# Search-Space Reduction via Essential Vertices Revisited: Vertex Multicut and Cograph Deletion

**Bart M. P. Jansen**
Eindhoven University of Technology, The Netherlands

**Ruben F. A. Verhaegh**
Eindhoven University of Technology, The Netherlands

## Abstract

For an optimization problem $\Pi$ on graphs whose solutions are vertex sets, a vertex $v$ is called *$c$-essential* for $\Pi$ if all solutions of size at most $c \cdot \text{OPT}$ contain $v$. Recent work showed that polynomial-time algorithms to detect $c$-essential vertices can be used to reduce the search space of fixed-parameter tractable algorithms solving such problems parameterized by the size $k$ of the solution. We provide several new upper- and lower bounds for detecting essential vertices. For example, we give a polynomial-time algorithm for 3-Essential detection for Vertex Multicut, which translates into an algorithm that finds a minimum multicut of an undirected $n$-vertex graph $G$ in time $2^{\mathcal{O}(\ell^3)} \cdot n^{\mathcal{O}(1)}$, where $\ell$ is the number of vertices in an optimal solution that are *not* 3-essential. Our positive results are obtained by analyzing the integrality gaps of certain linear programs. Our lower bounds show that for sufficiently small values of $c$, the detection task becomes NP-hard assuming the *Unique Games Conjecture*. For example, we show that $(2-\varepsilon)$-Essential detection for Directed Feedback Vertex Set is NP-hard under this conjecture, thereby proving that the existing algorithm that detects 2-essential vertices is best-possible.

**2012 ACM Subject Classification** Theory of computation → Graph algorithms analysis; Theory of computation → Linear programming; Theory of computation → Rounding techniques; Theory of computation → Fixed parameter tractability

**Keywords and phrases** fixed-parameter tractability, essential vertices, integrality gap

**Digital Object Identifier** 10.4230/LIPIcs.SWAT.2024.28

**Related Version** *Full Version*: https://arxiv.org/abs/2404.09769

**Funding** *Bart M. P. Jansen*: Supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 803421, ReduceSearch).

## 1 Introduction

Preprocessing is an important tool for dealing with NP-hard problems. The idea is that before starting a time-consuming computation on an input, one first exhaustively applies simple transformation steps that provably do not affect the desired output, but which make the subsequently applied solver more efficient. Preprocessing is often highly effective in practice [1, 34].

There have been several attempts to theoretically explain the speed-ups obtained by preprocessing. The concept of kernelization [12, 14], phrased in the language of parameterized complexity theory [9, 10], is one such attempt. Recently, Bumpus, Jansen, and de Kroon [4] proposed an alternative framework for developing and analyzing polynomial-time preprocessing algorithms that reduce the search space of subsequently applied algorithms for NP-hard graph problems. They presented the first positive and negative results in this framework, which revolves around the notion of so-called *$c$-essential vertices*. In this paper, we revisit this notion by providing new preprocessing results and new hardness proofs.

To be able to discuss our results, we first introduce and motivate the concept of $c$-essential vertices and the corresponding algorithmic preprocessing task. Our results apply to optimization problems on graphs in which the goal is to find a minimum-size vertex set that hits all obstacles of a certain kind. The (Undirected) Vertex Multicut problem is a prime example. Given an undirected graph $G$, annotated by a collection $\mathcal{T}$ consisting of pairs of terminal vertices, the goal is to find a minimum-size vertex set whose removal disconnects all terminal pairs. The decision version of this problem is NP-complete, but *fixed-parameter tractable* parameterized by the size of the solution: there is an algorithm by Marx and Razgon [26] that, given an $n$-vertex instance together with an integer $k$, runs in time $2^{\mathcal{O}(k^3)} \cdot n^{\mathcal{O}(1)}$ and outputs a solution of size at most $k$, if one exists. The running time therefore scales exponentially in the size of the solution, but polynomially in the size of the graph. This yields a great potential for preprocessing: if an efficient preprocessing phase manages to identify some vertices $S \subseteq V(G)$ that are guaranteed to be part of an optimal solution, then finding a solution of size $k$ in $G$ reduces to finding a solution of size $k - |S|$ in $G - S$, thereby reducing the running time of the applied algorithm and its search space. To be able to give guarantees on the amount of search-space reduction achieved, the question becomes: under which conditions can a polynomial-time preprocessing algorithm identify vertices that belong to an optimal solution?

**Essential vertices.** The approach that Bumpus et al. [4] take when answering this question originates from the idea that it may be feasible to detect vertices as belonging to an optimal solution when they are *essential* for making an optimal solution. This is formalized as follows. For a real number $c \geq 1$ and fixed optimization problem $\Pi$ on graphs whose solutions are vertex subsets, a vertex $v$ of an input instance $G$ is called *$c$-essential* if vertex $v$ is contained in all $c$-approximate solutions to the instance. Hence a $c$-essential vertex is not only contained in all optimal solutions, but even in all solutions whose size is at most $c \cdot \textsc{opt}$. To obtain efficient preprocessing algorithms with performance guarantees, the goal then becomes to develop polynomial-time algorithms to detect $c$-essential vertices in the input graph, when they are present.

For some problems like Vertex Cover (in which the goal is to find a minimum-size vertex set that intersects each edge), it is indeed possible to give a polynomial-time algorithm that, given a graph $G$, outputs a set $S$ of vertices that is part of an optimal vertex cover and contains all 2-essential vertices. For optimization problems whose structure is more intricate, like Odd Cycle Transversal, finding $c$-essential vertices from scratch still seems like a difficult task. Bumpus et al. [4] therefore formulated a slightly easier algorithmic task related to detecting essential vertices and proved that solving this simpler task is sufficient to be able to achieve search-space reduction. For a vertex hitting set problem $\Pi$ whose input is a (potentially annotated) graph $G$ and whose solutions are vertex sets hitting all (implicitly defined) constraints, we denote by $\textsc{opt}_\Pi(G)$ the cardinality of an optimal solution to $G$. The detection task is formally defined as follows, for each real $c \geq 1$.

$c$-Essential detection for $\Pi$
**Input:** A (potentially annotated) graph $G$ and integer $k$.
**Task:** Find a vertex set $S \subseteq V(G)$ such that:
**G1** if $\textsc{opt}_\Pi(G) \leq k$, then there is an optimal solution in $G$ containing all of $S$, and
**G2** if $\textsc{opt}_\Pi(G) = k$, then $S$ contains all $c$-essential vertices.

The definition above simplifies the detection task by supplying an integer $k$ in addition to the input graph, while only requiring the algorithm to work correctly for certain ranges of $k$. The intuition is as follows: when $k$ is correctly guessed as the size of an optimal

solution, the preprocessing algorithm should find all $c$-essential vertices, and is allowed to find additional vertices as long as they are part of an optimal solution. Bumpus et al. [4] give a *dove-tailing*-like scheme that manages to use algorithms for $c$-Essential detection for $\Pi$ to give improved fixed-parameter tractable running times for solving $\Pi$ from scratch. The exponential dependence of the running time of the resulting algorithm is not on the *total* size of the solution, but only on the number of vertices in the solution that are *not* $c$-essential. Hence their results show that large optimal solutions can be found efficiently, as long as they are composed primarily out of $c$-essential vertices. For example, they prove that a minimum vertex set intersecting all odd cycles (a solution to Odd Cycle Transversal) can be computed in time $2.3146^{\ell} \cdot n^{\mathcal{O}(1)}$, where $\ell$ is the number of vertices in an optimal solution that are not 2-essential and which are therefore avoided by at least one 2-approximation. Apart from polynomial-time algorithms for $c$-Essential detection for $\Pi$ for various combinations of $\Pi$ and $c$, they also prove several *lower bounds*. One of their main lower bounds concerns the Perfect Deletion problem, whose goal is to obtain a perfect graph by vertex deletions [18]. They rule out the existence of a polynomial-time algorithm for $c$-Essential detection for Perfect Deletion for any $c \geq 1$, assuming FPT $\neq$ W[1]. (They even rule out detection algorithms running in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some function $f$.)

We continue exploring the framework of search-space reduction by detecting essential vertices, from two directions. We provide both upper bounds (new algorithms for $c$-Essential detection for $\Pi$) as well as lower bounds. We start by discussing the upper bounds.

**Our results: Upper bounds.** The Vertex Multicut problem is the subject of our first results. The problem played a pivotal role in the development of the toolkit of parameterized algorithms for graph separation problems and stood as a famous open problem for years, until being independently resolved by two teams of researchers [3, 26]. The problem is not only difficult to solve exactly, but also to approximate: Chawla et al. [7] proved that, assuming Khot's [21] *Unique Games Conjecture* (UGC), it is NP-hard to approximate the edge-deletion version of the problem within any constant factor. A simple transformation shows that the same holds for the vertex-deletion problem.

Our first result (Theorem 4.2) is a polynomial-time algorithm for 3-Essential detection for Vertex Multicut, which is obtained by analyzing the integrality gap of a restricted type of linear program associated with the problem. Using known results, this preprocessing algorithm translates directly into search-space reduction for the current-best FPT algorithms for solving Vertex Multicut. This results in an algorithm (Corollary 4.3) that computes an optimal vertex multicut in an $n$-vertex graph in time $2^{\mathcal{O}(\ell^3)} \cdot n^{\mathcal{O}(1)}$, where $\ell$ is the number of vertices in an optimal solution that are *not* 3-essential.

Our approach for essential detection also applies for the variation of Vertex Multicut on *directed* graphs. Since the directed setting is more difficult to deal with, vertices have to be slightly more essential to be able to detect them, resulting in a polynomial-time algorithm for 5-Essential detection for Directed Vertex Multicut (Theorem 4.5). This detection algorithm does not directly translate into running-time guarantees for FPT algorithms, though, as Directed Vertex Multicut is W[1]-hard parameterized by the size of the solution [29]. (When the solution is forbidden from deleting terminals, the directed problem is already $W[1]$-hard with four terminal pairs, although the case of three terminal pairs is FPT [17].)

Our second positive result concerns the Cograph (Vertex) Deletion problem. Given an undirected graph $G$, it asks to find a minimum-size vertex set $S$ such that $G-S$ is a cograph, i.e., the graph $G - S$ does not contain the 4-vertex path $P_4$ as an induced subgraph. The

problem is motivated by the fact that efficient algorithms for solving optimization problems on cographs can often be extended to work on graphs which are *close* to being cographs, as long as a deletion set is known [6, §6]. The decision version of Cograph Deletion is NP-complete due to the generic results of Lewis and Yannakakis [24]. Parameterized by the size $k$ of the desired solution, Cograph Deletion is fixed-parameter tractable via the method of bounded-depth search trees [5]: branching on vertices of a $P_4$ results in a running time of $4^k \cdot n^{\mathcal{O}(1)}$. Nastos and Gao [27] proposed a refined branching strategy by exploiting the structure of $P_4$-*sparse graphs*, improving the running time to $3.115^k \cdot n^{\mathcal{O}(1)}$, following earlier improvements via the interpretation of Cograph Deletion as a 4-Hitting Set problem [13, 16, 28]. The latter viewpoint also gives a simple polynomial-time 4-approximation. Whether a $(4-\varepsilon)$-approximation can be computed in polynomial time is unknown; Drescher poses this [11, §8 Question 5] as an open problem for *vertex-weighted* graphs.

Our second result (Lemma 4.6) is a polynomial-time algorithm for 3.5-Essential detection for Cograph Deletion. It directly translates into an FPT algorithm (Corollary 4.8) that, given a graph $G$, outputs a minimum set $S$ for which $G-S$ is a cograph in time $3.115^\ell \cdot n^{\mathcal{O}(1)}$; here $\ell$ is the number of vertices in an optimal solution that are *not* 3.5-essential. Similarly as for Vertex Multicut, our detection algorithm arises from a new bound of 2.5 on the integrality gap of a restricted version of a natural linear-programming relaxation associated to the deletion problem.

The fact that our algorithm detects 3.5-essential vertices is noteworthy. It is known [4, §8] that for any $c \geq 1$, an algorithm for $c$-Essential detection for $\Pi$ follows from an algorithm that computes a factor-$c$ approximation for the problem of finding a minimum-size solution avoiding a given vertex $v$. In this setting, a 4-approximation algorithm for Cograph Deletion easily follows since the problem is a special case of $d$-Hitting Set. We consider it interesting that we can obtain a detection algorithm whose detection constant $c = 3.5$ is strictly better than the best-known approximation ratio 4 for the problem.

Since our positive results all arise from bounding the integrality gap of certain restricted LP-formulations, we also study the integrality gap of a standard Cograph Deletion LP and prove it to be 4 (Theorem 4.9) using the probabilistic method. This provides a sharp contrast to the gap of 2.5 in our restricted setting.

**Our results: Lower bounds.** Our second set of results concerns lower bounds, showing that for certain combinations of $\Pi$ and $c$ there are no efficient algorithms for $c$-Essential detection for $\Pi$ under common complexity-theoretic hypotheses. In their work, Bumpus et al. [4] identified several problems $\Pi$ such as Perfect Deletion for which the detection problem is intractable for *all* choices of $c$. Their proofs are based on the hardness of FPT-approximation for Dominating Set [30]. The setting for our lower bounds is different. We analyze problems for which the detection task is polynomial-time solvable for *some* essentiality threshold $c$, and investigate whether polynomial-time algorithms can exist for a smaller threshold $c' < c$.

Our most prominent lower bound concerns the Directed Feedback Vertex Set problem (DFVS), which has attracted a lot of attention from the parameterized complexity community [8, 25]. It asks for a minimum vertex set $S$ of a *directed* graph $G$ for which $G-S$ is acyclic. Svensson proved that under the UGC [32], the problem is NP-hard to approximate to within any constant factor. Nevertheless, a polynomial-time algorithm for 2-Essential detection for DFVS was given by Bumpus et al. [4, Lemma 3.3]. We prove (Theorem 5.2) that the detection threshold 2 achieved by their algorithm is likely optimal: assuming the

UGC, the detection problem for $c' = 2 - \varepsilon$ is NP-hard for any $\varepsilon \in (0, 1]$. To prove this, we show that an algorithm with $c' = (2 - \varepsilon)$ would be able to distinguish instances with small solutions from instances with large solutions, while the hardness of approximation result cited above [32] show this task to be NP-hard under the UGC.

Apart from Directed Feedback Vertex Set, we provide two further lower bounds. For the Vertex Cover (VC) problem, an algorithm to detect 2-essential vertices is known [4]. Assuming the UGC, we prove (Theorem 5.6) that $(1.5 - \varepsilon)$-Essential detection for VC is NP-hard for all $\varepsilon \in (0, 0.5]$. A simple transformation then shows $(1.5 - \varepsilon)$-Detection for Vertex Multicut is also NP-hard under the UGC. These bounds leave a gap with respect to the thresholds of the current-best detection algorithms (2 and 3, respectively). We leave it to future work to close the gap.

**Organization.** The remainder of the paper is organized as follows. In Section 2 we give preliminaries on graphs and linear programming. Section 3 introduces our formalization for hitting set problems on graphs and provides the connection between integrality gaps and detection algorithms. Section 4 contains our positive results, followed by the negative results in Section 5. We conclude with some open problems in Section 6. Due to space limitations, the proofs of statements marked (★) are deferred to the full version of this paper [20].

## 2 Preliminaries

We consider finite simple graphs, some of which are directed. Directed graphs or objects defined on directed graphs will always be explicitly indicated as such. We use standard notation for graphs and parameterized algorithms. We re-iterate the most relevant terminology and notation, but anything not defined here may be found in the textbook by Cygan et al. [9] or in the previous work on essential vertices [4].

**Graph notation.** We let $P_\ell$ denote the path graph on $\ell$ vertices. The weight of a path in a vertex-weighted graph is the sum of the weights of the vertices on that path, including the endpoints. Given two disjoint vertex sets $S_1$ and $S_2$ in a *(directed)* graph $G$, we call a third vertex set $X \subseteq V(G)$ a *(directed)* $(S_1, S_2)$-separator in $G$ if it intersects every *(directed)* $(S_1, S_2)$-path in $G$. Note that $X$ may intersect $S_1$ and $S_2$. If $S_1$ or $S_2$ is a singleton set, we may write the single element of the set instead to obtain a $(v, S_2)$-separator for example. Menger's theorem relates the maximum number of pairwise vertex-disjoint paths between two (sets of) vertices to the minimum size of a separator between those two (sets of) vertices. We consider the following formulation of the theorem:

▶ **Theorem 2.1** ([31, Corollary 9.1a]). *Let $G$ be a directed graph and let $s, t \in V(G)$ be non-adjacent. Then the maximum number of internally vertex-disjoint directed $(s, t)$-paths is equal to the minimum size of a directed $(s, t)$-separator that does not include $s$ or $t$.*

A *fractional (directed)* $(S_1, S_2)$-separator is a weight function that assigns every vertex in a graph a non-negative weight such that every *(directed)* $(S_1, S_2)$-path has a weight of at least 1. The total weight of a fractional *(directed)* separator is the sum of all vertex weights.

**Linear programming notation.** We employ well-known concepts from linear programming and refer to a textbook for additional background [31]. A solution to a linear program (LP) where all variables are assigned an integral value is called an *integral* solution. As we only consider LPs with a one-to-one correspondence between its variables and the vertices in a

graph, integral solutions admit an alternative interpretation as vertex sets: the set of vertices whose corresponding variables are assigned a positive value. We use the interpretations of integral solutions as variable assignments or vertex sets interchangeably. We say that a minimization LP has an integrality gap of at most $c$ for some $c \in \mathbb{R}$ if the cost of an optimal integral solution is at most $c$ times the cost of an optimal fractional solution.

## 3 Essential vertices for Vertex Hitting Set problems

Our positive contributions all build upon the same result from Bumpus et al. [4, Theorem 4.1], which relates integrality gaps of certain LPs to the existence of $c$-Essential detection algorithms. A slightly generalized formulation of this can be found below as Theorem 3.1. First, we introduce the required background and notation.

The result indicates a strategy towards constructing a polynomial-time algorithm for $c$-Essential detection for $\Pi$ for a vertex selection problem $\Pi$, by considering a specific special variant of that problem, that we refer to as its $v$-*Avoiding* variant. It is defined almost identically to the original problem $\Pi$, but the input additionally contains a distinguished vertex $v \in V(G)$ which is explicitly forbidden to be part of a solution.

The original theorem from Bumpus et al. [4] is specifically targeted at $\mathcal{C}$-Deletion problems for *hereditary* graph classes $\mathcal{C}$. A graph class $\mathcal{C}$ is said to be hereditary when it exhibits the property that all induced subgraphs of a graph in $\mathcal{C}$ are again in $\mathcal{C}$. The corresponding $\mathcal{C}$-Deletion problem is that of finding a minimum size vertex set whose removal turns the input graph into one contained in $\mathcal{C}$. We remark however that the theorem holds for a broader collection of problems, namely those that can be described as *Vertex Hitting Set problems*. To define which problems qualify as a Vertex Hitting Set problem, we first recall the definition of the well-known optimization problem Hitting Set, on which our definition of Vertex Hitting Set problems is based.

| Hitting Set | |
|---|---|
| **Input:** | A universe $U$ and a collection $\mathcal{S} \subseteq 2^U$ of subsets of $U$. |
| **Feasible solution:** | A set $X \subseteq U$ such that $X \cap S \neq \emptyset$ for all $S \in \mathcal{S}$. |
| **Objective:** | Find a feasible solution of minimum size. |

We define *Vertex Hitting Set problems* as vertex selection problems that can be described as a special case of Hitting Set where the universe $U$ is the vertex set of the input graph and the collection $\mathcal{S}$ is encoded implicitly by the graph.

This definition in particular contains all $\mathcal{C}$-Deletion problems for hereditary graph classes $\mathcal{C}$. This is because every hereditary graph class can be characterized by a (possibly infinite) set of forbidden induced subgraphs. A graph $G$ is in $\mathcal{C}$ if and only if none of its induced subgraphs are isomorphic to a forbidden induced subgraph. Therefore, a $\mathcal{C}$-Deletion instance $G$ is equivalent to the Hitting Set instance $(V(G), \mathcal{S})$, with $\mathcal{S}$ being the collection of all the vertex subsets that induce a forbidden subgraph in $G$.

Now, as mentioned, the $v$-Avoiding variants of vertex selection problems are of particular interest. A useful consequence of considering Vertex Hitting Set problems as special cases of Hitting Set, is that this yields a well-defined canonical LP formulation for such problems that can easily be modified to describe their $v$-Avoiding variant. This LP formulation is based on the following standard LP for a Hitting Set instance $(U, \mathcal{S})$, which uses variables $x_u$ for every $u \in U$:

$$\begin{aligned} \text{minimize} \quad & \sum_{u \in U} x_u & \\ \text{subject to:} \quad & \sum_{u \in S} x_u \geq 1 & \text{for every } S \in \mathcal{S} \\ & 0 \leq x_u \leq 1 & \text{for every } u \in U \end{aligned}$$

To describe the $v$-Avoiding variant of a Vertex Hitting Set problem, this LP can simply be modified by adding the constraint $x_v = 0$. For a given Vertex Hitting Set problem $\Pi$, a graph $G$ and a vertex $v \in V(G)$, we denote the resulting LP as $\mathrm{LP}_\Pi(G, v)$.

Although the original theorem from Bumpus et al. [4] makes a statement about $\mathcal{C}$-deletion problems only, it is not too hard to see that this statement also holds for any other Vertex Hitting Set problem. We therefore present this result as the following slight generalization.

▶ **Theorem 3.1.** *Let $\Pi$ be a Vertex Hitting Set problem and let $c \in \mathbb{R}_{\geq 1}$. Then there exists a polynomial-time algorithm for $(c+1)$-Essential detection for $\Pi$ if the following two conditions are met:*

1. *For all $G$ and $v \in V(G)$, there is a polynomial-time separation oracle for $\mathrm{LP}_\Pi(G, v)$.*
2. *For all $G$ and $v \in V(G)$ for which $\{v\}$ solves $\Pi$ on $G$, the integrality gap of $\mathrm{LP}_\Pi(G, v)$ is at most $c$.*

This statement admits a proof that is almost identical to the proof by Bumpus et al. [4, Theorem 4.1]. At any point in that proof where the assumption is used that $\Pi$ is a $\mathcal{C}$-Deletion problem for some hereditary $\mathcal{C}$, this assumption may be replaced by the property that any superset of a solution to $\Pi$ is also a solution. This property is satisfied for every Vertex Hitting Set problem. Otherwise, no changes to the proof are required. We therefore refer the reader to this prior work for the details of the proof.

Many known results about the approximation of Hitting Set or about the integrality gap of Hitting Set LPs consider the restriction to $d$-Hitting Set. This is the problem obtained by requiring every $S \in \mathcal{S}$ in the input to be of size at most $d$ for some positive integer $d$. Both upper bounds and lower bounds are known for the integrality gaps of the standard LP describing $d$-Hitting Set instances. The standard LP is the linear program given above for the general Hitting Set problem.

It is well-known that this LP has an integrality gap of at most $d$ and that there exist instances for which this bound is tight. This result is for example mentioned as an exercise in a book on approximation algorithms [33, Exercise 15.3], framed from the equivalent perspective of the Set Cover problem.

## 4 Positive results

This section contains our positive results for essential vertex detection. For three different problems $\Pi$ and corresponding values of $c$, we provide polynomial-time algorithms for $c$-Essential detection for $\Pi$. The first two of these, being strongly related, are presented in Section 4.1. There, we provide $c$-Essential detection algorithms for Vertex Multicut and Directed Vertex Multicut with $c = 3$ and $c = 5$ respectively. Afterward, we provide a 3.5-Essential detection algorithm for the Cograph Deletion problem in Section 4.2.

## 4.1 Vertex Multicut

Our first two positive results concern the well-studied Vertex Multicut problem and its directed counterpart Directed Vertex Multicut. These are optimization problems defined as follows.

*(Directed)* Vertex Multicut

**Input:** A *(directed)* graph $G$ and a set of *(ordered)* vertex pairs $\mathcal{T} = \{(s_1, t_1), \ldots, (s_r, t_r)\}$ called the terminal pairs.

**Task:** Find a minimum size vertex set $S \subseteq V(G)$ such that there is no $(s_i, t_i) \in \mathcal{T}$ for which $G - S$ contains a *(directed)* $(s_i, t_i)$-path.

We start by observing that both problems are Vertex Hitting Set problems: if we let $\mathcal{P}_{\mathcal{T}}(G)$ be the collection of vertex subsets that form a *(directed)* $(s_i, t_i)$-path in $G$, then the *(Directed)* Vertex Multicut instance $(G, \mathcal{T})$ is equivalent to the Hitting Set instance $(V(G), \mathcal{P}_{\mathcal{T}}(G))$. This interpretation of the problems as special cases of Hitting Set is also captured by the standard LP formulations of the problems, on which the $v$-Avoiding LP below is based:

$$\begin{aligned} \text{minimize} \quad & \sum_{u \in V(G)} x_u & \\ \text{subject to:} \quad & \sum_{u \in V(P)} x_u \geq 1 & \text{for every } \textit{(directed)} \text{ path } P \text{ from some } s_i \text{ to } t_i \\ & x_v = 0 & \\ & 0 \leq x_u \leq 1 & \text{for } u \in V(G) \end{aligned}$$

The set of constraints in this LP formulation not only depends on the structure of the input graph $G$, but also on the set $\mathcal{T}$ of terminal pairs. Hence, we denote the LP above as $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ for undirected $G$ or as $\mathrm{LP}_{\mathrm{DVM}}(G, \mathcal{T}, v)$ for directed $G$. The standard LP formulations of Vertex Multicut and Directed Vertex Multicut are obtained by simply removing the constraint $x_v = 0$.

**The undirected case.** We start with the undirected version of the problem and show in Lemma 4.1 that $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ has an integrality gap of at most 2 for all Vertex Multicut instances $(G, \mathcal{T})$ where $v \in V(G)$ is such that $\{v\}$ is a solution. This bound yields a polynomial-time algorithm for 3-Essential detection for Vertex Multicut as presented in Theorem 4.2.

▶ **Lemma 4.1.** *Let $(G, \mathcal{T})$ be a Vertex Multicut instance with some $v \in V(G)$ such that $\{v\}$ is a solution for this instance. Then $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ has an integrality gap of at most 2.*

**Proof.** Let $\mathbf{x} = (x_u)_{u \in V(G)}$ be an optimal solution to $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ and let $z = \sum_{u \in V(G)} x_u$ be its value. If we interpret the values of $x_u$, as given by $\mathbf{x}$, as vertex weights, then by definition of the LP, all $(s_i, t_i)$-paths have weight at least 1 for all $\{s_i, t_i\} \in \mathcal{T}$. Moreover, all such paths must pass through $v$ because $\{v\}$ is a solution, so we know for every $\{s_i, t_i\} \in \mathcal{T}$ that all $(s_i, v)$-paths or all $(t_i, v)$-paths (or both) have weight at least $\frac{1}{2}$.

We proceed by stating a reformulation of this property. Let $D \subseteq V(G)$ be the set of all vertices $u$ such that every $(u, v)$-path has weight at least $\frac{1}{2}$. Then, the above property can also be described as follows: for every $\{s_i, t_i\} \in \mathcal{T}$, at least one of $s_i$ and $t_i$ is in $D$.

Using this alternate formulation, it follows that every $(v, D)$-separator $X$ is also a valid solution to the given Vertex Multicut instance. To see this, consider an arbitrary $(s_i, t_i)$-path $P$ for some arbitrary $\{s_i, t_i\} \in \mathcal{T}$. Since $\{v\}$ is a solution, $P$ intersects $v$. If $s_i \in D$, then the fact that $X$ is a $(v, D)$-separator implies that $X$ intersects the subpath of $P$ between $v$ and $s_i$. The same holds for $t_i$. Since at least one of $s_i$ and $t_i$ is in $D$, it follows that $X$ must intersect $P$. Because $P$ was an arbitrary $(s_i, t_i)$-path for an arbitrary terminal pair $\{s_i, t_i\}$, $X$ hits all such paths and therefore it is a vertex multicut.

Now to prove that $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ has an integrality gap of at most 2, it suffices to show that there exists a $(v, D)$-separator $X \subseteq V(G)$ of size at most $2z$ that does not contain $v$. To see that this is indeed the case, we start by constructing a fractional $(v, D)$-separator $f \colon V(G) \to \mathbb{R}$ of weight at most $2z$ and with $f(v) = 0$. We obtain $f$ by simply doubling the values given by $\mathbf{x}$, i.e.: $f(u) := 2x_u$ for all $u \in V(G)$. This step is inspired by a proof from Golovin, Nagarajan, and Singh that shows an upper bound on the integrality gap of a Multicut variant in trees [15].

We observe that indeed $f(v) = 2 \cdot x_v = 0$, since $\mathbf{x}$ is a solution to $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$, which requires that $x_v = 0$. Furthermore, $D$ was constructed such that all paths from $v$ to a vertex in $D$ have a weight of at least $\frac{1}{2}$ under the vertex weights as given by $\mathbf{x}$. Hence, under the doubled weights of $f$, all such paths have a weight of at least 1, witnessing that $f$ is in fact a fractional $(v, D)$-separator.

The final step of the proof is now to show that the existence of this *fractional* $(v, D)$-separator of weight $2z$ implies the existence of an *integral* $(v, D)$-separator of size at most $2z$ that does not contain $v$. To do so, we use Menger's theorem on the auxiliary directed graph $G'$ obtained from $G$ by turning all undirected edges into bidirected edges, while adding a sink node $t$ with incoming edges from all vertices in $D$.

Consider a maximum collection $\mathcal{P}$ of internally vertex-disjoint directed $(v, t)$-paths in $G'$. Let $X \subseteq V(G') \setminus \{v, t\}$ be a directed $(v, t)$-separator in $G'$ of size $|\mathcal{P}|$, whose existence is guaranteed by Theorem 2.1. The construction of $G'$ ensures that $X$ is a $(v, D)$-separator in $G$ that does not contain $v$, and therefore corresponds to an integral solution to $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$. To bound the integrality gap by 2, it therefore suffices to prove that $|\mathcal{P}| = |X| \leq 2z$.

For each $(v, t)$-path $P \in \mathcal{P}$ in $G'$, the prefix obtained by omitting its endpoint $t$ yields a $(v, D)$-path in $G$. Since $f$ is a fractional $(v, D)$-separator, it must assign every such prefix of $P \in \mathcal{P}$ a weight of at least 1. Because $f(v) = 0$ and because the paths in $\mathcal{P}$ are internally vertex-disjoint, we find that the total weight of $f$ must be at least $|\mathcal{P}| = |X|$. Since the weight of $f$ is at most $2z$, we find that $|\mathcal{P}| = |X| \leq 2z$. This concludes the proof. ◀

We can even construct Vertex Multicut instances $(G, \mathcal{T})$ that are solved by some $\{v\} \subseteq V(G)$ for which the integrality gap of $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ is arbitrarily close to 2, showing that the bound in Lemma 4.1 is tight. To construct such an instance, let $G$ be a (large) star graph, let $v \in V(G)$ be its center and let $\mathcal{T} = \binom{V(G) \setminus \{v\}}{2}$. Clearly, $\{v\}$ is a solution to the Vertex Multicut instance $(G, \mathcal{T})$.

To determine the integrality gap of $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$, we first note that any solution to the Vertex Multicut instance that avoids $v$ must, at least, include all but one of the leaves from $G$. Any such set is indeed a solution, which shows that the smallest integral solution to $\mathrm{LP}_{\mathrm{VM}}(G, \mathcal{T}, v)$ has value $|V(G)| - 2$. A smaller fractional solution to the program may be obtained by assigning every leaf of $G$ a value of $\frac{1}{2}$, which would yield a solution with a total value of $\frac{1}{2} \cdot (|V(G)| - 1)$. Observe that such a construction of $G$, $\mathcal{T}$, and $v$ can be used to get an LP with an integrality gap arbitrarily close to 2 by having the star graph $G$ be arbitrarily large.

Regardless of the bound on the integrality gap being tight, Lemma 4.1 and Theorem 3.1 combine to prove the following result.

▶ **Theorem 4.2.** (★) *There exists a polynomial-time algorithm for* 3-Essential detection for Vertex Multicut.

The algorithm to detect 3-essential vertices leads in a black-box fashion to a search-space reduction guarantee for the current-best algorithm for solving Vertex Multicut due to Marx and Razgon [26]. This follows from a result of Bumpus et al. [4, Theorem 5.1] (cf. [20, Theorem A.1]). While they originally stated their connection between essential detection and search-space reduction for $\mathcal{C}$-Deletion problems, it is easy to see that the same proof applies for any Vertex Hitting Set problem: the only property of $\mathcal{C}$-Deletion that is used in their proof is that for any vertex set $X \subseteq V(G)$, a vertex set $Y \subseteq V(G - X)$ is a solution to $G - X$ if and only if $X \cup Y$ is a solution to $G$; this property holds for any Vertex Hitting Set problem.

▶ **Corollary 4.3.** *There is an algorithm that, given a Vertex Multicut instance $(G, \mathcal{T})$ on $n$ vertices, outputs an optimal solution in time $2^{\mathcal{O}(\ell^3)} \cdot n^{\mathcal{O}(1)}$, where $\ell$ is the number of vertices in an optimal solution that are* not *3-essential.*

**The directed case.** Keeping in mind the techniques used to prove Lemma 4.1, we proceed to the next problem: Directed Vertex Multicut. By similar arguments, we find the $v$-Avoiding LP of this problem to have a bounded integrality gap as well. However, adaptations to these arguments are required to take the directions of edges into consideration, yielding a higher bound on the integrality gap of the directed version of the problem.

▶ **Lemma 4.4.** (★) *Let $(G, \mathcal{T})$ be a Directed Vertex Multicut instance with some $v \in V(G)$ such that $\{v\}$ is a solution for it. Then $\mathrm{LP}_{\mathrm{DVM}}(G, \mathcal{T}, v)$ has an integrality gap of at most* 4.

Similar to the undirected setting, this upper bound on the integrality gap leads to the following algorithmic result when combined with Theorem 3.1.

▶ **Theorem 4.5.** *There exists a polynomial-time algorithm for* 5-Essential detection for Directed Vertex Multicut.

This statement admits a proof that is almost identical to the proof of Theorem 4.2, since the shortest-path algorithm that provides the separation oracle of the Vertex Multicut LP can also take directed graphs as input.

## 4.2 Cograph Deletion

Our next positive result concerns the Cograph Deletion problem. As this is a specific case of $\mathcal{C}$-Deletion, this is more in line with the original research direction for $c$-Essential detection introduced by Bumpus et al. [4], where a framework was built around $\mathcal{C}$-Deletion problems. The Cograph Deletion problem is defined as follows.

| Cograph Deletion | |
|---|---|
| **Input:** | An undirected graph $G$. |
| **Task:** | Find a minimum size set $S \subseteq V(G)$ such that $G - S$ is a cograph (i.e.: $G - S$ does not contain a path on 4 vertices as an induced subgraph). |

We start by observing that the Cograph Deletion problem is a Vertex Hitting Set problem: if we let $\mathcal{P}_4(G)$ be the collection of vertex subsets that induce a $P_4$ in $G$, then the Cograph Deletion instance $G$ is equivalent to the Hitting Set instance $(V(G), \mathcal{P}_4(G))$. Again, motivated by Theorem 3.1, we study the $v$-Avoiding LP for this problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{u \in V(G)} x_u \\ \text{subject to:} \quad & \sum_{u \in V(H)} x_u \geq 1 \quad \text{for every induced subgraph } H \text{ of } G \text{ isomorphic to } P_4 \\ & x_v = 0 \\ & 0 \leq x_u \leq 1 \quad \text{for } u \in V(G) \end{aligned}$$

For a given graph $G$ and vertex $v \in V(G)$, we denote the LP above as $\mathrm{LP}_{\mathrm{CD}}(G, v)$. Whenever $v$ is such that $G - v$ is a cograph, the resulting LP admits a simple upper bound on the integrality gap. This bound is derived from the observation that the $v$-Avoiding Cograph Deletion problem is a special case of 3-Hitting Set: the vertex sets to be hit in the problem are the triplets of vertices that, together with $v$, induce a $P_4$ in $G$. As the natural LP describing 3-Hitting Set has an integrality gap of at most 3, it follows that the natural LP formulation of $v$-Avoiding Cograph Deletion, to which the above LP is equivalent, also has an integrality gap of at most 3.

This section is dedicated to proving a stronger result than this trivial bound. We prove that, whenever $v$ is such that $G - v$ is a cograph, $\mathrm{LP}_{\mathrm{CD}}(G, v)$ has an integrality gap of at most 2.5. To prove this, we use a method inspired by iterative rounding [19], where an approximate integral solution can be obtained by solving the LP, picking all vertices that receive a large enough value, updating the LP to no longer contain these vertices and repeating these steps until a solution is found.

For our purposes, we consider values of at least 0.4 to be "large enough". However, we will see that an extension to the original method is required since $\mathrm{LP}_{\mathrm{CD}}(G, v)$ is not guaranteed to always have an optimal solution that assigns at least one vertex a value of $\geq 0.4$. This issue is reflected in the inductive proof below by having the step case split into two subcases. The first of these deals with the standard iterative rounding setup, while the second subcase deals with the possibility of an optimal solution not assigning any vertex a large value.

▶ **Lemma 4.6.** *Let $G$ be a graph and let $v \in V(G)$ be such that $G - v$ is a cograph. Then $\mathrm{LP}_{\mathrm{CD}}(G, v)$ has an integrality gap of at most 2.5.*

**Proof.** We prove the statement by induction on the value of an optimal fractional solution to the linear program.

First, consider as base case that $\mathrm{LP}_{\mathrm{CD}}(G, v)$ has an optimal fractional solution of value 0. Then this solution is the all-zero solution. This is also an integral optimum solution to the program, so the integrality gap of the program is 1 and the claim holds.

Next, let $\mathbf{x} = (x_u)_{u \in V(G)}$ be an optimal solution to $\mathrm{LP}_{\mathrm{CD}}(G, v)$, let $z = \sum_{u \in V(G)} x_u$ be its value and let $V_{\geq 0.4} \subseteq V(G)$ be the set of vertices that are assigned a value of at least 0.4 in this solution. We distinguish two cases.

**Case 1.** Suppose $V_{\geq 0.4} \neq \emptyset$. Consider the pair $(G - V_{\geq 0.4}, v)$ and note that $(G - V_{\geq 0.4}) - v$, being an induced subgraph of $G - v$, is a cograph. Also note that the restriction of $\mathbf{x}$ to $G - V_{\geq 0.4}$ is a feasible solution to $\mathrm{LP}_{\mathrm{CD}}(G - V_{\geq 0.4}, v)$. This solution has a value of $z - \sum_{u \in V_{\geq 0.4}} x_u \leq z - 0.4 \cdot |V_{\geq 0.4}|$, which is strictly smaller than $z$ by the assumption that $V_{\geq 0.4} \neq \emptyset$. If we let $z_{\mathrm{res}}$ be the value of an optimal solution to $\mathrm{LP}_{\mathrm{CD}}(G - V_{\geq 0.4}, v)$, then this implies that $z_{\mathrm{res}} \leq z - 0.4|V_{\geq 0.4}| < z$ as well.

Then, by the induction hypothesis, an integral solution $V_{\text{res}}$ to $\text{LP}_{\text{CD}}(G - V_{\geq 0.4}, v)$ with $|V_{\text{res}}| \leq 2.5 z_{\text{res}}$ exists. To prove that $\text{LP}_{\text{CD}}(G, v)$ has an integrality gap of at most 2.5, we proceed by showing that $V_{\text{res}} \cup V_{\geq 0.4}$ is an integral solution to $\text{LP}_{\text{CD}}(G, v)$ with value at most $2.5z$. We start by arguing that $V_{\text{res}} \cup V_{\geq 0.4}$ is a valid integral solution.

First note that neither $V_{\text{res}}$ nor $V_{\geq 0.4}$ contains $v$ since both $\text{LP}_{\text{CD}}(G - V_{\geq 0.4}, v)$ and $\text{LP}_{\text{CD}}(G, v)$ require $x_v = 0$. Therefore, the union of these two sets also does not contain $v$. Secondly, note that $V_{\text{res}}$ (by construction of $\text{LP}_{\text{CD}}(G - V_{\geq 0.4}, v)$) contains a vertex from every induced $P_4$ in $G$ that does not already have a vertex in $V_{\geq 0.4}$. As such, $V_{\text{res}} \cup V_{\geq 0.4}$ contains a vertex from every induced $P_4$ in $G$, which makes it a feasible solution to $\text{LP}_{\text{CD}}(G, v)$.

Knowing this, it remains to prove that $V_{\text{res}} \cup V_{\geq 0.4}$ has size at most $2.5z$. Recall that we derived $z_{\text{res}} \leq z - 0.4 \cdot |V_{\geq 0.4}|$. We can use this inequality to make the following derivation:

$$\begin{aligned} |V_{\text{res}} \cup V_{\geq 0.4}| \leq |V_{\text{res}}| + |V_{\geq 0.4}| \;&\leq\; 2.5 z_{\text{res}} + |V_{\geq 0.4}| && \text{by definition of } V_{\text{res}} \\ &\leq 2.5\,(z - 0.4 \cdot |V_{\geq 0.4}|) + |V_{\geq 0.4}| && \text{by the above inequality} \\ &= 2.5z - |V_{\geq 0.4}| + |V_{\geq 0.4}| \;=\; 2.5z && \text{since } 2.5 \cdot 0.4 = 1 \end{aligned}$$

**Case 2.** Suppose $V_{\geq 0.4} = \emptyset$. Let $V^* \subseteq V(G) \setminus \{v\}$ be the set of vertices other than $v$ that are part of at least one induced $P_4$ in $G$. To prove that $\text{LP}_{\text{CD}}(G, v)$ has an integrality gap of at most 2.5, we show that the smaller set of $V^* \cap N_G(v)$ and $V^* \setminus N_G(v)$ is a solution to the program with size at most $2.5z$.

We start by proving that $V^* \cap N_G(v)$ and $V^* \setminus N_G(v)$ are both feasible solutions to $\text{LP}_{\text{CD}}(G, v)$. We do so using an observation about the structure of the graph $P_4$. Observe that this graph has the property that each vertex has at least one neighbor and at least one non-neighbor. Since $v$ is part of every induced $P_4$ in $G$ by assumption, this means that every induced $P_4$ in $G$ contains both a neighbor and a non-neighbor of $v$.

The above observation implies that $V^* \cap N_G(v)$ and $V^* \setminus N_G(v)$ both intersect all induced subgraphs of $G$ isomorphic to $P_4$. Hence, both of these sets are feasible solutions to $\text{LP}_{\text{CD}}(G, v)$. It remains to prove that the smaller of the two sets has a size of at most $2.5z$.

Since $V^* \cap N_G(v)$ and $V^* \setminus N_G(v)$ form a partition of $V^*$ into two parts, the smaller of the two will always be of size at most $|V^*|/2$. Therefore, it suffices to show that $|V^*|/2 \leq 2.5z$. To prove this, we start by showing that the assumption that $V_{\geq 0.4} = \emptyset$ implies that $x_w \geq 0.2$ for all vertices $w \in V^*$.

We prove this property by contradiction, so suppose there is some vertex $w \in V(G) \setminus \{v\}$ that is part of an induced $P_4$, but which has $x_w < 0.2$. Let $H$ be an induced subgraph of $G$ that is isomorphic to $P_4$ and with $w \in V(H)$. Because $G - v$ is a cograph, $v$ is contained in every induced $P_4$ and in particular $v \in V(H)$. By definition of $\text{LP}_{\text{CD}}(G, v)$, we have $x_v = 0$. By the assumption that $V_{\geq 0.4} = \emptyset$, the two vertices in $V(H) \setminus \{w, v\}$ have value at most 0.4, so $\sum_{u \in V(H)} x_u < 1$, which contradicts the validity of $\mathbf{x}$.

Knowing that $x_w \geq 0.2$ for all $w \in V^*$, it follows that $z = \sum_{u \in V(G)} x_u \geq 0.2|V^*|$. Rewriting this inequality, we obtain $|V^*|/2 \leq 2.5z$. ◀

At the moment, we are not aware of any examples of pairs $(G, v)$ where $G - v$ is a cograph and for which $\text{LP}_{\text{CD}}(G, v)$ has an integrality gap of 2.5. Therefore, the bound above does not have to be tight and the integrality gap of such programs may even be as small as 2. However, there do exist pairs $(G, v)$ where $G - v$ is a cograph and for which $\text{LP}_{\text{CD}}(G, v)$ has an integrality gap arbitrarily close to 2.

Such a pair $(G, v)$ may be obtained by constructing $G$ as the union of $m$ disjoint edges and adding the vertex $v$ to it which is adjacent to exactly one endpoint of each of these $m$ edges. Then, any integral solution to $\text{LP}_{\text{CD}}(G, v)$ must include, at least, one endpoint from all but one of the original $m$ edges. Any such set of vertices is in fact a feasible integral solution, so a smallest integral solution has size $m - 1$.

An optimal fractional solution may be obtained by assigning all $m$ neighbors of $v$ a value of 0.5, which yields a total value of $m/2$. Hence, the integrality gap of $\mathrm{LP}_{\mathrm{CD}}(G, v)$ is $\frac{m-1}{m/2} = 2 \cdot \frac{m-1}{m}$, which can be arbitrarily close to 2 for arbitrarily large $m$.

Like earlier, the upper bound on the integrality gap shown in Lemma 4.6 leads to the following algorithmic result.

▶ **Theorem 4.7.** (★) *There exists a polynomial-time algorithm for* 3.5*-Essential detection for Cograph Deletion.*

The algorithm to detect 3.5-essential vertices leads to a search-space reduction guarantee for the current-best parameterized algorithm for Cograph Deletion [27] via Theorem 5.1 by Bumpus et al. [4].

▶ **Corollary 4.8.** *There is an algorithm that, given a Cograph Deletion instance $G$ on $n$ vertices, outputs an optimal solution in time $3.115^{\ell} \cdot n^{\mathcal{O}(1)}$, where $\ell$ is the number of vertices in an optimal solution that are* not 3.5*-essential.*

In the full version of this paper [20, Section 4.2.1] we contrast the integrality gap of 2.5 for the $v$-avoiding version of Cograph Deletion to the standard version for the problem, for which we provide the following lower bound using the probabilistic method.

▶ **Theorem 4.9.** (★) *For all $\varepsilon > 0$, the integrality gap of the standard Cograph Deletion LP is larger than $4 - \varepsilon$.*

## 5 Hardness results

In this section, we show two main hardness results regarding essential detection algorithms. The first of these concerns Directed Feedback Vertex Set (DFVS). The objective in this problem is to find a smallest vertex set $S$ in a directed input graph $G$ such that $G - S$ is acyclic. We slightly abuse notation by using the acronym DFVS to denote both a (not necessarily optimal) solution to a given input and the name of the problem itself. Additionally, we let $\mathrm{DFVS}(G)$ denote the size of a smallest DFVS in $G$. The hardness result obtained for DFVS can be extended to Directed Vertex Multicut. The second result concerns Vertex Cover (VC) and it can be extended to other vertex hitting set problems on undirected graphs, including Vertex Multicut.

Our results are based on the hardness assumption posed by the Unique Games Conjecture (UGC) [21]. Although the conjecture has remained open since its introduction in 2002, many conditional hardness results in the area of approximation algorithms follow from it. Before stating our first new hardness result, we mention the known result it is derived from, which itself is an implication of the UGC. By the nature of the UGC, many results derived from it show the conditional hardness of distinguishing between two types of problem inputs: one with a very small solution and one with a very large solution. Indeed, we derive our hardness from one such result due to Svensson [32, Theorem 1.1] that implies the following.

▶ **Lemma 5.1.** (★) *Assuming the UGC, the following problem is NP-hard for any integer $r \geq 2$ and sufficiently small constant $\delta > 0$. Given a directed $n$-vertex graph $G$, distinguish between the following two cases:*

- $\mathrm{DFVS}(G) \leq \left(\frac{1-\delta}{r} + \delta\right) n$
- $\mathrm{DFVS}(G) \geq (1 - \delta)n$

We use this formulation to prove the following.

▶ **Theorem 5.2.** *Assuming the UGC, $(2-\varepsilon)$-Essential detection for DFVS is NP-hard for any $\varepsilon \in (0, 1]$.*

**Proof.** Let $\varepsilon \in (0, 1]$ be given. We can assume w.l.o.g. that $\frac{2}{\varepsilon}$ is integral. If not, we could consider some $\varepsilon' < \varepsilon$ such that $\frac{2}{\varepsilon'}$ is integer and prove hardness for $(2-\varepsilon')$-essential detection. As a $(2-\varepsilon)$-essential detection algorithm is also a valid algorithm for $(2-\varepsilon')$-essential detection, this would imply the hardness of $(2-\varepsilon)$-essential detection as well.

Now, we use Lemma 5.1 as a starting point for hardness. To do so, let $G$ be an arbitrary directed graph on $n$ vertices. To use Lemma 5.1, we show how to reduce $G$ into a directed graph $G'$, such that solving $(2-\varepsilon)$-Essential detection for DFVS on $G'$ allows us to distinguish between $\text{DFVS}(G) \leq \left(\frac{1-\delta}{r} + \delta\right) n$ and $\text{DFVS}(G) \geq (1-\delta)n$ for some integer $r \geq 2$ and arbitrarily small $\delta > 0$. We assume w.l.o.g. that $n \cdot \varepsilon/2$ is integer. If not, we could consider the graph obtained by having $2/\varepsilon$ independent copies of $G$ instead, as the minimum size of a DFVS relative to the total graph size would be the same. We proceed by explaining the reduction, after which we prove its correctness.

Our reduction starts with the directed graph $G$ and depends on the value of $\varepsilon$. The full version of this paper contains a visual example [20, Figure 1]. We start the construction of $G'$ as a copy of $G$. To avoid confusion between vertices in $G$ and $G'$, we denote the current vertex set of $G'$ as $P$. Next, we expand the graph with two additional sets of vertices $Q_{\text{in}}$ and $Q_{\text{out}}$. These sets each consist of $m := (1 - \frac{\varepsilon}{2})n$ vertices, which is integer by our assumptions on $n$ and $\varepsilon$. We denote the vertices of $Q_{\text{in}}$ as $q_1, \ldots, q_m$ and the vertices of $Q_{\text{out}}$ as $q'_1, \ldots, q'_m$. We define $Q := Q_{\text{in}} \cup Q_{\text{out}}$.

We complete the construction of $G'$ by adding more arcs to it. For every $i \in [m]$, we add the arc $(q_i, q'_i)$. For every $p \in P$ and $q_i \in Q_{\text{in}}$, we add the arc $(p, q_i)$. For every $p \in P$ and $q'_i \in Q_{\text{out}}$, we add the arc $(q'_i, p)$. This completes the construction of $G'$. Observe that it ensures that $(p, q_i, q'_i)$ is a directed cycle for every $p \in P$ and $i \in [m]$.

To prove the correctness of this reduction, we show that the output of an algorithm for $(2-\varepsilon)$-Essential detection for DFVS on $G'$ can be used as subroutine to distinguish between $\text{DFVS}(G) \leq \left(\frac{1-\delta}{r} + \delta\right) n$ and $\text{DFVS}(G) \geq (1-\delta)n$ for some integer $r \geq 2$ and arbitrarily small $\delta > 0$. In particular, we show that this is possible for $r = \frac{4}{\varepsilon}$, which is integer by the assumption that $\frac{2}{\varepsilon}$ is integer. From now on, we fix $r = \frac{4}{\varepsilon}$ and $\delta > 0$ to be arbitrarily small so that $\delta \leq \frac{\varepsilon}{4}$ in particular.

Now, suppose that an algorithm for $(2-\varepsilon)$-Essential detection for DFVS exists and let $S \subseteq V(G')$ be its output when run on $G'$ with $k$ set to $n$. (Recall, $k$ represents a guess for (an upper bound) of the size of an optimal solution in $G'$. In this setting, that would be a guess for the size of a minimum size DFVS in $G'$.) We show that the following two implications hold:

▷ Claim 5.3. (★) If $\text{DFVS}(G) \leq \left(\frac{1-\delta}{r} + \delta\right) n$, then $|S| < n$.

▷ Claim 5.4. (★) If $\text{DFVS}(G) \geq (1-\delta)n$, then $|S| = n$.

Then, simply checking the size of the output set $S$ suffices to distinguish between $\text{DFVS}(G) \leq \left(\frac{1-\delta}{r} + \delta\right) n$ and $\text{DFVS}(G) \geq (1-\delta)n$. From Lemma 5.1, we know that this distinction is NP-hard to make under the UGC, meaning that $(2-\varepsilon)$-Essential detection for DFVS is also NP-hard when assuming the UGC. To prove Theorem 5.2, it therefore suffices to prove Claim 5.3 and Claim 5.4. The full proofs can be found in the full version.

Proof sketch for Claim 5.3. Let $X$ be a smallest DFVS in $G$. It follows from the construction of $G'$ that the set $X \cup Q_{\text{in}}$ is a DFVS in $G'$. Its size is strictly smaller than $n$, which follows from our choice of $r$ and by $\delta$ being arbitrarily small. Since we invoke the algorithm for $(2-\varepsilon)$-Essential detection for DFVS with $k = n$, by Property (G1) the set $S$ must be a subset of some smallest DFVS in $G'$. This implies that $|S| < n$, proving the claim. ◁

Proof sketch for Claim 5.4. Suppose that $\mathrm{DFVS}(G) \geq (1-\delta)n$. By construction of $G'$, the set $P$ is a DFVS in $G'$ so that $\mathrm{DFVS}(G') \leq |P| = n$. We aim to show that $P$ is in fact the unique smallest DFVS in $G'$, by showing that all vertices in $P$ are $(2-\varepsilon)$-essential in $G'$ and therefore cannot be avoided in any $(2-\varepsilon)$-approximate solution, let alone in an optimal solution. Assuming the $(2-\varepsilon)$-essentiality of the vertices in $P$, it follows from Property (G2) that the set $S$ (the output of running a $(2-\varepsilon)$-Essential detection for DFVS algorithm on $G'$ with $k$ set to $n$) must contain all of $P$, so $|S| \geq n$. By Property (G1), no other vertices can be in $S$, so $|S| = n$.

To establish the claim, it therefore suffices to prove that all vertices of $P$ are $(2-\varepsilon)$-essential. By construction of $G'$, each vertex $p \in P$ forms a directed cycle with each of the $m$ pairs $(q_i, q_i')$ in $Q$. Any solution avoiding $p$ therefore contains at least $m$ vertices from $Q$, but also contains at least $\mathrm{DFVS}(G) \geq (1-\delta)n$ vertices from $P$ to hit all cycles of $G'[P] = G$. Our choice of $m$ and $\delta$ ensure $m + (1-\delta)n \geq (2-\varepsilon)n \geq (2-\varepsilon)\mathrm{DFVS}(G')$. ◁

This concludes the proof of Theorem 5.2. ◀

The lower bound of Theorem 5.2 yields an analogous lower bound for Directed Vertex Multicut, since the set of solutions for Directed Feedback Vertex Set on a graph $G$ equals the set of solutions to the Directed Vertex Multicut instance obtained from $G$ by introducing a terminal pair $(u_2, u_1)$ for every arc $(u_1, u_2)$ of $G$.

▶ **Corollary 5.5.** *Assuming the UGC, $(2-\varepsilon)$-Essential detection for Directed Vertex Multicut is NP-hard for any $\varepsilon > 0$.*

By applying the proof technique above, but starting from a result about hardness of approximation for $d$-Hitting Set [22], we prove the following lower bound for Vertex Cover. It implies the same lower bound for Undirected Vertex Multicut.

▶ **Theorem 5.6.** (★) *Assuming the UGC, $(1.5-\varepsilon)$-Essential detection for VC is NP-hard for any $\varepsilon \in (0, 0.5]$.*

▶ **Corollary 5.7.** (★) *Assuming the UGC, $(1.5-\varepsilon)$-Essential detection for Vertex Multicut is NP-hard for any $\varepsilon \in (0, 0.5]$.*

## 6 Conclusion and discussion

We revisited the framework of search-space reduction via the detection of essential vertices. The improved running-time guarantees for fixed-parameter tractable algorithms that result from our detection algorithms give insight into which types of inputs of NP-hard vertex hitting set problems can be solved efficiently and optimally: not only the inputs whose total solution size is small, but also those in which all but a small number of vertices of an optimal solution are essential. Our detection algorithms arise by analyzing the integrality gap for the $v$-Avoiding version of the corresponding LP-relaxation, which only has to be analyzed for inputs in which $\{v\}$ is a singleton solution. Our results show that the integrality gaps in this setting are much smaller than for the standard linear program of the hitting set formulation.

For Directed Feedback Vertex Set, our lower bound shows that the polynomial-time algorithm that detects 2-essential vertices is best-possible under the UGC. For Vertex Cover and Vertex Multicut, our lower bounds do not match the existing upper bounds. It would be interesting to close these gaps.

Our positive results rely on standard linear programming formulations of the associated hitting set problems. In several scenarios, algorithms based on the standard linear program can be improved by considering stronger relaxations such as those derived from the Sherali-Adams hierarchy or Lasserre-hierachy (cf. [23]). For example, Aprile, Drescher, Fiorini, and

Huynh [2] proved that for the Cluster Vertex Deletion problem (which asks to hit all the induced $P_3$ subgraphs) the integrality gap of the standard LP-formulation is 3, but decreases to 2.5 using the first round of the Sherali-Adams hierarchy. Applying $(1/\varepsilon)^{\mathcal{O}(1)}$ rounds further decreases the gap to $2+\varepsilon$. Can such hierarchies lead to better algorithms for $c$-Essential detection?

So far, the notion of $c$-essentiality has been explored for vertex hitting set problems on graphs. For other optimization problems whose solutions are subsets of objects (for example, edge subsets, or subsets of tasks in a scheduling problem) one can define $c$-essential objects as those contained in all $c$-approximate solutions. Does this notion have interesting applications for problems that are not about graphs?

## References


1 Tobias Achterberg, Robert E. Bixby, Zonghao Gu, Edward Rothberg, and Dieter Weninger. Presolve reductions in mixed integer programming. Technical Report 16-44, ZIB, Takustr.7, 14195 Berlin, 2016. URL: http://nbn-resolving.de/urn:nbn:de:0297-zib-60370.

2 Manuel Aprile, Matthew Drescher, Samuel Fiorini, and Tony Huynh. A tight approximation algorithm for the cluster vertex deletion problem. *Math. Program.*, 197(2):1069–1091, 2023. doi:10.1007/S10107-021-01744-W.

3 Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. Multicut is FPT. *SIAM J. Comput.*, 47(1):166–207, 2018. doi:10.1137/140961808.

4 Benjamin Merlin Bumpus, Bart M. P. Jansen, and Jari J. H. de Kroon. Search-space reduction via essential vertices. In Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman, editors, *Proceedings of the 30th Annual European Symposium on Algorithms, ESA 2022*, volume 244 of *LIPIcs*, pages 30:1–30:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ESA.2022.30.

5 Leizhen Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Inf. Process. Lett.*, 58(4):171–176, 1996. doi:10.1016/0020-0190(96)00050-6.

6 Leizhen Cai. Parameterized complexity of vertex colouring. *Discret. Appl. Math.*, 127(3):415–429, 2003. doi:10.1016/S0166-218X(02)00242-1.

7 Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. *Comput. Complex.*, 15(2):94–114, 2006. doi:10.1007/S00037-006-0210-9.

8 Jianer Chen, Yang Liu, Songjian Lu, Barry O'Sullivan, and Igor Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *J. ACM*, 55(5):21:1–21:19, 2008. doi:10.1145/1411509.1411511.

9 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 Rodney G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer Publishing Company, Incorporated, 2012.

11 Matthew Drescher. *Two Approaches to Approximation Algorithms for Vertex Deletion Problems*. PhD thesis, Université libre de bruxelles, 2021. URL: https://knavely.github.io/knavely.gitub.io/thesis.pdf.

12 Michael R. Fellows. The lost continent of polynomial time: Preprocessing and kernelization. In *Proceedings of the 2nd International Workshop on Parameterized and Exact Computation, IWPEC 2006*, pages 276–277, 2006. doi:10.1007/11847250_25.

13 Henning Fernau. A top-down approach to search-trees: Improved algorithmics for 3-hitting set. *Algorithmica*, 57(1):97–118, 2010. doi:10.1007/S00453-008-9199-6.

14 Fedor Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019.

15 Daniel Golovin, Viswanath Nagarajan, and Mohit Singh. Approximating the $k$-multicut problem. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006*, pages 621–630. ACM Press, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109625.

**16** Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Automated generation of search tree algorithms for hard graph modification problems. *Algorithmica*, 39(4):321–347, 2004. doi:10.1007/S00453-004-1090-5.

**17** Meike Hatzel, Lars Jaffke, Paloma T. Lima, Tomás Masarík, Marcin Pilipczuk, Roohani Sharma, and Manuel Sorge. Fixed-parameter tractability of DIRECTED MULTICUT with three terminal pairs parameterized by the size of the cutset: twin-width meets flow-augmentation. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023*, pages 3229–3244. SIAM, 2023. doi:10.1137/1.9781611977554.CH123.

**18** Pinar Heggernes, Pim van 't Hof, Bart M. P. Jansen, Stefan Kratsch, and Yngve Villanger. Parameterized complexity of vertex deletion into perfect graph classes. *Theor. Comput. Sci.*, 511:172–180, 2013. doi:10.1016/J.TCS.2012.03.013.

**19** Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Comb.*, 21(1):39–60, 2001. doi:10.1007/s004930170004.

**20** Bart M. P. Jansen and Ruben F. A. Verhaegh. Search-space reduction via essential vertices revisited: Vertex multicut and cograph deletion, 2024. arXiv:2404.09769.

**21** Subhash Khot. On the power of unique 2-prover 1-round games. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, STOC 2002*, pages 767–775. ACM, 2002. doi:10.1145/509907.510017.

**22** Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within $2-\varepsilon$. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/J.JCSS.2007.06.019.

**23** Monique Laurent. A comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0-1 programming. *Math. Oper. Res.*, 28(3):470–496, 2003. doi:10.1287/MOOR.28.3.470.16391.

**24** John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

**25** Daniel Lokshtanov, Pranabendu Misra, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. FPT-approximation for FPT problems. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pages 199–218. SIAM, 2021. doi:10.1137/1.9781611976465.14.

**26** Dániel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.*, 43(2):355–388, 2014. doi:10.1137/110855247.

**27** James Nastos and Yong Gao. Bounded search tree algorithms for parametrized cograph deletion: Efficient branching rules by exploiting structures of special graph classes. *Discret. Math. Algorithms Appl.*, 4(1), 2012. doi:10.1142/S1793830912500085.

**28** Rolf Niedermeier and Peter Rossmanith. An efficient fixed-parameter algorithm for 3-hitting set. *J. Discrete Algorithms*, 1(1):89–102, 2003. doi:10.1016/S1570-8667(03)00009-1.

**29** Marcin Pilipczuk and Magnus Wahlström. Directed multicut is W[1]-hard, even for four terminal pairs. *ACM Trans. Comput. Theory*, 10(3):13:1–13:18, 2018. doi:10.1145/3201775.

**30** Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. *J. ACM*, 66(5):33:1–33:38, 2019. doi:10.1145/3325116.

**31** Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

**32** Ola Svensson. Hardness of vertex deletion and project scheduling. *Theory Comput.*, 9:759–781, 2013. doi:10.4086/toc.2013.v009a024.

**33** Vijay V. Vazirani. *Approximation algorithms*. Springer, 2001. URL: http://www.springer.com/computer/theoretical+computer+science/book/978-3-540-65367-7.

**34** Karsten Weihe. Covering trains by stations or the power of data reduction. In *Algorithms and Experiments (ALEX98)*, pages 1–8, 1998. URL: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.2173.