^n).

For example, let $p \in k[x_1, \dots, x_n]$ be any polynomial, and let

$$g \in k[x_1, \dots, x_n, x_{n+1}], \quad g = x_{n+1} + p(x_1, \dots, x_n).$$

Let $f = \pi_{n+1}$ sending (x_1, \dots, x_{n+1}) to x_{n+1} , and

$$\phi(x_1, \dots, x_{n+1}) = (x_1, \dots, x_n, x_{n+1} + p(x_1, \dots, x_n)).$$

Then, we have the following commuting diagram:

$$\begin{array}{ccc} \mathbb{A}^{n+1} & \xrightarrow{\pi_{n+1}} & \mathbb{A}^1 \\ \uparrow (x_1, \dots, x_n, x_{n+1} + p(x_1, \dots, x_n)) & \nearrow x_{n+1} + p(x_1, \dots, x_n) & \\ \mathbb{A}^{n+1} & & \end{array}$$

So, the diagram $D_{x_{n+1} + p(x_1, \dots, x_n)}$ is isomorphic to $\mathbb{A}^{n+1} \xrightarrow{\pi_{n+1}} \mathbb{A}^1$. So, while the circuit complexity of $x_{n+1} + p(x_1, \dots, x_n)$ can be very high (for example, p could be the permanent, or worse, a generic polynomial), the limit complexity of $D_{x_{n+1} + p(x_1, \dots, x_n)}$ as well as that of $V(x_{n+1} + p(x_1, \dots, x_n))$ is bounded by $O(n)$. This is because, geometrically, the zero-set of $x_{n+1} + p(x_1, \dots, x_n)$ is very simple; it is the graph of $-p$, and is therefore isomorphic to \mathbb{A}^n .

It should also be noted that reductions in circuit complexity do not immediately lead to reductions in the complexity in \mathbf{AffVar}_k . One can construct an arithmetic circuit for the polynomial $p(x_1, \dots, x_n)$ given one for $x_{n+1} + p(x_1, \dots, x_n)$ with only a constant increase in size. However, this does not lead to a diagram computation of $D_{p(x_1, \dots, x_n)}$ from a diagram computation of $D_{x_{n+1} + p(x_1, \dots, x_n)}$ with only a constant increase in cost. Still, we can ask whether polynomials that are believed to be hard to compute in the arithmetic circuit model also have high categorical complexity. For example, we have the following question.

QUESTION 5.1. *Is the limit/mixed complexity of $\mathbb{A}^{n^2} \xrightarrow{\text{perm}_n} \mathbb{A}^1$ polynomially bounded in n ?*

5.2. Complexity of pairs of graded algebras. It is possible that the difference between complexity of D_f in \mathbf{AffVar}_k and the arithmetic circuit complexity of f is caused by the large number of automorphisms of \mathbb{A}^n . The goal of this section is to consider a category where arithmetic circuit complexity is possibly close to categorical complexity, and we can hope to bound the arithmetic circuit complexity of a polynomial f from above by a polynomial function of the categorical complexity of the corresponding morphism (or that of the algebraic variety defined by f), and thus match the result in Part (i) of Theorem 4.5, which contains an inequality in the other direction.

The automorphism group of \mathbb{P}_k^n is much smaller than that of \mathbb{A}_k^n . Indeed, $\text{Aut}(\mathbb{P}_k^n) \cong PGL(n+1, k)$, where $PGL(n+1, k)$ denotes the group of invertible $(n+1) \times (n+1)$ matrices with entries in k modulo multiplication by k^* (see for example, [MFK94, Ch. 0, §5 b])). Thus, the automorphisms of \mathbb{P}_k^n are all linear, and hence the complexity of a projective hypersurface $Z(f) \subset \mathbb{P}_k^N$ should be closer to the arithmetic circuit complexity of its defining polynomial f . However, since the basic objects are not projective, it is better to consider the corresponding setup in the category of graded algebras.

To this end, we could consider the category \mathbf{GrAlg}_k of graded algebras (given a graded k -algebra S , the projective scheme $\text{Proj } S$ will be the corresponding geometric object). The following set of morphisms will be taken to be the basic morphisms.

$$\begin{aligned}
 k[z^n] &\xrightarrow{+} k[x^n, y^n], z^n \mapsto x^n + y^n, \\
 k[z^{2n}] &\xrightarrow{\cdot} k[x^n, y^n], z^{2n} \mapsto x^n y^n, \\
 k[z^n] &\xrightarrow{i_1} k[x^n, y^n], z^n \mapsto x^n, \\
 k[z^n] &\xrightarrow{i_2} k[x^n, y^n], z^n \mapsto y^n, \\
 k[z^n] &\xrightarrow{c \times} k[z^n], z^n \mapsto cz^n, c \in k, \\
 k[z^n] &\xrightarrow{\cdot} k, z^n \mapsto 0.
 \end{aligned} \tag{7}$$

Note that $k[x^n, y^n] \cong k[x^n] \otimes_k k[y^n]$ is the coproduct of $k[x^n]$ and $k[y^n]$ in \mathbf{GrAlg}_k , and hence using colimits we can build graded polynomial rings in any number of variables. Also, the last morphism in list (7) allows us, ‘geometrically speaking’, to build the inclusion of a variety $V \hookrightarrow \mathbb{P}_k^N$ defined by a set of homogeneous polynomials of the same degree. More precisely, in terms of graded algebras, if V is defined by the ideal \mathfrak{a} with generators $g_1, \dots, g_M \in k[z_0, \dots, z_N]$

homogeneous of the same degree, then the morphism $k[z_0, \dots, z_N] \rightarrow k[z_0, \dots, z_N]/\mathfrak{a}$ can be constructed by taking the colimit of the diagram

$$k \xleftarrow{0} k[y_0, \dots, y_M] \xrightarrow{y_i \mapsto g_i} k[z_0, \dots, z_N], \quad (8)$$

noting that the morphism $k[y_0, \dots, y_M] \xrightarrow{0} k$ can be built from the basic morphism $k[z] \rightarrow k, z \mapsto 0$ (cf. equation (7)) by taking colimits.

REMARK 5.2. We should remind the reader here of one unpleasant aspect of the translation between algebra and geometry via taking Proj of graded rings. Unlike the functor

$$\text{Spec} : \mathbf{Alg}_k^{\text{op}} \rightarrow \mathbf{AffSch}_k,$$

the map

$$\text{Proj} : \mathbf{GrAlg}_k^{\text{op}} \rightarrow \mathbf{ProjSch}_k$$

is not a functor since to a morphism of graded k -algebras one can associate only a partial map between the corresponding projective schemes, and for graded k -algebras R and R' , $\text{Proj } R$ and $\text{Proj } R'$ could be isomorphic, without R being isomorphic to R' . Thus, isomorphisms of projective varieties considered in this section do not necessarily correspond exactly to isomorphisms of the corresponding graded algebras. It might be possible to consider as our objects equivalence classes of graded algebras S giving isomorphic $\text{Proj } S$, but we will not attempt to do this here.

Now consider for a tuple of homogeneous polynomial

$$f = (f_0, \dots, f_N) \in k[x_0, \dots, x_n]^{N+1},$$

the diagram

$$k[z_0, \dots, z_N] \xrightarrow{z_i \mapsto f_i} k[x_0, \dots, x_n]$$

denoted also by D_f .

However, the complexity of D_f could still possibly be different from the arithmetic circuit complexity of f . The problem arises if morphism $\mathbb{P}_k^n \rightarrow \mathbb{P}_k^N$ induced by f is an embedding of varieties. More precisely, an embedding $v : \mathbb{P}_k^n = \text{Proj } k[x_0, \dots, x_n] \rightarrow \mathbb{P}_k^N = \text{Proj } k[z_0, \dots, z_N]$. In terms of graded algebras, this would correspond to a morphism $D_f : k[z_0, \dots, z_N] \rightarrow k[x_0, \dots, x_n]$ given by a tuple $f = (f_0, \dots, f_N)$ of homogeneous polynomials of the same degrees, with D_f being the map $z_i \mapsto f_i$. For example, v could be the Veronese embedding of degree d , and in which case $N = \binom{n+d}{d}$, and f

would be the tuple of all monomials in x_0, \dots, x_n of degree d . But the inclusion $\nu(\mathbb{P}_k^n) \hookrightarrow \mathbb{P}_k^N$ also corresponds to a morphism of graded algebras, namely $k[z_0, \dots, z_N] \rightarrow k[z_0, \dots, z_N]/\mathfrak{a}$, where \mathfrak{a} is the homogeneous ideal of $\nu(\mathbb{P}_k^n)$. Suppose that \mathfrak{a} is generated by the homogeneous polynomials g_0, \dots, g_M . In the case where ν is a Veronese embedding, the polynomials g_i can be taken to be a set of quadratic binomials. Now the morphism $k[z_0, \dots, z_N] \rightarrow k[z_0, \dots, z_N]/\mathfrak{a}$ can be constructed by taking the colimit (see (8) above).

If the g_i 's have small colimit complexity compared to the f_i 's, then the second morphism will have smaller colimit complexity. And thus the categorical complexity of the embedding $\nu : \mathbb{P}_k^n \rightarrow \mathbb{P}_k^N$ could be determined by a categorical computation involving only the polynomials g_0, \dots, g_M , and in principle could be much smaller than the arithmetic circuit complexity of f .

To prevent the phenomenon described above, which might cause the categorical complexity of a morphism D_f to diverge from the arithmetic circuit complexity of f , we consider the category whose objects are pairs (X, \mathbb{P}_k^n) , with X a subvariety of \mathbb{P}_k^n . Or rather, we consider the corresponding morphisms of the coordinate algebras. We denote by $\mathbf{GrAlgPairs}_k$ the category whose objects are surjective morphisms $(A \xrightarrow{f} B)$, where A is isomorphic to some polynomial ring $k[x_0, \dots, x_n]$ graded by degree. For example, suppose that $A = k[x]$ and $B = k[x]/(x^2)$ both graded by degree, and $A \xrightarrow{f} B$ the canonical surjection.

In order to define categorical complexity in $\mathbf{GrAlgPairs}_k$, we define the basic morphisms as follows. For each $n \geq 1$, we include in the set of basic morphisms the following set of morphisms (all polynomial rings appearing below are graded by degrees and the degrees of the indeterminates x, y, z, \dots appearing in the morphisms are all equal to 1):

$$\begin{aligned}
 & (k[z^n] \xrightarrow{\sim} k[z^n]) \xrightarrow{+} (k[x^n, y^n] \xrightarrow{\sim} k[x^n, y^n]), \quad z^n \mapsto x^n + y^n, \\
 & (k[z^{2n}] \xrightarrow{\sim} k[z^{2n}]) \xrightarrow{-} (k[x^n, y^n] \xrightarrow{\sim} k[x^n, y^n]), \quad z^{2n} \mapsto x^n y^n, \\
 & (k[z^n] \xrightarrow{\sim} k[z^n]) \xrightarrow{i_1} (k[x^n, y^n] \xrightarrow{\sim} k[x^n, y^n]), \quad z^n \mapsto x^n, \\
 & (k[z^n] \xrightarrow{\sim} k[z^n]) \xrightarrow{i_2} (k[x^n, y^n] \xrightarrow{\sim} k[x^n, y^n]), \quad z^n \mapsto y^n, \\
 & (k[z^n] \xrightarrow{\sim} k[z^n]) \xrightarrow{c \times} (k[z^n] \xrightarrow{\sim} k[z^n]), \quad z^n \mapsto cz^n, \quad c \in k, \\
 & (k[z^n] \xrightarrow{\sim} k[z^n]) \rightarrow (k[x^n] \xrightarrow{(x \mapsto 0)} k), \quad (z^n \mapsto x^n, z^n \mapsto 0).
 \end{aligned} \tag{9}$$

REMARK 5.3. Note that in all but the last morphism in the above list, the pairs occurring as source and target of the morphism consist of isomorphic objects, and the morphism between them is diagonal (that is, of the form (ϕ, ϕ) for some morphism ϕ in the category \mathbf{GrAlg}_k). Only in the last morphism this is not true, and the morphism between the pair in this case is (id, ϕ) , where ϕ is precisely the

morphism used to build inclusion of varieties \mathbf{GrAlg}_k (cf. (8)). This remark will be used in what follows.

Note that we can obtain the morphism

$$(k[z^n] \xrightarrow{(z^n \mapsto 0)} k) \xrightarrow{(z^n \mapsto 0)} (k[x^n] \xrightarrow{\sim} k[x^n])$$

as the colimit diagram:

$$\begin{array}{ccc} (k[t^n] \xrightarrow{\sim} k[t^n]) & \xrightarrow{t^n \mapsto 0, t^n \mapsto 0} & (k[y^n] \xrightarrow{\sim} k[y^n]) \\ \downarrow t^n \mapsto z^n, t^n \mapsto 0 & & \downarrow \\ (k[z^n] \xrightarrow{(z^n \mapsto 0)} k) & \xrightarrow{(z^n \mapsto 0)} & (k[x^n] \xrightarrow{\sim} k[x^n]) \end{array}$$

Given a tuple of homogeneous polynomials

$$f = (f_0, \dots, f_N) \in k[x_0, \dots, x_n]^{N+1},$$

we consider the categorical complexity of the diagram M_f defined as

$$(k[z_0, \dots, z_N] \xrightarrow{\sim} k[z_0, \dots, z_N]) \xrightarrow{M_f} (k[z_0, \dots, z_n] \xrightarrow{\sim} k[x_0, \dots, x_n]), z_i \mapsto f_i.$$

Following the same proof as Theorem 4.2, we have the next proposition.

PROPOSITION 5.4. *Given a tuple of homogeneous polynomials*

$$f = (f_0, \dots, f_N) \in k[x_0, \dots, x_n]^{N+1}$$

having arithmetic circuit complexity M , the colimit complexity of the morphism diagram M_f is in $O(M)$.

We claim also that the phenomenon described previously in the nonembedded setting does not occur in this embedded pair setting. Consider, for example, an embedding $\nu : \mathbb{P}_k^n \rightarrow \mathbb{P}_k^N$ as before, but now consider the embedding of the pair $(\mathbb{P}_k^n, \mathbb{P}_k^n)$ into $(\mathbb{P}_k^N, \mathbb{P}_k^N)$ by the diagonal morphism (ν, ν) . In terms of pairs of graded algebras, this would correspond to a morphism

$$M_f = (D_f, D_f) : (k[z_0, \dots, z_N] \xrightarrow{\sim} k[z_0, \dots, z_N]) \rightarrow (k[x_0, \dots, x_n] \xrightarrow{\sim} k[x_0, \dots, x_n]),$$

given by a tuple $f = (f_0, \dots, f_N)$ of homogeneous polynomials of the same degrees, with D_f being the map $z_i \mapsto f_i$. Suppose also as before that the inclusion $v(\mathbb{P}_k^n) \hookrightarrow \mathbb{P}_k^N$ also corresponds to a morphism of graded algebras, namely $k[z_0, \dots, z_N] \rightarrow k[z_0, \dots, z_N]/\mathfrak{a}$, where \mathfrak{a} is the homogeneous ideal of $v(\mathbb{P}_k^n)$. Suppose that \mathfrak{a} is generated by the homogeneous polynomials g_0, \dots, g_M . However, unlike in the nonembedded case, we cannot construct the morphism

$$(k[z_0, \dots, z_N] \xrightarrow{\sim} k[z_0, \dots, z_N]) \rightarrow (k[z_0, \dots, z_N]/\mathfrak{a} \xrightarrow{\sim} k[z_0, \dots, z_N]/\mathfrak{a})$$

using the polynomials g_i and taking a colimit, since the morphism $k[z] \xrightarrow{0} k$ used to construct the morphism $k[y_0, \dots, y_M] \xrightarrow{0} k$ in the colimit in (8) is available only in the second slot of the pairs and not in the first (see Remark 5.3).

We can now ask whether the converse of Proposition 5.4 is true.

QUESTION 5.5. *For a tuple of homogeneous polynomials $f = (f_0, \dots, f_N) \in k[x_0, \dots, x_n]^{N+1}$ for which the colimit complexity of the morphism \mathbf{M}_f in $\mathbf{GrAlgPairs}_k$ is M , is the arithmetic circuit complexity of f polynomially bounded in M ?*

5.3. Arithmetic circuit and categorical complexity of modules over polynomial rings. We now consider the relationship between arithmetic circuit complexity and categorical complexity in categories of modules. Unlike in the categories of affine varieties and graded algebras (namely, \mathbf{Aff}_k , \mathbf{GrAlg}_k , $\mathbf{GrAlgPairs}_k$) considered in the last two sections, we are able to relate polynomially the arithmetic circuit complexity and categorical complexity of polynomials (and their induced morphisms in the module category) by proving inequalities in both directions. Thus, we are able to prove a stronger relation between circuit complexity and categorical complexity—though the category is perhaps less interesting from the geometric point of view than the ones considered previously.

Let $R = k[x_1, \dots, x_n]$. We consider the category $R\text{-Mod}$ of R -modules. We consider colimit computations in $R\text{-Mod}$ with the following set of basic morphisms.

$$\begin{aligned} R &\xrightarrow{x_i} R, & \text{for each } i = 1, \dots, n, \\ R &\xrightarrow{c} R, & \text{for each } c \in k, \\ R &\xrightarrow{i_1, i_2} R \oplus R, \\ R &\xrightarrow{\Delta} R \oplus R, \end{aligned} \tag{10}$$

$$\begin{aligned} R \oplus R &\xrightarrow{+} R, \\ R &\rightarrow \{0\}. \end{aligned}$$

For a polynomial $f \in k[x_1, \dots, x_n]$, we consider the corresponding morphism $R \xrightarrow{f} R$ that sends 1 to f . Recall that a formula is an arithmetic circuit or straight-line program where past intermediate computations cannot be reused (see [Bür00, §2.2.1]).

PROPOSITION 5.6. *If a polynomial $f \in k[x_1, \dots, x_n]$ is computed by a formula of size s , then the diagram $R \xrightarrow{f} R$ is computed by a colimit computation in $R\text{-Mod}$ with cost bounded by $O(s)$.*

Proof. Without loss of generality, assume that all sum and product gates have fan-in at most two. We will build, for each formula C , a diagram D_C whose colimit will contain $R \xrightarrow{p_C} R$, where p_C is the output polynomial of C . This will be done inductively on the size of C .

Each D_C will be a diagram of the form

$$R \longrightarrow \boxed{} \longrightarrow R$$

whose colimit is R with the morphism from the R on the right to the colimit being id_R and the morphism from the R on the left to the colimit being defined by $1 \mapsto p_C$.

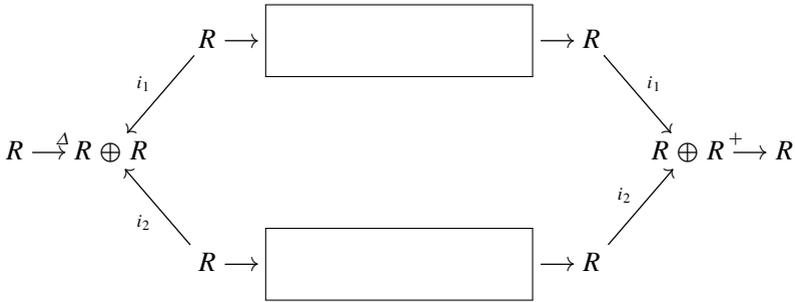
If the output p_C of C is one of the variables x_i , let D_C be the diagram $R \xrightarrow{x_i} R$. If it is just a constant, then D_C is $R \xrightarrow{c} R$.

If the top gate of C is a product gate with C' and C'' as the left and right subcircuits, then we set D_C by chaining together $D_{C'}$ and $D_{C''}$:

$$R \longrightarrow \boxed{} \longrightarrow R \longrightarrow \boxed{} \longrightarrow R .$$

The map from the leftmost R to the colimit is the composition $R \xrightarrow{p_{C'}} R \xrightarrow{p_{C''}} R$, which is $R \xrightarrow{p_{C'} p_{C''}} R$.

If the top gate of C is a sum gate with C' and C'' as the left and right subcircuits, then we define D_C as



where the top and bottom rows are $D_{C'}$ and $D_{C''}$. The colimit of this diagram is again R with the map from the leftmost R to the colimit being $p_C + p_{C'}$. \square

What about a converse? What does the existence of a colimit computation in $R\text{-Mod}$ that produces $R \xrightarrow{1 \mapsto f} R$ say about the complexity of f ?

THEOREM 5.7. *Let $R = k[x_1, \dots, x_n]$. If $R \xrightarrow{f} R$ is computed in a colimit computation with cost c in $R\text{-Mod}$, then there is an arithmetic circuit of size $\text{poly}(c)$ with inputs x_1, \dots, x_n , which computes f .*

Proof. First of all note that R is a unique factorization domain. Hence, given any finite tuple (f_1, \dots, f_q) of elements of R , there exists a unique (up to multiplication by units) greatest common divisor (GCD) of f_1, \dots, f_q . We will need an algorithmic fact related to the computation of GCD in the proof below (cf. (5.3)).

Consider a diagram $D : I \rightarrow R\text{-Mod}$ consisting only of the basic morphisms described above in (10). Assume that we have $\text{colim } D = R$.

For each vertex $v \in I$, we have that $D(v)$ is R , $R \oplus R$ or $\{0\}$. For each v such that $D(v) = R$, let f_v be the image of 1 under the morphism $R \xrightarrow{1 \mapsto f_v} R$ from $D(v)$ to the colimit R . If $D(v) = \{0\}$, then we set $f_v = 0$. If $D(v) = R \oplus R$, then we set two polynomials f_v and $f_{v'}$ so that the map $R \oplus R \rightarrow R$ to the colimit is given by $(1, 0) \mapsto f_v$ and $(0, 1) \mapsto f_{v'}$. We will prove that each f_v is computed by a polynomially sized circuit.

We are considering the f_v 's as unknowns in a system of equations. For each arrow in D , we consider one or two R -linear equations. For an arrow $D(v_1) \rightarrow D(v_2)$ of the form given in the left column, we add the equations in the right column:

$$\begin{array}{ll} R \xrightarrow{x_i} R & f_{v_1} - x_i f_{v_2} \\ R \xrightarrow{c} R & f_{v_1} - c f_{v_2} \end{array}$$

$$\begin{aligned}
 R &\xrightarrow{i_1, i_2} R \oplus R \quad f_{v_1} - f_{v_2} \text{ or } f_{v_1} - f_{v'_2} \\
 R &\xrightarrow{\Delta} R \oplus R \quad f_{v_1} - f_{v_2} - f_{v'_2} \\
 R \oplus R &\xrightarrow{+} R \quad f_{v_1} - f_{v_2} \text{ and } f_{v'_1} - f_{v_2} \\
 R &\rightarrow \{0\} \quad f_{v_1} = 0 \text{ and } f_{v_2} = 0.
 \end{aligned}$$

In this way, we obtain a homogeneous system of $k[x_1, \dots, x_n]$ -linear equations; $A\vec{f} = 0$, $A \in \text{Mat}_{n_2 \times s}(R)$. Tuples $\vec{f} = (f_{v_1}, f_{v_2}, f_{v_3}, \dots, f_{v_s}) \in k[x_1, \dots, x_n]^s$ that satisfy this system of equations correspond to cocones of the diagram D with target R .

Since the colimit of D is R , for any such cocone corresponding to $(f_{v_1}, \dots, f_{v_s})$, there will be a map $(\text{colim } D = R) \xrightarrow{1 \mapsto g} R$ making the diagram containing the new cocone, the colimit cocone and the map $R \xrightarrow{1 \mapsto g} R$ commute. This implies that g divides each f_{v_j} . Since the colimit is the initial cocone, we can find the tuple of polynomials corresponding to the colimit cocone by taking $(\frac{f_{v_1}}{h}, \frac{f_{v_2}}{h}, \dots, \frac{f_{v_s}}{h})$, where $h = \text{gcd}(f_{v_1}, \dots, f_{v_s})$. Thus, assuming the colimit is R , to compute the map from every $D(v)$ to the colimit, it suffices to: (i) find a solution to the above system of equations for D and (ii) divide by $\text{gcd}(f_{v_1}, \dots, f_{v_s})$.

To solve the system $A\vec{f} = 0$, we use Gaussian elimination over the field $k(x_1, \dots, x_n)$. Let R and C be square matrices with entries in $k(x_1, \dots, x_n)$ such that RAC is a diagonal matrix, in the sense that it contains an $r \times r$ minor that is I_r , with $r < s$, and all other entries are 0. Following the steps of Gaussian elimination, there exist circuits (or straight-line programs) with division that produce each entry of R and C in time $\text{poly}(s)$. Also, observe that the entries of R and C have degree at most s . Without loss of generality, assume $RACe_1 = 0$. Then

$$Ce_1 = \left(\frac{p_1}{q_1}, \dots, \frac{p_s}{q_s} \right)^T \in k(x_1, \dots, x_n)^s$$

is a solution to $A\vec{f} = 0$.

Before we proceed further and get this solution into $k[x_1, \dots, x_n]^s$, we make a small digression into algorithms about straight-line programs. We consider the following, which are both proven by the algorithms in Kaltofen's work [Kal188]:

(GCD) Greatest common divisor: Given polynomials $f_1, \dots, f_q \in k[x_1, \dots, x_n]$, which are the output of a circuit of size s' , there is a circuit of size $\text{poly}(s')$ that produces the GCD of f_1, \dots, f_q .

(DE) Denominator extraction: Given a reduced rational function

$$\frac{p(x)}{q(x)} \in k(x_1, \dots, x_n),$$

produced by a circuit with division of size s' , there is a circuit of size $\text{poly}(s')$ that produces q .

In *loc. cit.*, randomized algorithms that produce the output circuits for both DE and GCD are presented. But the circuits that output the gcd and the denominator are themselves not randomized. This will be enough in our nonuniform setting; we only use the existence of the polynomial size circuits that produce the gcd and the denominator and do not use the algorithm that produces the circuits themselves.

Going back to the proof, we made a system of equations $A\vec{f} = 0$ from the diagram of basic morphisms and got a solution of the form

$$Ce_1 = (p_1/q_1, \dots, p_s/q_s)^T \in k(x_1, \dots, x_n)^s$$

and wanted to produce a solution in $k[x_1, \dots, x_n]^s$ instead. To do this, first use Kaltofen's GCD to assume without loss of generality that each $\frac{p_i}{q_i}$ is reduced. Then use Kaltofen's DE to extract the denominators q_i from each fraction. The element $q_1q_2 \dots q_s Ce_1$ is in $k[x_1, \dots, x_n]^s$ and is a solution to $A\vec{f} = 0$. Now divide $q_1q_2 \dots q_s Ce_1$ by the gcd of all of its entries to obtain $(f_{v_1}, \dots, f_{v_s}) \in k[x_1, \dots, x_n]^s$. The polynomials f_1, \dots, f_s correspond to the morphisms from D to the colimit R . Each f_v is produced by a circuit with division, but since the degrees of f_v are polynomially bounded, each such circuit can be turned into a circuit without division using Strassen's method [Str73]. This concludes the proof that for any diagram of basic morphisms with R as a colimit, every colimit cocone morphism from an object in D sends 1 to an f_v , which is computed by a circuit of size polynomial in s .

We now prove the theorem. Let $R \xrightarrow{1 \mapsto f} R$ be a subdiagram of a colimit computation with initial step D_0 . Call the source R_1 and the target R_2 . By Lemma 3.19, there is a subdiagram $D'_0 \subset D_0$ of basic morphisms whose colimit is R_1 ; and (cf. Remark 3.23) there is a subdiagram $D''_0 \subset D_0$, which contains D'_0 , and whose colimit is R_2 , with the induced map $R_1 \rightarrow R_2$ being a map that sends $1 \mapsto f$. This implies, combined with the first part of this proof applied to both D'_0 and D''_0 , that f is the quotient of two polynomials computed by polynomially sized circuits. Hence, by Strassen's method, f is computed by a circuit of size polynomial in the size of D_0 . \square

REMARK 5.8. Note that the set of sequences of polynomials having formula sizes bounded by some quasipolynomial function (that is, a function of the form $n^{\log^c n}$) coincides with the sequence of polynomials having arithmetic circuit complexity bounded by some quasipolynomial function over any field; that is, using the notation in [Bür00], $\text{VQP}_e = \text{VQP}$ [Bür00, Corollary 2.27]. Thus, Proposition 5.6 and Theorem 5.7 together imply that the colimit complexities of a sequence of polynomial morphisms $(k[x_1, \dots, x_n] \xrightarrow{1 \mapsto f_n} k[x_1, \dots, x_n])_{n>0}$ are bounded by a quasipolynomial function of n if and only if the arithmetic circuit complexity of the sequence $(f_n)_{n>0}$ is also bounded by a quasipolynomial function.

6. Functors

In this section, we discuss the interplay of categorical complexity with functors between categories.

6.1. Preservation under functors. Let $F : \mathcal{C} \rightarrow \mathcal{D}$ be a functor. If D is a diagram in \mathcal{C} , the image diagram $F(D)$ is defined in the obvious way. If F preserves finite limits, then for every limit computation (D_0, \dots, D_s) in \mathcal{C} , starting with a set of basic morphisms \mathcal{A} will correspond to a limit computation $(F(D_0), \dots, F(D_s))$ in \mathcal{D} , which starts with basic morphisms in $F(\mathcal{A})$. Therefore, in this case, we have

$$c_{\mathcal{D}, F(\mathcal{A})}^{\text{lim}}(F(D)) \leq c_{\mathcal{C}, \mathcal{A}}^{\text{lim}}(D).$$

We have the analogous statement for colimit computations if F preserves colimits.

Since equivalences preserve limits and colimits, categorical complexity, limit complexity, colimit complexity and mixed complexity are all invariant under equivalences of categories. A more general case is that of adjoint functors. Let $R : \mathcal{C} \rightarrow \mathcal{D}$ be a functor right-adjoint to a functor $L : \mathcal{D} \rightarrow \mathcal{C}$. Then, since right adjoints preserve limits and left adjoints preserve colimits, we have the following.

LEMMA 6.1. *Let $R : \mathcal{C} \rightarrow \mathcal{D}$ and $L : \mathcal{D} \rightarrow \mathcal{C}$ be a pair of adjoint functors. Let \mathcal{A} be a set of basic morphisms in \mathcal{C} and \mathcal{A}' be a set of basic morphisms in \mathcal{D} such that $R(\mathcal{A}) \subset \mathcal{A}'$ and $L(\mathcal{A}') \subset \mathcal{A}$. Then we have inequalities of complexities*

$$\begin{aligned} c_{\mathcal{D}, \mathcal{A}'}^{\text{lim}}(R(D)) &\leq c_{\mathcal{C}, \mathcal{A}}^{\text{lim}}(D), \\ c_{\mathcal{C}, \mathcal{A}}^{\text{colim}}(L(D')) &\leq c_{\mathcal{D}, \mathcal{A}'}^{\text{colim}}(D'), \end{aligned}$$

for every diagram D in \mathcal{C} and D' in \mathcal{D} .

EXAMPLE 6.2. Let \mathbf{Vect}_k be the category of vector spaces over a field k . As the free vector space functor $\mathbf{Fr} : \mathbf{Set} \rightarrow \mathbf{Vect}_k$ is left-adjoint to the forgetful functor $\mathbf{Fo} : \mathbf{Vect}_k \rightarrow \mathbf{Set}$, we have that \mathbf{Fr} preserves colimits. In particular, the colimit complexity of a vector space is bounded above by its dimension.

6.2. Complexity of functors. We discussed in Section 6.1 above, how certain functors preserve limits or colimits, and does also corresponding notions of complexities as well. In this section, we will study complexities of general functors. The first step is to define the complexity of a functor between categories.

DEFINITION 6.3. A *complexity function* on a category \mathcal{C} is a function that takes (finite) diagrams of \mathcal{C} to $\mathbb{N} \cup \{\infty\}$.

EXAMPLE 6.4. For example, if \mathcal{A} is a set of morphisms of \mathcal{C} (assumed to have unit cost), then $c_{\mathcal{C},\mathcal{A}}^{\lim}$, $c_{\mathcal{C},\mathcal{A}}^{\text{colim}}$, $c_{\mathcal{C},\mathcal{A}}^{\text{mixed}}$ are all examples of complexity functions on \mathcal{C} .

DEFINITION 6.5 (Complexity of functors). Let \mathcal{C}, \mathcal{D} be two categories with complexity functions, ϕ, ψ , and let $F : \mathcal{C} \rightarrow \mathcal{D}$ be a functor. We define the complexity, $C_{\phi,\psi}(F) : \mathbb{N} \rightarrow \mathbb{N}$ by

$$C_{\phi,\psi}(F)(n) = \sup \{ \psi(F(D)) \mid I \text{ a finite directed graph, } D \in \mathcal{C}^I, \phi(D) \leq n \}.$$

If F is an endofunctor (that is, $\mathcal{C} = \mathcal{D}$) and $\phi = \psi$, then we will denote $C_{\phi,\psi}(F)$ just by $C_{\phi}(F)$.

One key example of functor complexity comes from *image functors*. In order to define the image functor, we first need to recall a few relevant notions from category theory.

6.3. Monomorphisms, slice categories, and the image functor. Let \mathcal{C} be any category. Recall the following standard definitions.

DEFINITION 6.6 (Monomorphisms). A morphism $f : A \rightarrow B$ in a category \mathcal{C} is called a *monomorphism* if for all pairs of morphisms $g, h : C \rightarrow A$, $f \circ g = f \circ h \implies g = h$.

DEFINITION 6.7 (Slice category). Let A be an object of \mathcal{C} . The *slice category*, \mathcal{C}/A , is the category whose objects are morphisms $B \rightarrow A$, and whose morphisms

are commutative diagrams:

$$\begin{array}{ccc} B & \xrightarrow{f} & C \\ & \searrow & \swarrow \\ & A & \end{array}$$

which compose in the obvious way.

The full subcategory of \mathcal{C}/A consisting of monomorphisms $f : C \rightarrow A$ is denoted by $\mathbf{Sub}(A)$ (subobjects of A).

DEFINITION 6.8. We say that a category \mathcal{C} has images if for every object A of \mathcal{C} , the inclusion functor $i_{\mathcal{C},A} : \mathbf{Sub}(A) \rightarrow \mathcal{C}/A$ has a left adjoint $\text{im}_{\mathcal{C},A}$ (cf. [Joh02, Lemma 1.3.1]).

$$\begin{array}{ccc} & \text{im}_{\mathcal{C},A} & \\ & \curvearrowright & \\ \mathbf{Sub}(A) & \xrightarrow{i_{\mathcal{C},A}} & \mathcal{C}/A \end{array}$$

REMARK 6.9. Note that the adjointness $\text{im}_{\mathcal{C},A} \dashv i_{\mathcal{C},A}$ is equivalent to saying that given a monomorphism $f : B \rightarrow A$ and a morphism $g : C \rightarrow A$ in \mathcal{C}/A , there is a natural isomorphism

$$\mathcal{C}/A(g, i_{\mathcal{C},A}(f)) \cong \mathbf{Sub}(A)(\text{im}_{\mathcal{C},A}(g), f),$$

where the isomorphism takes $\theta \in \mathcal{C}/A(g, i_{\mathcal{C},A}(f))$ given by the diagram

$$\begin{array}{ccc} C & \xrightarrow{\theta} & B \\ & \searrow g & \swarrow f \\ & A & \end{array}$$

to the element in $\mathbf{Sub}(A)(\text{im}_{\mathcal{C}}(g), f)$ described by the diagram

$$\begin{array}{ccc} \text{dom}(\text{im}_{\mathcal{C}/A}(g)) & \xrightarrow{\text{im}_{\mathcal{C}/A}(\theta)} & B \\ & \searrow \text{im}_{\mathcal{C}/A}(g) & \swarrow f \\ & A & \end{array}$$

In particular, taking $f = \text{im}_{\mathcal{C},A}(g)$, the image under the inverse of the isomorphism defined above of the element $1_f \in \mathbf{Sub}(A)$ induces a morphism $\varepsilon(g) : B \rightarrow \text{dom}(\text{im}_{\mathcal{C}/A}(f))$, making the following diagram commute:

$$\begin{array}{ccc}
 B & \xrightarrow{\varepsilon(g)} & \text{dom}(\text{im}_{\mathcal{C}/A}(f)) \\
 & \searrow g & \swarrow \text{im}_{\mathcal{C}/A}(g) \\
 & & A
 \end{array}$$

We now extend the image functor from the slice categories \mathcal{C}/A to the diagram category $\mathcal{C}^{\rightarrow\bullet}$.

PROPOSITION 6.10. *Suppose that \mathcal{C} is a category that has pullbacks and images. Then in the diagram category $\mathcal{C}^{\rightarrow\bullet}$, letting $\text{Mon}_{\mathcal{C}}$ denote the full subcategory of monomorphisms, and $i_{\mathcal{C}} : \text{Mon}_{\mathcal{C}} \rightarrow \mathcal{C}^{\rightarrow\bullet}$ the inclusion functor, there exists a left adjoint $\text{im}_{\mathcal{C}}$ to $i_{\mathcal{C}}$.*

Proof. The functor $\text{im}_{\mathcal{C}}$ is defined as follows. For an object $f : C \rightarrow A$ of $\mathcal{C}^{\rightarrow\bullet}$, we set $\text{im}_{\mathcal{C}}(f) = \text{im}_{\mathcal{C}/A}(f)$. For a morphism, $\theta = (\theta_0, \theta_1)$ of $\mathcal{C}^{\rightarrow\bullet}$, given by the following commutative diagram,

$$\begin{array}{ccc}
 C & \xrightarrow{\theta_1} & B \\
 \downarrow f & & \downarrow g \\
 A & \xrightarrow{\theta_0} & D,
 \end{array}$$

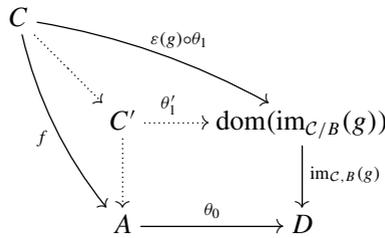
consider the pullback diagram

$$\begin{array}{ccc}
 C' & \xrightarrow{\theta'_1} & \text{dom}(\text{im}_{\mathcal{C},B}(g)) \\
 \downarrow & & \downarrow \text{im}_{\mathcal{C},B}(g) \\
 A & \xrightarrow{\theta_0} & D
 \end{array}$$

Since, the morphism $\text{im}_{\mathcal{C},B}(g) : \text{dom}(\text{im}_{\mathcal{C},B}(g)) \rightarrow D$ is a monomorphism, and the pullback of a monomorphism is again a monomorphism, the left vertical arrow is a monomorphism. Moreover, from the universal property of the pullback applied to the diagram

$$\begin{array}{ccc}
 C & & \\
 \searrow f & \searrow \varepsilon(g) \circ \theta_1 & \text{dom}(\text{im}_{\mathcal{C}/B}(g)) \\
 & & \downarrow \text{im}_{\mathcal{C},B}(g) \\
 A & \xrightarrow{\theta_0} & D
 \end{array}$$

we obtain the commuting diagram

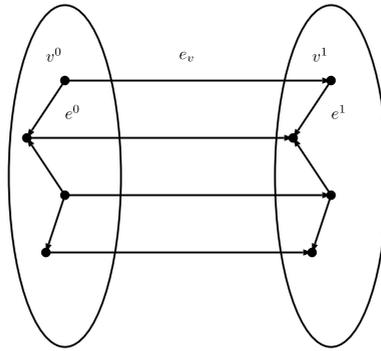


where the dotted diagonal arrow is isomorphic to $\varepsilon(f) : C \rightarrow \text{dom}(\text{im}_{C/A}(f))$ (using the fact that the left vertical arrow is a monomorphism). We set $\text{im}_C(\theta) = \theta' = (\theta'_1, \theta_0) \in \text{Mon}_C(\text{im}_C(f), \text{im}_C(g))$.

To check im_C is left-adjoint to i_C is an easy (if tedious) exercise. □

EXAMPLE 6.11 (Examples of categories with images). Most of the categories introduced before have images (and also pullbacks). It is easy to verify that the categories **Vect_k**, **Set**, **Top**, **SL**, **SA** have images as well as pullbacks. The fact that the last two categories (see Notation 2.3, Part D for their definitions) have images is a consequence of the fact that the images of semilinear (respectively, semialgebraic) sets are closed under taking images under affine (respectively, polynomial) maps.

6.4. Complexity of the image functor. The P versus NP question in computational complexity (say, in the real B–S–S model of computation) is fundamentally about comparing the complexities of sequences of semialgebraic sets, which belong to an ‘easy’ class (that is, the B–S–S complexity class $P_{\mathbb{R}}$), with the complexities of sequences obtained by taking images under certain projections of sequences belonging to the ‘easy’ class (by taking the images under projections of sequences in the class $P_{\mathbb{R}}$, one obtains the B–S–S complexity class $NP_{\mathbb{R}}$). A formal definition of $P_{\mathbb{R}}$, $NP_{\mathbb{R}}$ (and indeed of the whole polynomial hierarchy $PH_{\mathbb{R}}$) can be found in [BZ10, §1.2.1] and will not be repeated here. (We only note that existential quantifiers used in the definition of $NP_{\mathbb{R}}$ in [BZ10] geometrically correspond to taking the image under projection, or more precisely, elimination of existential quantifiers geometrically corresponds to taking the image under a linear projection map.) By replacing the notion of B–S–S complexity by other nonuniform notions of complexity, one can extend these definitions to the nonuniform case as well (see [Isi19] and [BP18]). Finally, what is perhaps closest to the point of view of this paper, in [Bas15], a sheaf theoretic analog of the classes $P_{\mathbb{R}}$ and $NP_{\mathbb{R}}$ is defined. The elements of these classes

Figure 9. The directed graph I' .

are no longer sequences of semialgebraic sets, but rather of semialgebraically constructible sheaves, and the role of taking images under projection maps is replaced by the direct image functor. A ‘P versus NP’ conjecture in this general setting is formulated in [Bas15] (Conjecture 3.83). It should be clear from the above discussion that the P versus NP question is intimately related to studying how badly the complexity of certain objects (say semialgebraic sets) blows up (that is, whether the blow-up is polynomially bounded or not) after taking the image under certain tame maps. The notion of ‘complexity’ of a sequence of sets in the formulations described above is different in each case, but is related for example to the number of steps taken by a B–S–S machine, or the size of an arithmetic circuit, required for testing membership in the corresponding sequence of sets.

With the discussion above as background, we proceed to define an analog of the P versus NP question in the categorical setting. The main idea is to replace ‘taking images under projections’ by the image functor in the category under question (assuming it has an image functor) and replacing the notion of ‘complexity’ by categorical complexity. The categorical analog turns out to be simpler, coordinate independent, and also free of taking sequences.

In this paper, we do not attempt to prove any formal relationships between the categorical and standard versions of the P versus NP question, leaving such questions for future investigations. However, we hope the discussion in the previous paragraph makes clear the analogy between the standard ‘P versus NP’ questions, with the categorical analog.

First, observe that for any category \mathcal{C} , a diagram $D : I = (V, E, s, t) \rightarrow \mathcal{C}^{\bullet \rightarrow \bullet}$ in the diagram category of $\mathcal{C}^{\bullet \rightarrow \bullet}$ induces a diagram $D' : I' = (V', E', s', t') \rightarrow \mathcal{C}$ as follows (cf. Figure 9).

Here $V' = V^0 \cup V^1$, where $V^i = \{v^i \mid v \in V\}$ is a copy of V for $i = 0, 1$, and $E' = E^0 \cup E^1 \cup \bigcup_{v \in V} \{e_v\}$, where $E^i = \{e^i \mid e \in E\}$ is a copy of E for $i = 0, 1$, and

$$\left. \begin{aligned} s'(e^i) &= s(e)^i \\ t'(e^i) &= t(e)^i \end{aligned} \right\} \text{ for } e \in E, i = 0, 1,$$

$$\left. \begin{aligned} s'(e_v) &= v^0, \\ t'(e_v) &= v^1. \end{aligned} \right\} \text{ for } v \in V,$$

and

$$\begin{aligned} D'(e^0) &= D(s(e)) \\ D'(e^1) &= D(t(e)) \\ D'(e_v) &= D(v), v \in V, \end{aligned}$$

remembering that for each vertex $v \in V$, $D(v)$ is a morphism of \mathcal{C} .

Using the notation introduced above, we have the following definition.

DEFINITION 6.12 (Image complexity). If \mathcal{A} is a set of morphisms in \mathcal{C} , we will denote for $*$ = lim, colim, mixed for any diagram D of $\mathcal{C}^{\bullet \rightarrow \bullet}$ (respectively, $\text{Mon}_{\mathcal{C}}$); we define the image complexity $c_{\mathcal{C}^{\bullet \rightarrow \bullet}, \mathcal{A}}^*(D)$ (respectively, $c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*(D)$) of the diagram D by

$$c_{\mathcal{C}^{\bullet \rightarrow \bullet}, \mathcal{A}}^*(D) = c_{\mathcal{C}, \mathcal{A}}^*(D')$$

(respectively, $c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*(D) = c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*(D')$), where the right-hand side is the (limit, colimit or mixed) complexity of the diagram D' in the category \mathcal{C} (cf. Definition 3.13), with \mathcal{A} as the set of basic morphisms.

Now suppose that a category \mathcal{C} has images and \mathcal{A} is a set of morphisms of \mathcal{C} . Recall the definition of the complexity of a functor between two categories with complexity functions (cf. Definition 6.5). Applying Definition 6.5 to the functor

$$\text{im}_{\mathcal{C}} : \mathcal{C}^{\bullet \rightarrow \bullet} \rightarrow \text{Mon}_{\mathcal{C}},$$

with complexity functions $c_{\mathcal{C}^{\bullet \rightarrow \bullet}, \mathcal{A}}^*$ and $c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*$ (with $*$ = lim, colim, mixed), on the source and target of the functor $\text{im}_{\mathcal{C}}$, we obtain the function

$$C_{c_{\mathcal{C}^{\bullet \rightarrow \bullet}, \mathcal{A}}^*, c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*}(\text{im}_{\mathcal{C}})(n).$$

DEFINITION 6.13 (Complexity of the image functor). With $*$ = lim, colim, mixed, we will denote

$$\text{IFC}_{\mathcal{C}, \mathcal{A}}^*(n) = C_{c_{\mathcal{C}^{\bullet \rightarrow \bullet}, \mathcal{A}}^*, c_{\text{Mon}_{\mathcal{C}}, \mathcal{A}}^*}(\text{im}_{\mathcal{C}})(n) \quad (11)$$

and call $\text{IFC}_{\mathcal{C}, \mathcal{A}}^*$ the **-complexity of the image functor* of the category \mathcal{C} (with respect to the basic set of morphisms \mathcal{A}).

Now for any category \mathcal{C} that has images, and for any subset \mathcal{A} of morphisms of \mathcal{C} , and $*$ = lim, colim, mixed, we can ask the following question.

QUESTION 6.14. *Is the function*

$$\text{IFC}_{\mathcal{C}, \mathcal{A}}^*$$

bounded by some polynomial?

6.5. The image functor in the categories **SL and **SA**.** First, observe that the class of semilinear (respectively, semialgebraic) sets is closed under taking images of affine (respectively, polynomial) maps. This is a consequence of Fourier–Motzkin elimination (respectively, Tarski–Seidenberg principle). However, the answer to Question 6.14 may depend on the type of categorical complexity that is being considered.

We consider the cases of limit and mixed complexity, starting with the limit complexity in the categories **SL** and **SA**. We prove the answer to Question 6.14 in **SL** and **SA** for limit complexity with the choice of the basic morphisms defined below. This should not be too much of a surprise since the power of limit computation with these choices of basic morphisms is quite limited. Only closed convex polyhedral subsets of \mathbb{R}^n can be constructed in **SL** (respectively, basic closed semialgebraic sets) using limit computations. (A basic closed semialgebraic set is a semialgebraic set defined by a conjunction of a finite number of weak polynomial inequalities of the form $P \geq 0$.) Moreover, in the case of the category **SL**, we prove a polynomial upper bound on the number of facets of an object in terms of limit complexity (Lemma 6.15 below). The negative answer to Question 6.14 in this case can be deduced by exhibiting a sequence of morphisms of polynomially bounded complexity such that the number of facets of the image grows super-polynomially. We exhibit such a sequence using the well-known properties of cyclic polytopes. In the case of the category **SA**, the situation is even simpler since it is well known that images under polynomial maps (in fact, projections along a coordinate) of a basic closed semialgebraic set need not be basic closed. We exhibit a simple example of this phenomenon.

6.5.1. Limit complexity of the image functor in **SL.** Recall that the objects of the category **SL** are (embedded) semilinear sets and morphisms between such sets, which are restrictions of affine mappings. Let \mathcal{A} consist of the scalar

multiplication morphisms $\mathbb{R} \xrightarrow{c} \mathbb{R}$ for each $c \in \mathbb{R}$, the addition morphism $\mathbb{R}^2 \xrightarrow{+} \mathbb{R}$, and morphisms $[0, \infty) \hookrightarrow \mathbb{R}, \mathbb{R} \rightarrow 0$. It is easy to see that using limit computations, one can produce every morphism $f : (A \subset \mathbb{R}^m) \rightarrow (B \subset \mathbb{R}^n)$, where A, B are closed polyhedral subsets of \mathbb{R}^m and \mathbb{R}^n , respectively.

LEMMA 6.15. *The number of facets of each object A is bounded from above by $(c_{\text{SL},A}^{\text{lim}}(A))^2$.*

Proof. Let (D_0, \dots, D_s) be the sequence of diagrams of a limit computation of A . We prove by an induction on s the following statement from which the lemma will follow.

For each vertex v of D_s , let $h_s(v) = \text{card}(H_s(v))$, where $H_s(v)$ is the set of vertices of D_s which was introduced by taking a limit, and which has v in its cone. We follow the convention that if v is a vertex of D_0 , then $H_s(v) = \{v\}$. Because of the constructivity assumption, $h_s(v)$ is well defined.

We claim that the number of facets of A is bounded by $\sum_{v \in D_s} h_s(v)$. We prove the claim by an induction on s . The claim is clearly true if $s = 0$ since the objects that are domains or codomains of the basic morphisms have at most one facet, and $h_s(v) = 1$ for each vertex of D_0 . Now, assume that the claim holds for all $s' < s$.

For each vertex v of D_{s-1} , let $A_v = D_{s-1}(v)$. Let V_{s-1} be the set of vertices of D_{s-1} , which do not belong to the limit cone of any vertex v of D_{s-1} not equal to itself. Then, A is the intersection of the product of the polytopes $A_v, v \in V_{s-1}$ with an affine subspace. This implies that the number of facets of A is bounded by the sum of the number of facets of $A_v, v \in V_{s-1}$. Now use the induction hypothesis to finish the proof of the claim.

It is clear that the lemma follows from the claim since for each vertex v of D_s , $h_s(v)$ is bounded by the total number of vertices of D_s , which itself is bounded by the cost of the diagram computation (D_0, \dots, D_s) . □

Consider now the following example. Denote by $\theta_m(t) = (t, t^2, \dots, t^{2m}) \subset \mathbb{R}^{2m}$ the moment curve in \mathbb{R}^{2m} . The convex hull of n distinct points on the moment curve is called a cyclic polytope, $\text{Cyl}(n, m)$. For $n > 2m$, the number of facets of $\text{Cyl}(n, m)$ is given by [Gal63, Theorem 4]

$$\binom{n - m}{m} + \binom{n - m - 1}{m - 1}.$$

In particular, the number of facets of $\text{Cyl}(4m + 1, 2m)$ equals $4m$. It is easy to derive that the limit complexity of $\text{Cyl}(4m + 1, 2m)$ is bounded by $O(m^2)$. Now let $\pi_m : \mathbb{R}^{4m} \rightarrow \mathbb{R}^{2m}$ be the projection on the first $2m$ coordinates. Clearly, $\pi_m(\text{Cyl}(4m + 1, 2m)) = \text{Cyl}(4m + 1, m)$, and the number of facets

of $\text{Cyl}(4m + 1, m)$ equals

$$\binom{3m + 1}{m} + \binom{4m}{m - 1},$$

which is clearly exponential in m . It follows from Lemma 6.15 that the limit complexity of $\pi_m(\text{Cyl}(4m + 1, 2m))$ is exponentially large in m .

Thus, the object $\pi_m : \text{Cyl}(4m + 1, 2m) \rightarrow \mathbb{R}^{2m}$ of the category $\mathbf{SL}^{\bullet \rightarrow \bullet}$ has limit complexity polynomially bounded in m ; the limit complexity of $(im_{\mathbf{SL}})(\pi_m)$ is exponentially large in m . It follows from this example the next proposition.

PROPOSITION 6.16. *The function $\text{IFC}_{\mathbf{SL}, \mathcal{A}}^{\text{lim}}$ is not polynomially bounded (where \mathcal{A} consists of the scalar multiplication morphisms $\mathbb{R} \xrightarrow{c} \mathbb{R}$ for each $c \in \mathbb{R}$, the addition morphism $\mathbb{R}^2 \xrightarrow{+} \mathbb{R}$, and morphisms $[0, \infty) \hookrightarrow \mathbb{R}, \mathbb{R} \rightarrow 0$).*

6.5.2. Limit complexity of the image functor in \mathbf{SA} . The case of the category \mathbf{SA} with respect to limit complexity is simpler. Recall that the objects of the category \mathbf{SA} are (embedded) semialgebraic sets and morphisms between such sets, which are restrictions of polynomial mappings. Let \mathcal{A} consist of the scalar multiplication morphisms $\mathbb{R} \xrightarrow{c} \mathbb{R}$ for each $c \in \mathbb{R}$, the addition morphism $\mathbb{R}^2 \xrightarrow{+} \mathbb{R}$, the multiplication morphism $\mathbb{R}^2 \xrightarrow{\cdot} \mathbb{R}$, and morphisms $[0, \infty) \hookrightarrow \mathbb{R}, \mathbb{R} \rightarrow 0$. It is not difficult to see that the objects of \mathbf{SA} that can be constructed using a limit computation are exactly the basic closed semialgebraic sets. On the other hand, it is well known that the image under polynomial maps (for example, projections along some coordinates) of a basic closed semialgebraic set need not be a basic closed semialgebraic set. For example, consider the real variety V defined by

$$(X_1 - X_3^2)(X_2 - X_4^2) = 0.$$

Denoting by $\pi : \mathbb{R}^4 \rightarrow \mathbb{R}^2$ the projection to X_1, X_2 -coordinates, $\pi(V) = \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1 \geq 0 \vee x_2 \geq 0\}$, which is not a basic closed semialgebraic set (as observed by Lojasiewicz; see [AR94, page 466]), and hence $\pi(V)$ has infinite limit complexity.

The next proposition follows.

PROPOSITION 6.17. *The function $\text{IFC}_{\mathbf{SA}, \mathcal{A}}^{\text{lim}}$ is not polynomially bounded (where \mathcal{A} consists of the scalar multiplication morphisms $\mathbb{R} \xrightarrow{c} \mathbb{R}$ for each $c \in \mathbb{R}$, the addition morphism $\mathbb{R}^2 \xrightarrow{+} \mathbb{R}$, the multiplication morphism $\mathbb{R}^2 \xrightarrow{\cdot} \mathbb{R}$, and morphisms $[0, \infty) \hookrightarrow \mathbb{R}, \mathbb{R} \rightarrow 0$).*

6.5.3. *Complexity of the image functor with respect to mixed complexity in **SL** and **SA**.* Mixed computation is much more powerful than limit computation, and we are unable to resolve the following question, which we posit as the categorical analog of the P versus NP question in the categories **SL** and **SA** (refer to the discussion in the beginning of this section).

QUESTION 6.18. *Are the functions $\text{IFC}_{\mathbf{SL}, \mathcal{A}}^{\text{mixed}}$, $\text{IFC}_{\mathbf{SA}, \mathcal{A}}^{\text{mixed}}$ polynomially bounded (with the set of morphisms \mathcal{A} being the same as in Propositions 6.16 and 6.17, respectively)?*

We also make the following conjecture.

CONJECTURE 6.19. *The function $\text{IFC}_{\mathbf{SA}, \mathcal{A}}^{\text{mixed}}$ is bounded singly exponentially.*

This should be thought of as the categorical analog of the existence of algorithms with singly exponential algorithms for the elimination of one block of existential quantifiers in the first-order theory of the reals (see for example, [BPR06]).

6.6. The image functor in the category $k[x_1, \dots, x_n]$ -Mod and colimit complexity. We have considered complexity of the image functor with respect to limit and mixed complexity in several categories. We now consider the image functor from the point of view of colimit complexity in the category R -Mod, where $R = k[x_1, \dots, x_n]$, with the set basic morphisms defined in (10).

We describe a method for writing a colimit computation to compute the image of morphism of modules. A careful analysis of the part of the proof of Proposition 6.20, which makes use of the existence of Gröbner basis of modules, would give an upper bound for the complexity of the image functor in R -Mod. With a naive analysis, we can only obtain an unnecessarily high bound (doubly exponential), so we leave the complexity analysis (that is the explicit dependence on the parameter s) out of the following statement.

PROPOSITION 6.20. *Let $M \xrightarrow{\varphi} N$ be a morphism diagram in R -Mod, computed by a colimit computation of size s . Then, there is a colimit computation that computes $\text{im}(\varphi) \rightarrow N$ of size bounded by some function of s .*

Proof. By Lemma 3.19 and Remark 3.23, the colimit computation that produces $M \xrightarrow{\varphi} N$ can be simplified to produce (only) $M \xrightarrow{\varphi} N$ in two colimit steps. The first one is taking the colimit of diagram D_M of basic morphisms, which produces M . The second one is the colimit of a larger diagram $D_N \supset D_M$ together with M

and all the cocone morphisms from the objects in D_M to M . Replacing colimits with coequalizers and coproducts, we have that M is isomorphic to the quotient

$$\bigoplus_{\rho \in s(D_M)} R^{j_\rho} \xrightarrow{A_M} \bigoplus_{\gamma \in v(D_M)} R^{j_\gamma} \longrightarrow M,$$

where γ runs over all the vertices in D_M and ρ runs over all sources of arrows in D_M ; and j_ρ and j_γ are 0, 1, or 2, based on whether the corresponding basic object is $\{0\}$, R , or R^2 . (Note that when using Remark 3.23, we are implicitly using the fact that in the category $R\text{-Mod}$, the coequalizer of two morphisms $\phi, \psi : A \rightarrow B$ is isomorphic to the cokernel of $\phi - \psi$.)

Similarly we can write N as a quotient,

$$\bigoplus_{\rho \in s(D_N)} R^{j_\rho} \oplus \bigoplus_{v \in v(D_M)} R^{j_v} \longrightarrow \bigoplus_{\gamma \in v(D_N)} R^{j_\gamma} \oplus M \longrightarrow N. \quad (12)$$

Since M is the colimit of D_M , we can remove the extra sum on the left and the M summand in the middle. So N is the quotient:

$$\bigoplus_{\rho \in s(D_N)} R^{j_\rho} \xrightarrow{A} \bigoplus_{\gamma \in v(D_N)} R^{j_\gamma} \longrightarrow N$$

Combining the direct sums, we get a commuting diagram with exact rows:

$$\begin{array}{ccccccc} R^{m_2} & \xrightarrow{A_M} & R^{m_1} & \longrightarrow & M & \longrightarrow & 0 \\ \downarrow i_2 & & \downarrow i_1 & & \downarrow \varphi & & \\ R^{n_2} & \xrightarrow{A} & R^{n_1} & \longrightarrow & N & \longrightarrow & 0, \end{array}$$

where the maps i_1 and i_2 are the inclusion of the first m_1 and m_2 coordinate spaces, respectively. Written this way, the map φ from M to N is induced by the inclusion of the generators of M into the generators of N .

To construct the image $\text{im}(\varphi)$, we need to find the relations among the images of the generators of M , that is, the first m_1 generators of N . It is possible to use Gröbner basis methods to obtain a presentation of $\text{im}(\varphi)$, that is, obtain a $B = m_1 \times u$ matrix with entries in R such that

$$\begin{array}{ccccccc} R^u & \xrightarrow{B} & R^{m_1} & \longrightarrow & \text{im } \varphi & \longrightarrow & 0 \\ & & \downarrow i_1 & & \downarrow \iota & & \\ R^{n_2} & \longrightarrow & R^{m_2} & \longrightarrow & N & \longrightarrow & 0 \end{array}$$

(see for instance [KR00, Proposition 3.3.1, Part (b)]). The size of the matrix B and the degrees of the polynomials appearing in it are bounded by some explicit (possibly doubly exponential or worse) function of the degrees of the polynomials appearing in the matrix A coming from the analysis of algorithms for computing Gröbner basis of submodules of a free R -module given a set of generators. The important point for us is that these degrees and the size of B are thus bounded by some function of s (the colimit complexity of the given morphism $M \xrightarrow{\varphi} N$).

It is easy to see that using the basic morphisms and colimits, one can realize using a colimit computation any R mod homomorphisms between two free R -modules of finite rank, with cost depending on the degrees of the polynomials appearing in the matrix corresponding to the homomorphism. Thus, we can obtain the homomorphism $\mathbb{R}^u \xrightarrow{B} \mathbb{R}^{m_1}$ using colimit computation of size bounded by some function of s .

We can now obtain the R -module $\text{im}(\varphi)$ using a colimit computation by taking the colimit of the following diagram:

$$0 \leftarrow R^u \xrightarrow{B} R^{m_1}.$$

The inclusion homomorphism $\iota : \text{im}(\varphi) \hookrightarrow N$ can be computed by constructing N again by taking the basic object R 's in the computation of $\text{im}(\varphi)$ corresponding to the generators of M and constructing any remaining generators and relations for N from (12). \square

7. Open problems and future directions

In this section, we list some open problems suggested by the contents of this paper.

- (1) Resolve Questions 5.1 and 5.5.
- (2) Prove singly exponential upper bound on the mixed complexity of the image functor in the semialgebraic category. This would correspond to the singly exponential complexity algorithms for computing the image of semialgebraic sets under polynomial maps, which follow from the critical point method in algorithmic real algebraic geometry (see for example, [BPR06, Ch. 14]).
- (3) Prove singly exponential upper bound on the categorical complexity of the image functor in the category $R\text{-Mod}$ (cf. Proposition 6.20).
- (4) One important research direction in quantitative semialgebraic geometry has been to prove tight bounds on the topological complexity of

semialgebraic sets in terms of various parameters bounding the sizes of the formulas defining them—such as the number of polynomial inequalities, their degrees or the number of monomials in their support, or even the additive complexity (see for example, [BPR05]). One could ask for similar results in the categorical setting. In the language introduced in this paper, this would mean investigating the *complexity of the homology functor* $H : \mathbf{SA} \rightarrow \mathbb{Z}\text{-Mod}$.

- (5) Investigate the relationship between the B–S–S complexity classes over the real numbers and the (mixed)-categorical complexity of objects in \mathbf{SA} .
- (6) Prove or disprove that the image functor in various categories has polynomially bounded (mixed) complexity. This is the categorical version of the P versus NP question in ordinary complexity theory (see Questions 6.14 and 6.18).
- (7) Note that the image functor im_C has a right adjoint i_C whose complexity is clearly bounded polynomially (in fact, $C_{\text{i}_C, \mathcal{A}}(n) = n$ for any set \mathcal{A} of basic morphisms in \mathcal{C}). More generally, functors often come in adjoint pairs, and often it is easy to show that one of them has polynomially bounded complexity, while the other is conjecturally hard (that is, not polynomially bounded). More examples are given in the category of constructible sheaves in [Bas15]. As already noted, left-adjoint functors preserve colimits and thus are well behaved with respect to colimit computations, while right-adjoint ones preserve limits and are well behaved with respect to limit computations. It will be interesting to analyze the role of adjointness on mixed limit–colimit computations in different categories.
- (8) Extend the notion of categorical and functor complexity to categories that are further enriched—for example, triangulated categories, derived categories, and so on. This would enable one to study for example the complexity of sheaves in the derived category and various natural functors (for example, the six operations of Grothendieck) from the point of view of categorical complexity. A first attempt toward studying the complexity of constructible sheaves and their functors is undertaken in [Bas15]. However, the definition of complexity in [Bas15] is not categorical.

Acknowledgements

The first author was supported in part by NSF grants CCF-1319080, CCF-1618981, DMS-1620271, and CCF-1910441 while working on this paper.

Conflict of Interest: None.

References

- [AR94] C. Andradas and J. M. Ruiz, ‘Ubiquity of Lojasiewicz’s example of a nonbasic semialgebraic set’, *Michigan Math. J.* **41**(3) (1994), 465–472. MR 1297702.
- [Awo10] S. Awodey, *Category Theory* (Oxford University Press, Oxford, UK, 2010).
- [Bas15] S. Basu, ‘A complexity theory of constructible functions and sheaves’, *Found. Comput. Math.* **15**(1) (2015), 199–279.
- [BP18] S. Basu and D. Patel, ‘Connectivity of joins, cohomological quantifier elimination, and an algebraic Toda’s theorem’, Preprint, 2018, [arXiv:1812.07483](https://arxiv.org/abs/1812.07483).
- [BPR05] S. Basu, R. Pollack and M.-F. Roy, ‘Betti number bounds, applications and algorithms’, in *Current Trends in Combinatorial and Computational Geometry: Papers from the Special Program at MSRI* (Cambridge University Press, Cambridge, UK, 2005), 87–97. MSRI Publications, 52.
- [BPR06] S. Basu, R. Pollack and M.-F. Roy, *Algorithms in Real Algebraic Geometry*, Algorithms and Computation in Mathematics, 10 (Springer, Berlin, 2006), MR 1998147 (2004g:14064).
- [BZ10] S. Basu and T. Zell, ‘Polynomial hierarchy, Betti numbers, and a real analogue of Toda’s theorem’, *Found. Comput. Math.* **10**(4) (2010), 429–454. MR 2657948.
- [BGMW11] E. Bierstone, D. Grigoriev, P. Milman and J. Włodarczyk, ‘Effective Hironaka resolution and its complexity’, *Asian J. Math.* **15**(2) (2011), 193–228. MR 2838220.
- [BCS97] P. Bürgisser, M. Clausen and M. Amin Shokrollahi, *Algebraic Complexity Theory*, Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], 315 (Springer, Berlin, 1997), With the collaboration of Thomas Lickteig. MR 1440179.
- [BCSS98] L. Blum, F. Cucker, M. Shub and S. Smale, *Complexity and Real Computation* (Springer, New York, 1998), With a foreword by Richard M. Karp. MR 1479636 (99a:68070).
- [Bür00] P. Bürgisser, *Completeness and Reduction in Algebraic Complexity Theory*, Algorithms and Computation in Mathematics, 7 (Springer, Berlin, 2000).
- [Gal63] D. Gale, *Neighborly and Cyclic Polytopes*, Proceedings of Symposia in Pure Mathematics, VII (American Mathematical Society, Providence, RI, 1963), 225–232. MR 0152944.
- [Imm95] N. Immerman, ‘Descriptive complexity: a logician’s approach to computation’, *Not. Amer. Math. Soc.* **42**(10) (1995), 1127–1133. MR 1350010.
- [Isi19] M. Umut Isik, ‘Complexity classes and completeness in algebraic geometry’, *Found. Comput. Math.* **19**(2) (2019), 245–258. MR 3937954.
- [Joh02] P. T. Johnstone, *Sketches of an Elephant: a Topos Theory Compendium*, Oxford Logic Guides, 43 1 (The Clarendon Press, Oxford University Press, New York, 2002), MR 1953060.
- [Kal88] E. Kaltofen, ‘Greatest common divisors of polynomials given by straight-line programs’, *J. ACM (JACM)* **35**(1) (1988), 231–264.
- [KR00] M. Kreuzer and L. Robbiano, *Computational Commutative Algebra. 1* (Springer, Berlin, 2000), MR 1790326.
- [LS88] J. Lambek and P. J. Scott, *Introduction to Higher Order Categorical Logic*, Cambridge Studies in Advanced Mathematics, 7 (Cambridge University Press, Cambridge, 1988), Reprint of the 1986 original. MR 939612.

- [MFK94] D. Mumford, J. Fogarty and F. Kirwan, *Geometric Invariant Theory*, 3rd edn, *Ergebnisse der Mathematik und ihrer Grenzgebiete (2) [Results in Mathematics and Related Areas (2)]*, 34 (Springer, Berlin, 1994), MR 1304906.
- [ML98] S. M. Lane, *Categories for the Working Mathematician*, 2nd edn, *Graduate Texts in Mathematics*, 5 (Springer, New York, 1998), MR 1712872.
- [MLM94] S. M. Lane and I. Moerdijk, *Sheaves in Geometry and Logic*, Universitext (Springer, New York, 1994), A first introduction to topos theory, Corrected reprint of the 1992 edition. MR 1300636 MR 1300636.
- [MO15] D. Mumford and T. Oda, *Algebraic Geometry. II*, *Texts and Readings in Mathematics*, 73 (Hindustan Book Agency, New Delhi, 2015), MR 3443857.
- [Poi95] B. Poizat, *Les petits cailloux*, Nur al-Mantiq wal-Ma'rifah [Light of Logic and Knowledge], 3, Aléas, Lyon, 1995, Une approche modèle-théorique de l'algorithmie. [A model-theoretic approach to algorithms]. MR 1333892.
- [Sim11] H. Simmons, *An Introduction to Category Theory* (Cambridge University Press, Cambridge, 2011), MR 2858226.
- [Str73] V. Strassen, 'Vermeidung von divisionen', *J. Reine Angew. Math.* **264** (1973), 184–202.
- [Val79a] L. G. Valiant, 'Completeness classes in algebra', in *Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing* (ACM, New York, NY, USA, 1979), 249–261.
- [Val79b] L. G. Valiant, 'The complexity of computing the permanent', *Theoret. Comput. Sci.* **8**(2) (1979), 189–201.
- [vzG87] J. von zur Gathen, 'Feasible arithmetic computations: Valiant's hypothesis', *J. Symbolic Comput.* **4**(2) (1987), 137–172.
- [Yan15] N. S. Yanofsky, 'Computability and complexity of categorical structures', Preprint, 2015, CoRR [arXiv:1507.05305](https://arxiv.org/abs/1507.05305).