Sparsification Upper and Lower Bounds for Graphs Problems and Not-All-Equal SAT*

Bart M. P. Jansen and Astrid Pieterse

Eindhoven University of Technology
P. O. Box 513, Eindhoven, The Netherlands
{b.m.p.jansen,a.pieterse}@tue.nl

Abstract

We present several sparsification lower and upper bounds for classic problems in graph theory and logic. For the problems 4-Coloring, (Directed) Hamiltonian Cycle, and (Connected) Dominating Set, we prove that there is no polynomial-time algorithm that reduces any n-vertex input to an equivalent instance, of an arbitrary problem, with bitsize $\mathcal{O}(n^{2-\varepsilon})$ for $\varepsilon > 0$, unless NP $\subseteq$ coNP/poly and the polynomial-time hierarchy collapses. These results imply that existing linear-vertex kernels for k-Nonblocker and k-Max Leaf Spanning Tree (the parametric duals of (Connected) Dominating Set) cannot be improved to have $\mathcal{O}(k^{2-\varepsilon})$ edges, unless NP $\subseteq$ coNP/poly. We also present a positive result and exhibit a non-trivial sparsification algorithm for d-Not-All-Equal-SAT. We give an algorithm that reduces an n-variable input with clauses of size at most d to an equivalent input with $\mathcal{O}(n^{d-1})$ clauses, for any fixed d. Our algorithm is based on a linear-algebraic proof of Lovász that bounds the number of hyperedges in critically 3-chromatic d-uniform n-vertex hypergraphs by $\binom{n}{d-1}$. We show that our kernel is tight under the assumption that NP $\not\subseteq$ coNP/poly.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2.2 Graph Theory

Keywords and phrases sparsification, graph coloring, Hamiltonian cycle, satisfiability

Digital Object Identifier 10.4230/LIPIcs.IPEC.2015.163

1 Introduction

Background. Sparsification refers to the method of reducing an object such as a graph or CNF-formula to an equivalent object that is less dense, that is, an object in which the ratio of edges to vertices (or clauses to variables) is smaller. The notion is fruitful in theoretical [16] and practical (cf. [10]) settings when working with (hyper)graphs and formulas. The theory of kernelization, originating from the field of parameterized complexity theory, can be used to analyze the limits of polynomial-time sparsification. Using tools developed in the last five years, it has become possible to address questions such as: "Is there a polynomial-time algorithm that reduces an n-vertex instance of my favorite graph problem to an equivalent instance with a subquadratic number of edges?"

The impetus for this line of analysis was given by an influential paper by Dell and van Melkebeek [8] (conference version in 2010). One of their main results states that if there is an $\varepsilon > 0$ and a polynomial-time algorithm that reduces any n-vertex instance of Vertex Cover to an equivalent instance, of an arbitrary problem, that can be encoded in $\mathcal{O}(n^{2-\varepsilon})$

* This work was supported by NWO Veni grant "Frontiers in Parameterized Preprocessing" and NWO Gravity grant "Networks".

bits, then NP $\subseteq$ coNP/poly and the polynomial-time hierarchy collapses. Since any nontrivial input (G, k) of Vertex Cover has $k \leq n = |V(G)|$, their result implies that the number of edges in the $2k$-vertex kernel for k-Vertex Cover [22] cannot be improved to $\mathcal{O}(k^{2-\varepsilon})$ unless NP $\subseteq$ coNP/poly.

Using related techniques, Dell and van Melkebeek also proved important lower bounds for d-cnf-sat problems: testing the satisfiability of a propositional formula in CNF form, where each clause has at most d literals. They proved that for every fixed integer $d \geq 3$, the existence of a polynomial-time algorithm that reduces any n-variable instance of d-cnf-sat to an equivalent instance, of an arbitrary problem, with $\mathcal{O}(n^{d-\varepsilon})$ bits, for some $\varepsilon > 0$ implies NP $\subseteq$ coNP/poly. Their lower bound is tight: there are $\mathcal{O}(n^d)$ possible clauses of size d over n variables, allowing an instance to be represented by a vector of $\mathcal{O}(n^d)$ bits that specifies for each clause whether or not it is present.

Our results. We continue this line of investigation and analyze sparsification for several classic problems in graph theory and logic. We obtain several sparsification lower bounds that imply that the quadratic number of edges in existing linear-vertex kernels is likely to be unavoidable. When it comes to problems from logic, we give the—to the best of our knowledge—first example of a problem that *does* admit nontrivial sparsification: d-Not-All-Equal-SAT. We also provide a matching lower bound.

The first problem we consider is 4-Coloring, which asks whether the input graph has a proper vertex coloring with 4 colors. Using several new gadgets, we give a cross-composition [3] to show that the problem has no compression of size $\mathcal{O}(n^{2-\varepsilon})$ unless NP $\subseteq$ coNP/poly. To obtain the lower bound, we give a polynomial-time construction that embeds the logical or of a series of t size-n inputs of an NP-hard problem into a graph G' with $\mathcal{O}(\sqrt{t} \cdot n^{\mathcal{O}(1)})$ vertices, such that G' has a proper 4-coloring if and only if there is a *yes*-instance among the inputs. The main structure of the reduction follows the approach of Dell and Marx [7]: we create a table with two rows and $\mathcal{O}(\sqrt{t})$ columns and $\mathcal{O}(n^{\mathcal{O}(1)})$ vertices in each cell. For each way of picking one cell from each row, we aim to embed one instance into the edge set between the corresponding groups of vertices. When the NP-hard starting problem is chosen such that the t inputs each decompose into two induced subgraphs with a simple structure, one can create the vertex groups and their connections such that for each pair of cells (i, j), the subgraph they induce represents the $i \cdot \sqrt{t} + j$-th input. If there is a *yes*-instance among the inputs, this leads to a pair of cells that can be properly colored in a structured way. The challenging part of the reduction is to ensure that the edges in the graph corresponding to *no*-inputs do not give conflicts when extending this partial coloring to the entire graph.

The next problem we attack is Hamiltonian Cycle. We rule out compressions of size $\mathcal{O}(n^{2-\varepsilon})$ for the directed and undirected variant of the problem, under the assumption that NP $\not\subseteq$ coNP/poly. The construction is inspired by kernelization lower bounds for Directed Hamiltonian Cycle parameterized by the vertex-deletion distance to a directed graph whose underlying undirected graph is a path [2].

By combining gadgets from kernelization lower bounds for two different parameterizations of Red Blue Dominating Set, we prove that there is no compression of size $\mathcal{O}(n^{2-\varepsilon})$ for Dominating Set unless NP $\subseteq$ coNP/poly. The same construction rules out subquadratic compressions for Connected Dominating Set. These lower bounds have implications for the kernelization complexity of the parametric duals Nonblocker and Max Leaf Spanning Tree of (Connected) Dominating Set. For both Nonblocker and Max Leaf there are kernels with $\mathcal{O}(k)$ vertices [6, 11] that have $\Theta(k^2)$ edges. Our lower bounds imply that the number of edges in these kernels cannot be improved to $\mathcal{O}(k^{2-\varepsilon})$, unless NP $\subseteq$ coNP/poly.

The final family of problems we consider is d-Not-All-Equal-SAT for fixed $d \geq 4$. The input consists of a formula in CNF-form with at most d literals per clause. The question is whether there is an assignment to the variables such that each clause contains both a variable that evaluates to *true* and one that evaluates to *false*. There is a simple linear-parameter transformation from d-CNF-SAT to $(d+1)$-NAE-SAT that consists of adding one variable that occurs as a positive literal in all clauses. By the results of Dell and van Melkebeek discussed above, this implies that d-NAE-SAT does not admit compressions of size $\mathcal{O}(n^{d-1-\varepsilon})$ unless NP $\subseteq$ coNP/poly. We prove the surprising result that this lower bound is tight! A linear-algebraic result due to Lovász [21], concerning the size of critically 3-chromatic d-uniform hypergraphs, can be used to give a kernel for d-NAE-SAT with $\mathcal{O}(n^{d-1})$ clauses for every fixed d. The kernel is obtained by computing the basis of an associated matrix and removing the clauses that can be expressed as a linear combination of the basis clauses.

Related work. Dell and Marx introduced the table structure for compression lower bounds in their study of compression for packing problems [7]. Hermelin and Wu [15] analyzed similar problems. Other papers about polynomial kernelization and sparsification lower bounds include [5] and [17].

2 Preliminaries

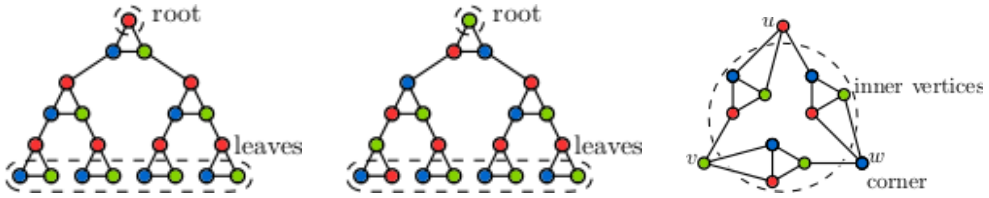
A parameterized problem $\mathcal{Q}$ is a subset of $\Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. Let $\mathcal{Q}, \mathcal{Q}' \subseteq \Sigma^* \times \mathbb{N}$ be parameterized problems and let $h\colon \mathbb{N} \to \mathbb{N}$ be a computable function. A *generalized kernel for $\mathcal{Q}$ into $\mathcal{Q}'$ of size $h(k)$* is an algorithm that, on input $(x,k) \in \Sigma^* \times \mathbb{N}$, takes time polynomial in $|x|+k$ and outputs an instance (x',k') such that: (i) $|x'|$ and k' are bounded by $h(k)$, and (ii) $(x',k') \in \mathcal{Q}'$ if and only if $(x,k) \in \mathcal{Q}$. The algorithm is a *kernel* for $\mathcal{Q}$ if $\mathcal{Q}' = \mathcal{Q}$. It is a *polynomial (generalized) kernel* if $h(k)$ is a polynomial.

Since a polynomial-time reduction to an equivalent sparse instance yields a generalized kernel, we will use the concept of generalized kernels in the remainder of this paper to prove the non-existence of such sparsification algorithms. We employ the cross-composition framework by Bodlaender *et al.* [3], which builds on earlier work by several authors [1, 8, 13].

▶ **Definition 1** (Polynomial equivalence relation)**.** An equivalence relation $\mathcal{R}$ on Σ^* is called a *polynomial equivalence relation* if the following conditions hold. (i) There is an algorithm that, given two strings $x, y \in \Sigma^*$, decides whether x and y belong to the same equivalence class in time polynomial in $|x|+|y|$. (ii) For any finite set $S \subseteq \Sigma^*$ the equivalence relation $\mathcal{R}$ partitions the elements of S into a number of classes that is polynomially bounded in the size of the largest element of S.

▶ **Definition 2** (Cross-composition)**.** Let $L \subseteq \Sigma^*$ be a language, let $\mathcal{R}$ be a polynomial equivalence relation on Σ^*, let $\mathcal{Q} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem, and let $f\colon \mathbb{N} \to \mathbb{N}$ be a function. An OR-*cross-composition of L into $\mathcal{Q}$* (with respect to $\mathcal{R}$) *of cost $f(t)$* is an algorithm that, given t instances $x_1, x_2, \ldots, x_t \in \Sigma^*$ of L belonging to the same equivalence class of $\mathcal{R}$, takes time polynomial in $\sum_{i=1}^{t} |x_i|$ and outputs an instance $(y,k) \in \Sigma^* \times \mathbb{N}$ such that: (i) the parameter k is bounded by $\mathcal{O}(f(t) \cdot (\max_i |x_i|)^c)$, where c is some constant independent of t, and (ii) $(y,k) \in \mathcal{Q}$ if and only if there is an $i \in [t]$ such that $x_i \in L$.

▶ **Theorem 3** ([3])**.** *Let $L \subseteq \Sigma^*$ be a language, let $\mathcal{Q} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem, and let d, ε be positive reals. If L is NP-hard under Karp reductions, has an* OR*-cross-composition into $\mathcal{Q}$ with cost $f(t) = t^{1/d+o(1)}$, where t denotes the number of instances, and $\mathcal{Q}$ has a polynomial (generalized) kernelization with size bound $\mathcal{O}(k^{d-\varepsilon})$, then* NP $\subseteq$ coNP/poly.



(a) Treegadget with no red leaf. **(b)** Treegadget where one of the leaves is red. **(c)** Triangular gadget.

Figure 1 Used gadgets with example colorings.

For $r \in \mathbb{N}$ we will refer to an OR-cross-composition of cost $f(t) = t^{1/r} \log(t)$ as a *degree-r cross-composition*. By Theorem 2, a degree-r cross-composition can be used to rule out generalized kernels of size $\mathcal{O}(k^{r-\varepsilon})$. We frequently use the fact that a polynomial-time linear-parameter transformation from problem $\mathcal{Q}$ to $\mathcal{Q}'$ implies that any generalized kernelization lower bound for $\mathcal{Q}$, also holds for $\mathcal{Q}'$ (cf. [3, 4]). Let $[r]$ be defined as $[r] := \{x \in \mathbb{N} \mid 1 \leq x \leq r\}$. For statements marked with a $(\star)$, the proof can be found in the full version [19].

3 4-Coloring

In this section we analyze the 4-Coloring problem, which asks whether it is possible to assign each vertex of the input graph one out of 4 possible colors, such that there is no edge whose endpoints share the same color. We show that 4-Coloring does not have a generalized kernel of size $\mathcal{O}(n^{2-\varepsilon})$, by giving a degree-2 cross-composition from a tailor-made problem that will be introduced below. Before giving the construction, we first present and analyze some of the gadgets that will be needed.

▶ **Definition 4.** A *treegadget* is the graph obtained from a complete binary tree by replacing each vertex v by a triangle on vertices r_v, x_v and y_v. Let r_v be connected to the parent of v and let x_v and y_v be connected to the left and right subtree of v. An example of a treegadget with 8 leaves is shown in Figure 1. If vertex v is the root of the tree, then r_v is named the *root* of the treegadget. If v does not have a left subtree, then x_v is a *leaf* of this gadget, similarly, if v does not have a right subtree then we refer to y_v as a leaf of the gadget. Let the *height* of a treegadget be equal to the height of its corresponding binary tree.

It is easy to see that a treegadget is 3-colorable. The important property of this gadget is that if there is a color that does not appear on any leaf in a proper 3-coloring, then this must be the color of the root. See Figure 1a for an illustration.

▶ **Lemma 5.** *Let T be a treegadget with root r and let $c\colon V(T) \to \{1, 2, 3\}$ be a proper 3-coloring of T. If $k \in \{1, 2, 3\}$ such that $c(v) \neq k$ for every leaf v of T, then $c(r) = k$.*

Proof. This will be proven using induction on the structure of a treegadget. For a single triangle, the result is obvious. Suppose we are given a treegadget of height h and that the statement holds for all treegadgets of smaller height. Consider the top triangle r, x, y where r is the root. Then, by the induction hypothesis, the roots of the left and right subtree (if non-empty) are colored using k. If the left or right subtree is empty, x or y is a leaf. Hence x and y do not use color k. Since x, y, r is a triangle, r has color k in the 3-coloring. ◀

The following lemma will be used in the correctness proof of the cross-composition to argue that the existence of a single *yes*-input is sufficient for 4-colorability of the entire graph.

▶ **Lemma 6.** *Let T be a treegadget with leaves $L \subseteq V(T)$ and root r. Any 3-coloring $c' : L \to \{1,2,3\}$ that is proper on $T[L]$ can be extended to a proper 3-coloring of T. If there is a leaf $v \in L$ such that $c'(v) = i$, then such an extension exists with $c(r) \neq i$.*

Proof. We will prove this by induction on the height of the treegadget. For a single triangle, the result is obvious. Suppose the lemma is true for all treegadgets up to height $h-1$ and we are given a treegadget of height h with root triangle r, x, y and with coloring of the leaves c'. Let one of the leaves be colored using i. Without loss of generality assume this leaf is in the left subtree, which is connected to x. By the induction hypothesis, we can extend the coloring restricted to the leaves of the left subtree to a proper 3-coloring of the left subtree such that $c(r_1) \neq i$. We assign color i to x. Since c' restricted to the leaves in the right subtree is a proper 3-coloring of the leaves in the right subtree, by induction we can extend that coloring to a proper 3-coloring of the right subtree. Suppose the root of this subtree gets color $j \in \{1,2,3\}$. We now color y with a color $k \in \{1,2,3\} \setminus \{i,j\}$, which must exist. Finally, choose $c(r) \in \{1,2,3\} \setminus \{i,k\}$. By definition, the vertices r, y, and x are now assigned a different color. Both x and y have a different color than the root of their corresponding subtree, thereby c is a proper coloring. We obtain that the defined coloring c is a proper coloring extending c' with $c(r) \neq i$. ◀

▶ **Definition 7.** A *triangular gadget* is a graph on 12 vertices depicted in Figure 1c. Vertices u, v, and w are the *corners* of the gadget, all other vertices are referred to as *inner vertices*.

It is easy to see that a triangular gadget is always 3-colorable in such a way that every corner gets a different color. Moreover, we make the following observation.

▶ **Observation 8.** *Let G be a triangular gadget with corners u,v and w and let $c : V(G) \to \{1,2,3\}$ be a proper 3-coloring of G. Then $c(v) \neq c(u) \neq c(w) \neq c(v)$. Furthermore, every partial coloring that assigns distinct colors to the three corners of a triangular gadget can be extended to a proper 3-coloring of the entire gadget.*

Having presented all the gadgets we use in our construction, we now define the source problem for the cross-composition. It is a variant of the problem that was used to prove kernel lower bounds for Chromatic Number parameterized by vertex cover [3].

2-3-Coloring with Triangle Split Decomposition
Input: A graph G with a partition of its vertex set into $X \cup Y$ such that $G[X]$ is an edgeless graph and $G[Y]$ is a disjoint union of triangles.
Question: Is there a proper 3-coloring $c : V(G) \to \{1,2,3\}$ of G, such that $c(x) \in \{1,2\}$ for all $x \in X$? We will refer to such a coloring as a *2-3-coloring* of G.

▶ **Lemma 9** (⋆). 2-3-Coloring with Triangle Split Decomposition *is NP-complete.*

▶ **Theorem 10.** 4-Coloring *parameterized by the number of vertices n does not have a generalized kernel of size $\mathcal{O}(n^{2-\varepsilon})$ for any $\varepsilon > 0$, unless* NP $\subseteq$ coNP/poly.

Proof. By Theorem 3 and Lemma 9 it suffices to give a degree-2 cross-composition from the 2-3-coloring problem defined above into 4-Coloring parameterized by the number of vertices. For ease of presentation, we will actually give a cross-composition into the 4-List Coloring problem, whose input consists of a graph G and a list function that assigns every vertex $v \in V(G)$ a list $L(v) \subseteq [4]$ of allowed colors. The question is whether there is a proper coloring of the graph in which every vertex is assigned a color from its list. The 4-List Coloring reduces to the ordinary 4-Coloring by a simple transformation that adds a

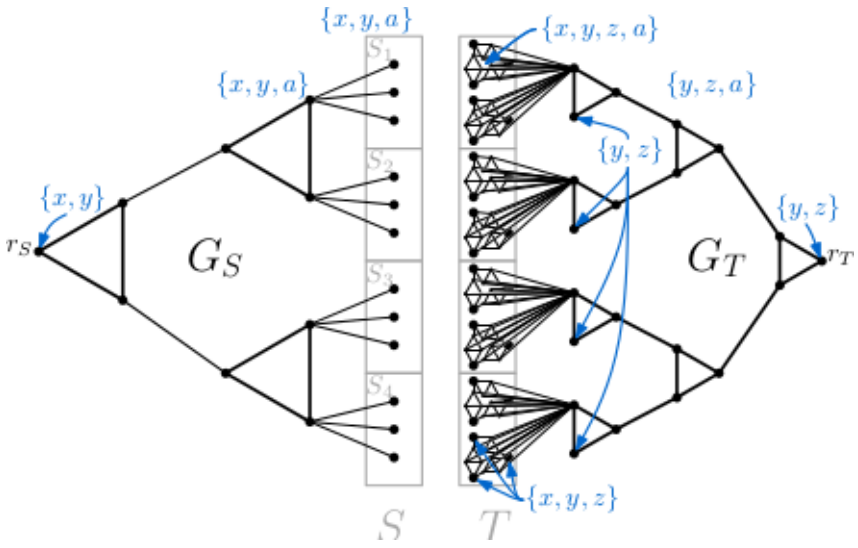


Figure 2 Graph G' for $t' = 4$, $m = 3$ and $n = 2$. Edges between vertices in S and T are left out.

4-clique to enforce the color lists, which will prove the theorem. For now, we focus on giving a cross-composition into 4-List Coloring.

We start by defining a polynomial equivalence relation on inputs of 2-3-Coloring with Triangle Split Decomposition. Let two instances of 2-3-Coloring with Triangle Split Decomposition be equivalent under equivalence relation $\mathcal{R}$ when they have the same number of triangles and the independent sets also have the same size. It is easy to see that $\mathcal{R}$ is a polynomial equivalence relation. By duplicating one of the inputs, we can ensure that the number of inputs to the cross-composition is an even power of two; this does not change the value of or, and increases the total input size by at most a factor four. We will therefore assume that the input consists of t instances of 2-3-Coloring with Triangle Split Decomposition such that $t = 2^{2i}$ for some integer i, implying that $\sqrt{t}$ and $\log \sqrt{t}$ are integers. Let $t' := \sqrt{t}$. Enumerate the instances as $X_{i,j}$ for $1 \leq i, j \leq t'$. Each input $X_{i,j}$ consists of a graph $G_{i,j}$ and a partition of its vertex set into sets U and V, such that U is an independent set of size m and $G_{i,j}[V]$ consists of n vertex-disjoint triangles. Enumerate the vertices in U and V as $u_1, \ldots, u_m$ and $v_1, \ldots, v_{3n}$, such that vertices $v_{3\ell-2}, v_{3\ell-1}$ and $v_{3\ell}$ form a triangle, for $\ell \in [n]$. We will create an instance G' of the 4-List-Coloring problem, which consists of a graph G' and a list function L that assigns each vertex a subset of the color palette $\{x, y, z, a\}$. Refer to Figure 2 for a sketch of G'.

1. Initialize G' as the graph containing t' sets of m vertices each, called S_i for $i \in [t']$. Label the vertices in each of these sets as s^i_ℓ for $i \in [t']$, $\ell \in [m]$ and let $L(s^i_\ell) := \{x, y, a\}$.
2. Add t' sets of n triangular gadgets each, labeled T_j for $j \in [t']$. Label the corner vertices in T_j as t^j_ℓ for $\ell \in [3n]$, such that vertices $t^j_{3\ell-2}, t^j_{3\ell-1}$ and $t^j_{3\ell}$ are the corner vertices of one of the gadgets for $\ell \in [n]$. Let $L(t^j_\ell) := \{x, y, z\}$ and for any inner vertex v of a triangular gadget, let $L(v) := \{x, y, z, a\}$.
3. Connect vertex s^i_k to vertex t^j_ℓ if in graph $G_{i,j}$ vertex u_k is connected to v_ℓ, for $k \in [m]$ and $\ell \in [3n]$. By this construction, the subgraph of G' induced by $S_i \cup T_j$ is isomorphic to the graph obtained from $G_{i,j}$ by replacing each triangle with a triangular gadget.
4. Add a treegadget G_S with t' leaves to G' and enumerate these leaves as $1, \ldots, t'$; recall that t' is a power of two. Connect the i'th leaf of G_S to every vertex in S_i. Let the root of G_S be r_S and define $L(r_S) := \{x, y\}$. For every other vertex v in G_S let $L(v) := \{x, y, a\}$.

5. Add a treegadget G_T with $2t'$ leaves to G' and enumerate these leaves as $1, \ldots, 2t'$. For $j \in [t']$, connect every inner vertex of a triangular gadget in group T_j to leaf number $2j-1$ of G_T. For every leaf v with an even index let $L(v) := \{y, z\}$ and let the root r_T have list $L(r_T) := \{y, z\}$. For every other vertex v of gadget G_T let $L(v) := \{y, z, a\}$.

▶ **Claim 11.** *The graph G' is 4-list-colorable $\Leftrightarrow$ some input instance $X_{i^*j^*}$ is 2-3-colorable.*

Proof. ($\Rightarrow$) Suppose we are given a 4-list coloring c for G'. By definition, $c(r_S) \neq a$. From Lemma 5 it follows that there is a leaf v of G_S such that $c(v) = a$. This leaf is connected to all vertices in some S_{i^*}, which implies that none of the vertices in S_{i^*} are colored using a. Therefore all vertices in S_{i^*} are colored using x and y. Similarly the gadget G_T has at least one leaf v such that $c(v) = a$, note that this must be a leaf with an odd index. Therefore there exists T_{j^*} where all vertices are colored using x,y or z. Thereby in $S_{i^*} \cup T_{j^*}$ only three colors are used, such that S_{i^*} is colored using only two colors. Using Observation 8 and the fact that $G'[S_{i^*} \cup T_{j^*}]$ is isomorphic to the graph obtained from G_{i^*,j^*} by replacing triangles by triangular gadgets, we conclude that $X_{i^*j^*}$ has a proper 2-3-coloring.

($\Leftarrow$) Suppose $c\colon V(G_{i^*,j^*}) \to \{x, y, z\}$ is a proper 2-3-coloring for X_{i^*,j^*}. We will construct a 4-list coloring $c'\colon V(G') \to \{x, y, z, a\}$ for G'. For u_k, $k \in [m]$ in instance X_{i^*,j^*} let $c'(s_k^{i^*}) := c(u_k)$ and for v_ℓ for $\ell \in [3n]$ let $c'(t_\ell^{j^*}) := c(v_\ell)$. Let $c'(s_\ell^i) := a$ for $i \neq i^*$ and $\ell \in [n]$, furthermore let $c'(t_\ell^j) := z$ for $j \neq j^*$ and $\ell \in [3m]$. For triangular gadgets in T_{j^*} the coloring c' defines all corners to have distinct colors; by Observation 8 we can color the inner vertices consistently using $\{x, y, z\}$. For T_j with $j \in [t']$ and $j \neq j^*$, the corners of triangular gadgets have color z and we can now consistently color the inner vertices using $\{x, y, a\}$.

The leaf of gadget G_S that is connected to S_{i^*} can be colored using a. Every other leaf can use both x and y, so we can properly 3-color the leaves such that one leaf has color a. From Lemma 6 it follows that we can consistently 3-color G_S such that the root r_S does not receive color a, as required by $L(r_S)$. Similarly, in triangular gadgets in T_{j^*} the inner vertices do not have color a. As such, leaf $2j^* - 1$ of G_T can be colored using a and we color leaf $2j^*$ with y. For $j \in [t']$ with $j \neq j^*$ color leaf $2j-1$ with z and leaf $2j$ using y. Now the leaves of G_T are properly 3-colored and one is colored a. It follows from Observation 8 that we can color G_T such that the root is not colored a. This completes the 4-list coloring of G'. ◀

The claim shows that the construction serves as a cross-composition into 4-List Coloring. To prove the theorem, we add four new vertices to simulate the list function. Add a clique on 4 vertices $\{x, y, z, a\}$. If for any vertex v in G', some color is not contained in $L(v)$, connect v to the vertex corresponding to this color. As proper colorings of the resulting graph correspond to proper list colorings of G', the resulting graph is 4-colorable if and only if there is a *yes*-instance among the inputs. It remains to bound the parameter of the problem, i.e., the number of vertices. Observe that a treegadget has at least as many leaves as its corresponding binary tree, therefore the graph G' has at most $12mt' + nt' + 6t' + 12t' + 4 = \mathcal{O}(t' \cdot (m+n)) = \mathcal{O}(\sqrt{t} \max |X_{i,j}|)$ vertices. Theorem 10 now follows from Theorem 3 and Lemma 9. ◀

4 Hamiltonian cycle

In this section we prove a sparsification lower bound for Hamiltonian Cycle and its directed variant. The starting problem is Hamiltonian $s-t$ path on bipartite graphs.

HAMILTONIAN $s-t$ PATH ON BIPARTITE GRAPHS
Input: An undirected bipartite graph G with partite sets A and B such that $|B| = n = |A| + 1$, together with two distinguished vertices b_1 and b_n that have degree 1.
Question: Does G have a Hamiltonian path from b_1 to b_n?

It is known that Hamiltonian path is NP-complete on bipartite graphs [14] and it is easy to see that is remains NP-complete when fixing a degree 1 start and endpoint.

▶ **Theorem 12.** (DIRECTED) HAMILTONIAN CYCLE *parameterized by the number of vertices n does not have a generalized kernel of size $\mathcal{O}(n^{2-\varepsilon})$ for any $\varepsilon > 0$, unless* $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

Proof. By a suitable choice of polynomial equivalence relation, and by padding the number of inputs, it suffices to give a cross-composition from the $s-t$ problem on bipartite graphs when the input consists of t instances $X_{i,j}$ for $i, j \in [\sqrt{t}]$ (i.e., $\sqrt{t}$ is an integer), such that each instance $X_{i,j}$ encodes a bipartite graph $G_{i,j}$ with partite sets $A^*_{i,j}$ and $B^*_{i,j}$ with $|A^*_{i,j}| = m$ and $|B^*_{i,j}| = n = m + 1$, for some $m \in \mathbb{N}$. For each instance, label all elements in $A^*_{i,j}$ as $a^*_1, \ldots, a^*_m$ and all elements in $B^*_{i,j}$ as $b^*_1, \ldots, b^*_n$ such that b^*_1 and b^*_n have degree 1.

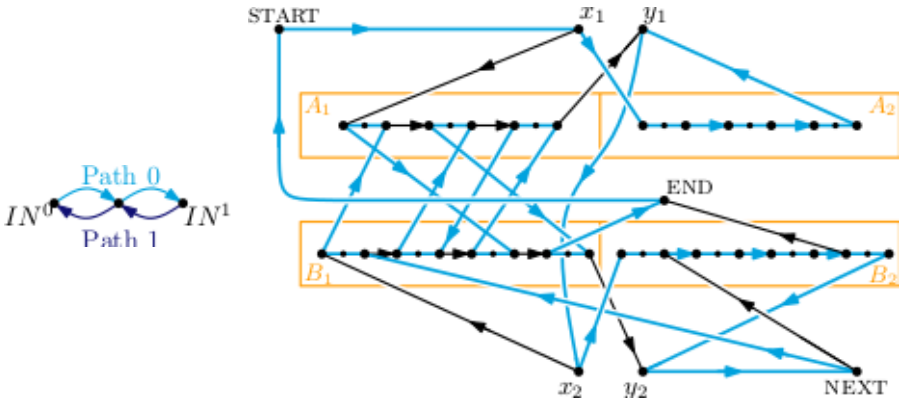
The construction makes extensive use of the path gadget depicted in Figure 3a. Observe that if G' contains a path gadget as an induced subgraph, while the remainder of the graph only connects to its terminals IN^0 and IN^1, then any Hamiltonian cycle in G' traverses the path gadget in one of the two ways depicted in Figure 3a. We create an instance G' of DIRECTED HAMILTONIAN CYCLE that acts as the logical OR of the inputs.

1. First of all construct $\sqrt{t}$ groups of m path gadgets each. Refer to these groups as A_i, for $i \in [\sqrt{t}]$, and label the gadgets within group A_i as $a^i_1, \ldots, a^i_m$. Let the union of all created sets A_i be named A. Similarly, construct $\sqrt{t}$ groups of n path gadgets each. Refer to these groups as B_j, for $j \in [\sqrt{t}]$, and label the gadgets within group B_j as $b^j_1, \ldots, b^j_n$. Let B be the union of all B_j for $j \in [\sqrt{t}]$.
2. For every input instance $X_{i,j}$, for each edge $\{a^*_k, b^*_\ell\}$ in $X_{i,j}$ with $k \in [m]$, $\ell \in [n]$, add an arc from IN^0 of a^i_k to IN^1 of b^j_ℓ and an arc from IN^0 of b^j_ℓ to IN^1 of a^i_k.

If some $X_{i,j}$ has a Hamiltonian $s-t$ path, it can be mimicked by the combination of A_i and B_j, where for each vertex in $X_{i,j}$ we traverse its path gadget in G', following Path 1. The following construction steps are needed to extend such a path to a Hamiltonian cycle in G'.

3. Add an arc from the IN^1 terminal of a^i_ℓ to the IN^0 terminal of $a^i_{\ell+1}$ for all $\ell \in [m-1]$ and all $i \in [\sqrt{t}]$. Similarly add an arc from the IN^1 terminal of b^i_ℓ to the IN^0 of $b^i_{\ell+1}$ for all $\ell \in [n-1]$ and all $i \in [\sqrt{t}]$.
4. Add a vertex START and a vertex END and the arc (END, START).
5. Let $r := \sqrt{t} - 1$, add $2r$ tuples of vertices, x_i, y_i for $i \in [2r]$ and connect START to x_1. Furthermore, add the arcs (y_i, x_{i+1}) for $i \in [2r-1]$.
6. For $i \leq r$ we add arcs from x_i to the IN^0 terminal of the gadgets $a^j_1, j \in [\sqrt{t}]$. Furthermore we add an arc from IN^1 of a^j_m to y_i for all $j \in [\sqrt{t}]$ and $i \in [r]$. When $i > r$ add arcs from x_i to the IN^0 terminal of b^j_1 for $j \in [\sqrt{t}]$ and connect IN^1 of b^j_n to y_i.
7. Add a vertex NEXT and the arc (y_{2r}, NEXT) and an arc from NEXT to the IN^1 terminal of all gadgets b^j_1 for $j \in [\sqrt{t}]$.
8. Furthermore, add arcs from IN^0 of all gadgets b^j_n to END for $j \in [\sqrt{t}]$. So for each B_j, exactly one vertex has an outgoing arc to END and one has an incoming arc from NEXT.

This completes the construction of G'. A sketch of G' is shown in Figure 3b.



(a) A path gadget. **(b)** The general structure of the created graph, when given 4 inputs with $n = 3$ and $m = 4$.

Figure 3 Illustrations for the lower bound for Hamiltonian Cycle.

▶ **Lemma 13** (⋆). *Graph G' has a directed Hamiltonian cycle if and only if at least one of the instances $X_{i,j}$ has a Hamiltonian $s-t$-path.*

The number of vertices of G' is $3(m+n)\sqrt{t}+3\cdot 2(\sqrt{t}-1)+3 = \mathcal{O}(\sqrt{t}\cdot(m+n)) = \mathcal{O}(\sqrt{t}\cdot\max|X_{i,j}|)$. By with Lemma 13 the construction is a degree-2 cross-composition from Hamiltonian $s-t$-paths in Bipartite graphs to Directed Hamiltonian cycle parameterized by the number of vertices, proving the generalized kernel lower bound for the directed problem. Karp [20] gave a polynomial-time reduction that, given an n-vertex directed graph G, produces an undirected graph G' with $3n$ vertices such that G has a directed Hamiltonian cycle if and only if G' has a Hamiltonian cycle. This is a linear parameter transformation from directed Hamiltonian cycle to Hamiltonian cycle. Since linear-parameter transformations transfer lower bounds [3, 4], we conclude that (Directed) Hamiltonian cycle does not have a generalized kernel of size $\mathcal{O}(n^{2-\varepsilon})$ for any $\varepsilon > 0$. ◀

5 Dominating set

In this section we discuss the Dominating Set problem and its variants. Dom *et al.* [9] proved several kernelization lower bounds for the variant Red-Blue Dominating Set, which is the variant on bipartite (red/blue colored) graphs in which the goal is to dominate all the blue vertices by selecting a small subset of red vertices. Using ideas from their kernel lower bounds for the parameterization by either the number of red or the number of blue vertices, we prove sparsification lower bounds for (Connected) Dominating Set. Since we parameterize by the number of vertices, the same lower bounds apply to the dual problems Nonblocker and Max Leaf Spanning Tree, resulting in the following theorem.

▶ **Theorem 14** (⋆). (Connected) Dominating Set, Nonblocker, *and* Max Leaf Spanning Tree *parameterized by the number of vertices n do not have a generalized kernel of size $\mathcal{O}(n^{2-\varepsilon})$ for any $\varepsilon > 0$, unless* NP $\subseteq$ coNP/poly.

The proof is deferred to the complete version [19] due to space restrictions. Just as the sparsification lower bounds for Vertex Cover that were presented by Dell and van Melkebeek [8] had implications for the parameterization by the solution size k, Theorem 14 has implications for the kernelization complexity of k-Nonblocker and k-Max Leaf.

Since the solution size k never exceeds the number of vertices in this problem, a kernel with $\mathcal{O}(k^{2-\epsilon})$ edges would give a nontrivial sparsification, contradicting Theorem 14. Hence our results show that the existing linear-vertex kernels for k-Nonblocker [6] and k-Max Leaf [11] cannot be improved to $\mathcal{O}(k^{2-\varepsilon})$ edges unless NP $\subseteq$ coNP/poly.

6 d-Hypergraph 2-Colorability and d-NAE-SAT

The goal of this section is to give a nontrivial sparsification algorithm for NAE-SAT and prove a matching lower bound. For ease of presentation, we start by analyzing the closely related hypergraph 2-colorability problem. Recall that a hypergraph consists of a vertex set V and a set E of *hyperedges*; each hyperedge $e \in E$ is a subset of V. A 2-coloring of a hypergraph is a function $c\colon V \to \{1,2\}$; such a coloring is *proper* if there is no hyperedge whose vertices all obtain the same color. We will use d-Hypergraph 2-Colorability to refer to the setting where hyperedges have size at most d. The corresponding decision problem asks, given a hypergraph, whether it is 2-colorable.

▶ **Theorem 15.** *d-Hypergraph 2-Colorability parameterized by the number of vertices n has a kernel with $2 \cdot n^{d-1}$ hyperedges that can be encoded in $\mathcal{O}(n^{d-1} \cdot d \cdot \log n)$ bits.*

Proof. Suppose we are given a hypergraph with vertex set V and hyperedges E, where each hyperedge contains at most d vertices. We show how to reduce the number of hyperedges without changing the 2-colorability status. Let $E_r \subseteq E$ denote the set of edges in E that contain exactly r vertices. For each E_r we construct a set $E'_r \subseteq E_r$ of *representative hyperedges*. Enumerate the edges in E_r as $e^r_1, \ldots, e^r_k$. We construct a $(0,1)$-matrix M_r with $N := \binom{n}{r-1}$ rows and k columns. Consider all possible subsets $A_1, \ldots, A_N$ of size $r-1$ of the set of vertices V. Define the elements $m_{i,j}$ for $i \in N$ and $j \in k$ of M_r as follows.

$$m_{i,j} := \begin{cases} 1 & \text{if } A_i \subseteq e^r_j; \\ 0 & \text{otherwise.} \end{cases}$$

Using Gaussian elimination, compute a basis B of the columns of this matrix, which is a subset of the columns that span the column space of M_r. Let E'_r contain edge e^r_i if the i'th column of M_r is contained in B, and define $E' := \bigcup_{r \in [d]} E'_r$, which forms the kernel. Using a lemma due to Lovász [21], we can prove that E' preserves the 2-colorability status.

▶ **Lemma 16** ($\star$). *(V,E) has a proper 2-coloring $\Leftrightarrow$ (V,E') has a proper 2-coloring.*

To bound the size of the kernel, consider the matrix M_r for $r \in [d]$. Its rank is bounded by the minimum of its number of rows and columns, which is at most $\binom{n}{r-1} \leq n^{r-1}$. As such, we get $|E'_r| \leq \operatorname{rank}(M_r) \leq n^{r-1}$. Note that $d \leq n$, such that $|E'| \leq \sum_{r=1}^{d} n^{r-1} = n^{d-1} + \sum_{r=1}^{d-1} n^{r-1} \leq 2 \cdot n^{d-1}$. So E' contains at most $2n^{d-1}$ hyperedges. Since a hyperedge consists of at most d vertices, the kernel can be encoded in $\mathcal{O}(n^{d-1} \cdot d \cdot \log n)$ bits. ◀

By a folklore reduction, Theorem 15 gives a sparsification for NAE-SAT. Consider an instance of d-NAE-SAT, which is a conjunction of clauses of size at most d over variables $x_1, \ldots, x_n$. The formula gives rise to a hypergraph on vertex set $\{x_i, \neg x_i \mid i \in [n]\}$ containing one hyperedge per clause, whose vertices correspond to the literals in the clause. When additionally adding n hyperedges $\{x_i, \neg x_i\}$ for $i \in [n]$, it is easy to see that the resulting hypergraph is 2-colorable if and only if there is a NAE-satisfying assignment to the formula. The maximum size of a hyperedge matches the maximum size of a clause and the number of created vertices is twice the number of variables. We can therefore sparsify an n-variable

instance of d-NAE-SAT in the following way: reduce it to a d-hypergraph with $n' := 2n$ vertices and apply the kernelization algorithm of Theorem 15. It is easy to verify that restricting the formula to the representative hyperedges in the kernel gives an equisatisfiable formula containing $2 \cdot (n')^{d-1} \in \mathcal{O}(2^{d-1}n^{d-1})$ clauses, giving a sparsification for NAE-SAT. As mentioned in the introduction, the existence of a linear-parameter transformation [18] from d-CNF-SAT to $(d+1)$-NAE-SAT also implies a sparsification *lower bound* for d-NAE-SAT, using the results of Dell and van Melkebeek [8]. Hence we obtain the following theorem.

▶ **Theorem 17.** *For every fixed $d \geq 4$, the d-NAE-SAT problem parameterized by the number of variables n has a kernel with $\mathcal{O}(n^{d-1})$ clauses that can be encoded in $\mathcal{O}(n^{d-1} \cdot \log n)$ bits, but admits no generalized kernel of size $\mathcal{O}(n^{d-1-\varepsilon})$ for $\varepsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.*

7 Conclusion

We have added several classic graph problems to a growing list of problems for which non-trivial polynomial-time sparsification is provably impossible under the assumption that $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$. Our results for (Connected) Dominating Set proved that the linear-vertex kernels with $\Theta(k^2)$ edges for k-Nonblocker and k-Max Leaf Spanning Tree cannot be improved to $\mathcal{O}(k^{2-\varepsilon})$ edges unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

The graph problems for which we proved sparsification lower bounds can be defined in terms of vertices: the 4-Coloring problem asks for a partition of the vertex set into four independent sets, Dominating Set asks for a dominating subset of vertices, and Hamiltonian Cycle asks for a permutation of the vertices that forms a cycle. In contrast, not much is known concerning sparsification lower bounds for problems whose solution is an edge subset of possibly quadratic size. For example, no sparsification lower bounds are known for well-studied problems such as Max Cut, Cluster Editing, or Feedback Arc Set in Tournaments. Difficulties arise when attempting to mimic our lower bound constructions for such edge-based problems. Our constructions all embed t instances into a $2 \times \sqrt{t}$ table, using each combination of a cell in the top row and bottom row to embed one input. For problems defined in terms of edge subsets, it becomes difficult to "turn off" the contribution of edges that are incident on vertices that do not belong to the two cells that correspond to a *yes*-instance among the inputs to the or-construction. This could be interpreted as evidence that edge-based problems such as Max Cut might admit non-trivial polynomial sparsification. We have not been able to answer this question in either direction, and leave it as an open problem. For completeness, we point out that Karp's reduction [20] from Vertex Cover to Feedback Arc Set (which only doubles the number of vertices) implies, using existing bounds for Vertex Cover [8], that Feedback Arc Set does not have a compression of size $\mathcal{O}(n^{2-\varepsilon})$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

Another problem whose compression remains elusive is 3-Coloring. In several settings (cf. [12]), the optimal kernel size matches the size of minimal obstructions in a problem-specific partial order. This is the case for d-NAE-SAT, whose kernel with $\mathcal{O}(n^{d-1})$ clauses matches the fact that critically 3-chromatic d-uniform hypergraphs have at most $\mathcal{O}(n^{d-1})$ hyperedges. Following this line of reasoning, it is tempting to conjecture that 3-Coloring does not admit subquadratic compressions: there are critically 4-chromatic graphs with $\Theta(n^2)$ edges [23].

The kernel we have given for d-NAE-SAT is one of the first examples of non-trivial polynomial-time sparsification for general structures that are not planar or similarly guaranteed to be sparse. Obtaining non-trivial sparsification algorithms for other problems is an interesting challenge for future work. Are there natural problems defined on general graphs that admit subquadratic sparsification?

References

1 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.

2 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernel bounds for path and cycle problems. *Theor. Comput. Sci.*, 511:117–136, 2013.

3 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM J. Discrete Math.*, 28(1):277–305, 2014.

4 Hans L. Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel bounds for disjoint cycles and disjoint paths. *Theor. Comput. Sci.*, 412(35):4570–4578, 2011.

5 Marek Cygan, Fabrizio Grandoni, and Danny Hermelin. Tight kernel bounds for problems on graphs with small degeneracy. In *Proc. 21st ESA*, pages 361–372, 2013.

6 Frank K. H. A. Dehne, Michael R. Fellows, Henning Fernau, Elena Prieto, and Frances A. Rosamond. NONBLOCKER: parameterized algorithmics for minimum dominating set. In *Proc. 32nd SOFSEM*, pages 237–245, 2006.

7 Holger Dell and Dániel Marx. Kernelization of packing problems. In *Proc. 23rd SODA*, pages 68–81, 2012.

8 Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *J. ACM*, 61(4):23:1–23:27, 2014.

9 Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and IDs. *ACM Trans. Algorithms*, 11(2):13:1–13:20, 2014.

10 David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Amnon Nissenzweig. Sparsification - a technique for speeding up dynamic graph algorithms. *J. ACM*, 44(5):669–696, 1997.

11 Vladimir Estivill-Castro, Michael Fellows, Michael Langston, and Frances Rosamond. FPT is P-time extremal structure I. In *Proc. 1st ACiD*, pages 1–41, 2005.

12 Michael R. Fellows and Bart M. P. Jansen. FPT is characterized by useful obstruction sets: Connecting algorithms, kernels, and quasi-orders. *TOCT*, 6(4):16, 2014.

13 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.

14 Michael R. Garey and David S. Johnson. *Computers and Intractability.* W.H. Freeman, 1979.

15 Danny Hermelin and Xi Wu. Weak compositions and their applications to polynomial lower bounds for kernelization. In *Proc. 23rd SODA*, pages 104–113, 2012.

16 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

17 Bart M. P. Jansen. On sparsification for computing treewidth. *Algorithmica*, 71(3):605–635, 2015.

18 Bart M. P. Jansen and Stefan Kratsch. Data reduction for graph coloring problems. *Information and Computation*, 231:70–88, 2013.

19 Bart M. P. Jansen and Astrid Pieterse. Sparsification upper and lower bounds for graphs problems and not-all-equal SAT. *CoRR*, abs/1509.07437, 2015.

20 Richard M. Karp. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

21 Lásló Lovász. Chromatic number of hypergraphs and linear algebra. In *Studia Scientiarum Mathematicarum Hungarica 11*, pages 113–114, 1976.

22 George L. Nemhauser and Leslie E. Trotter Jr. Vertex packings: structural properties and algorithms. *Math. Program.*, 8:232–248, 1975.

23 Bjarne Toft. On the maximal number of edges of critical k-chromatic graphs. *Studia Scientiarum Mathematicarum Hungarica*, 5:461–470, 1970.