Faster FPT Algorithm for 5-Path Vertex Cover

Radovan Červený
Department of Theoretical Computer Science, Faculty of Information Technology,
Czech Technical University in Prague, Prague, Czech Republic
radovan.cerveny@fit.cvut.cz

Ondřej Suchý
Department of Theoretical Computer Science, Faculty of Information Technology,
Czech Technical University in Prague, Prague, Czech Republic
ondrej.suchy@fit.cvut.cz

Abstract

The problem of d-Path Vertex Cover, d-PVC lies in determining a subset F of vertices of a given graph $G = (V, E)$ such that $G \setminus F$ does not contain a path on d vertices. The paths we aim to cover need not to be induced. It is known that the d-PVC problem is NP-complete for any $d \geq 2$. When parameterized by the size of the solution k, 5-PVC has direct trivial algorithm with $\mathcal{O}(5^k n^{\mathcal{O}(1)})$ running time and, since d-PVC is a special case of d-Hitting Set, an algorithm running in $\mathcal{O}(4.0755^k n^{\mathcal{O}(1)})$ time is known. In this paper we present an iterative compression algorithm that solves the 5-PVC problem in $\mathcal{O}(4^k n^{\mathcal{O}(1)})$ time.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Fixed parameter tractability

Keywords and phrases graph algorithms, Hitting Set, iterative compression, parameterized complexity, d-Path Vertex Cover

Digital Object Identifier 10.4230/LIPIcs.MFCS.2019.32

Related Version `https://arxiv.org/abs/1906.09213`

Funding *Radovan Červený*: Supported by grant 17-20065S of the Czech Science Foundation.
Ondřej Suchý: The author acknowledges the support of the OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

1 Introduction

The problem of d-Path Vertex Cover, d-PVC lies in determining a subset F of vertices of a given graph $G = (V, E)$ such that $G \setminus F$ does not contain a path on d vertices (even not a non-induced one). The problem was first introduced by Brešar et al. [1], but its NP-completeness for any $d \geq 2$ follows already from the meta-theorem of Lewis and Yannakakis [11]. The 2-PVC problem corresponds to the well known Vertex Cover problem and the 3-PVC problem is also known as Maximum Dissociation Set. The d-PVC problem is motivated by the field of designing secure wireless communication protocols [12] or in route planning and speeding up shortest path queries [9].

Since the problem is NP-hard, any algorithm solving the problem exactly is expected to have exponential running time. If one measures the running time solely in terms of the input size, then several efficient (faster than trivial enumeration) exact algorithms are known for 2-PVC and 3-PVC. In particular, 2-PVC (Vertex Cover) can be solved in $\mathcal{O}(1.1996^n)$ time and polynomial space due to Xiao and Nagamochi [20] and 3-PVC can be solved in $\mathcal{O}(1.4656^n)$ time and polynomial space due to Xiao and Kou [18].

In this paper we aim on the parameterized analysis of the problem, that is, to confine the exponential part of the running time to a specific parameter of the input, presumably much

smaller than the input size. Algorithms achieving running time $f(k)n^{\mathcal{O}(1)}$ are called *parameterized*, *fixed-parameter tractable*, or *FPT*. See Cygan et al. [3] for a broader introduction to parameterized algorithms.

When parameterized by the size of the solution k, the d-PVC problem is directly solvable by a trivial FPT algorithm that runs in $\mathcal{O}^*(d^k)$ time.[1] However, since d-PVC is a special case of d-Hitting Set, it was shown by Fomin et al. [6] that for any $d \geq 4$ we have an algorithm solving d-PVC in $\mathcal{O}^*((d-0.9245)^k)$. In order to find more efficient solutions, the problem has been extensively studied in a setting where d is a small constant. For the 2-PVC (Vertex Cover) problem, the algorithm of Chen, Kanj, and Xia [2] has the currently best known running time of $\mathcal{O}^*(1.2738^k)$. For 3-PVC, Tu [16] used iterative compression to achieve a running time $\mathcal{O}^*(2^k)$. This was later improved by Katrenič [10] to $\mathcal{O}^*(1.8127^k)$, by Xiao and Kou [19] to $\mathcal{O}^*(1.7485^k)$ by using a branch-and-reduce approach and it was further improved by Tsur [15] to $\mathcal{O}^*(1.713^k)$. For the 4-PVC problem, Tu and Jin [17] again used iterative compression and achieved a running time $\mathcal{O}^*(3^k)$ and Tsur [14] claims to have an algorithm with running time $\mathcal{O}^*(2.619^k)$.

We present an algorithm that solves the 5-PVC problem parameterized by the size of the solution k in $\mathcal{O}^*(4^k)$ time by employing the iterative compression technique. Using the result of Fomin et al. [7] this also yields $\mathcal{O}(1.7501^n)$ time algorithm improving upon previously known $\mathcal{O}(1.7547^n)$ time algorithm.

Organization of this paper. We introduce the notation and define the 5-PVC problem in section 2. Our disjoint compression routine for iterative compression is exposed in section 3. We conclude this paper with a few open questions. Due to space constraints most proofs were deferred to the full version of this paper. However, to showcase our technique of proving the correctness of our algorithm, we included the proofs of correctness for Rules (R10) and (R13.1), and proofs of Lemmata 18 and 21.

2 Preliminaries

We use the $\mathcal{O}^*$ *notation* as described by Fomin and Kratsch [8], which is a modification of the big-$\mathcal{O}$ notation suppressing all polynomially bounded factors. We use the notation of parameterized complexity as described by Cygan et al. [3]. We use standard graph notation and consider simple and undirected graphs unless otherwise stated. Vertices of graph G are denoted by $V(G)$, edges by $E(G)$. By $G[X]$ we denote the subgraph of G induced by vertices of $X \subseteq V(G)$. By $N(v)$ we denote the set of neighbors of $v \in V(G)$ in G. Analogically, $N(X) = \bigcup_{x \in X} N(x)$ denotes the set of neighbors of vertices in $X \subseteq V(G)$. The degree of vertex v is denoted by $deg(v) = |N(v)|$. For simplicity, we write $G \setminus v$ for $v \in V(G)$ and $G \setminus X$ for $X \subseteq V(G)$ as shorthands for $G[V(G) \setminus \{v\}]$ and $G[V(G) \setminus X]$, respectively.

A *k-path*, denoted as an ordered k-tuple $P_k = (p_1, p_2, \ldots, p_k)$, is a path on k vertices $\{p_1, p_2, \ldots, p_k\}$. A path P_k *starts* at vertex x when $p_1 = x$. A *k-cycle* is a cycle on k vertices. A *triangle* is a 3-cycle. A P_5*-free* graph is a graph that does not contain a P_5 as a subgraph (the P_5 need not to be induced). The 5-Path Vertex Cover problem is formally defined as follows:

5-Path Vertex Cover, 5-PVC	
Input:	A graph $G = (V, E)$, an integer $k \in Z_0^+$.
Output:	A set $F \subseteq V$, such that $\lvert F \rvert \leq k$ and $G \setminus F$ is a P_5-free graph.

[1] The $\mathcal{O}^*()$ notation suppresses all factors polynomial in the input size.

▶ **Definition 1.** *A star is a graph S with vertices $V(S) = \{s\} \cup \{l_1, \ldots, l_k\}$, $k \geq 3$ and edges $E(S) = \{\{s, l_i\} \mid i \in \{1, \ldots, k\}\}$. Vertex s is called a center, vertices $L = \{l_1, \ldots, l_k\}$ are called leaves.*

▶ **Definition 2.** *A star with a triangle is a graph $S^{\triangle}$ with vertices $V(S^{\triangle}) = \{s, t_1, t_2\} \cup \{l_1, \ldots, l_k\}$, $k \geq 1$ and edges $E(S^{\triangle}) = \{\{s, t_1\}, \{s, t_2\}, \{t_1, t_2\}\} \cup \{\{s, l_i\} \mid i \in \{1, \ldots, k\}\}$. Vertex s is called a center, vertices $T = \{t_1, t_2\}$ are called triangle vertices and vertices $L = \{l_1, \ldots, l_k\}$ are called leaves.*

▶ **Definition 3.** *A di-star is a graph D with vertices $V(D) = \{s, s'\} \cup \{l_1, \ldots, l_k\} \cup \{l'_1, \ldots, l'_m\}$, $k \geq 1, m \geq 1$ and edges $E(D) = \{\{s, s'\}\} \cup \{\{s, l_i\} \mid i \in \{1, \ldots, k\}\} \cup \{\{s', l'_j\} \mid j \in \{1, \ldots, m\}\}$. Vertices s, s' are called centers, vertices $L = \{l_1, \ldots, l_k\}$ and $L' = \{l'_1, \ldots, l'_m\}$ are called leaves.*

▶ **Lemma 4.** *If a connected graph is P_5-free and has more than 5 vertices, then it is a star, a star with a triangle, or a di-star.*

3 5-PVC with P_5-free bipartition

We employ the generic iterative compression framework as described by Cygan et al. [3, pages 80–81]. We skip the generic steps and only present the disjoint compression routine (see the full version of the paper for a brief discussion of the whole iterative compression algorithm). That is, we assume that we are given a solution to the problem and search for another solution which is strictly smaller than and disjoint from the given one. Moreover, if the graph induced by the given solution contains a P_5, then we can directly answer no. Hence our routine DISJOINT_R restricts itself to a problem called 5-PVC WITH P_5-FREE BIPARTITION and we need it to run in $\mathcal{O}^*(3^k)$ time.

A *P_5-free bipartition* of graph $G = (V, E)$ is a pair (V_1, V_2) such that $V = V_1 \cup V_2$, $V_1 \cap V_2 = \emptyset$ and $G[V_1], G[V_2]$ are P_5-free. The 5-PVC WITH P_5-FREE BIPARTITION problem is formally defined as follows:

5-PVC WITH P_5-FREE BIPARTITION, 5-PVCwB	
INPUT:	A graph $G = (V, E)$ with P_5-free bipartition (V_1, V_2), an integer $k \in Z_0^+$.
OUTPUT:	A set $F \subseteq V_2$, such that $\lvert F\rvert \leq k$ and $G \setminus F$ is a P_5-free graph.

Throughout this paper the vertices from V_1 will be also referred to as "red" vertices and vertices from V_2 will be also refereed to as "blue" vertices.

3.1 Algorithm

Our algorithm is a recursive procedure DISJOINT_R(G, V_1, V_2, F, k), where G is the input graph, V_1, V_2 are the partitions of the P_5-free bipartition of G, F is the solution being constructed, and k is the maximum number of vertices we can still add to F. The procedure repeatedly tries to apply a series of rules with a condition that a rule (RI) can be applied only if all rules that come before (RI) cannot be applied. It is paramount that in every call of DISJOINT_R at least one rule can be applied. The main work is done in rules of two types: *reduction rules* and *branching rules*. To make it easier for the reader we also use rules called *context rules*, which only describe the configuration we are currently in and serve as some sort of a parent rules for their subrules.

A *reduction rule* is used to simplify a problem instance, i.e. remove some vertices or edges from G and possibly add some vertices to a solution, or to halt the algorithm. A *branching rule* splits the problem instance into at least two subinstances. The branching is based on subsets of vertices that we try to add to a solution and by adding them to the solution we also remove them from G.

The notation we use to denote the individual branches of a branching rule is as follows: $\langle X_1 \mid X_2 \mid \ldots \mid X_l \rangle$. Such a rule has l branches and $X_1, X_2, \ldots, X_l$ are subsets of V_2 which we try to add to the solution. This rule is translated into the following l calls of the procedure:

$$\text{DISJOINT\_R}(G \setminus X_i, V_1, V_2 \setminus X_i, F \cup X_i, k - |X_i|) \text{ for } i \in \{1, \ldots, l\}$$

A rule is *applicable* if the conditions of the rule are satisfied and none of the previous rules is applicable. If a context rule is not applicable, it means that none of its subrules is applicable.

A reduction rule is *correct* if it satisfies that the problem instance has a solution if and only if the simplified problem instance has a solution. A branching rule is *correct* if it satisfies that the problem instance has a solution if and only if at least one of the branches of the rule will return a solution.

When we say we *delete* a vertex, we mean that we remove it from G and also add it to the solution F. When we say we *remove* a vertex, we mean that we remove it from G and *do not* add it to the solution F.

For the rest of this paper assume that the parameters of the current call of DISJOINT_R are G, V_1, V_2, F, k.

3.2 Preprocessing

▶ **Reduction rule (R0).** This rule stops the recursion of DISJOINT_R. It has three stopping conditions:

1. If $k < 0$, return *no solution*;
2. else if G is P_5-free, return F;
3. else if $k = 0$, return *no solution*.

▶ **Reduction rule (R1).** Let $v \in V(G)$ be a vertex such that there is no P_5 in G that uses v. Then remove v from G.

▶ **Branching rule (R2).** Let P be a P_5 in G with $X = V(P) \cap V_2$ such that $|X| \leq 3$. Then branch on $\langle x_1 \mid x_2 \mid \ldots \rangle, x_i \in X$, i.e. branch on the blue vertices of P.

▶ **Lemma 5.** *Assume that Rules (R0) – (R2) are not applicable. Then for each vertex $v \in V(G)$ there exists a P_5 in G that uses v; every P_5 in G uses exactly one red vertex; and there are only isolated vertices in $G[V_1]$.*

3.3 Dealing with isolated vertices in $G[V_2]$

▶ **Lemma 6.** *Assume that Rules (R0) – (R2) are not applicable. Let v be an isolated vertex in $G[V_2]$ and let F be a solution to* 5-PVCWB *which uses vertex v. Then there exists a solution F' that does not use vertex v and $|F'| \leq |F|$.*

▶ **Branching rule (R3).** Let v be an isolated vertex in $G[V_2]$ and let $P = (v, w, x, y, z)$ be a P_5 where w is a red vertex. Then branch on $\langle x \mid y \mid z \rangle$.

▶ **Lemma 7.** *Assume that Rules (R0) – (R3) are not applicable. Then there are no isolated vertices in $G[V_2]$.*

3.4 Dealing with isolated edges in $G[V_2]$

▶ **Lemma 8.** *Assume that Rules (R0) – (R3) are not applicable. Let v be a blue vertex to which at least two red vertices are connected and let C_v be a connected component of $G[V_2]$ which contains v. Then for each red vertex w connected to v we have that $N(w) \subseteq V(C_v)$.*

▶ **Lemma 9.** *Assume that Rules (R0) – (R3) are not applicable. Let $e = \{u, v\}$ be a blue edge to which at least two red vertices are connected in a way that to both u and v there is at least one red vertex connected. Let C_e be a connected component of $G[V_2]$ which contains e. Then for each red vertex w connected to e we have that $N(w) \subseteq V(C_e)$.*

▶ **Lemma 10.** *Let X be a subset of V_2 such that $N(X) \cap V_1 = \emptyset$ and $|N(X) \cap V_2| = 1$. If there exists a solution F such that $F \cap X \neq \emptyset$, then there exists a solution F' such that $F' \cap X = \emptyset$ and $|F'| \leq |F|$.*

▶ **Definition 11.** *We say that two nodes x, y are twins if $N(x) \setminus \{y\} = N(y) \setminus \{x\}$.*

▶ **Lemma 12.** *Let x, y be blue vertices that are twins. Let F be a solution and $x \in F$. Then at least one of the following holds:*
(1) $y \in F$,
(2) $F' = (F \setminus \{x\}) \cup \{y\}$ is a solution.

▶ **Branching rule (R4).** Let $e = \{u, v\}$ be an isolated edge in $G[V_2]$. We know from Lemmata 8 and 9 that there is only one red vertex w connected to e, because if there were at least two red vertices connected to e, then there would be no P_5 that uses vertices from e. Let there be a red vertex w connected to at least one vertex in e. If w is connected only to one vertex in e, let that vertex be v. Assume that x is some vertex to which w connects outside e and let y be a neighbor of x in $G[V_2]$. Then branch on $\langle v \mid x \mid y \rangle$.

▶ **Lemma 13.** *Assume that Rules (R0) – (R4) are not applicable. Then there are no isolated edges in $G[V_2]$.*

3.5 Dealing with isolated P_3 paths in $G[V_2]$

▶ **Context rule (R5).** Let P be a $P_3 = (t, u, v)$ that forms a connected component in $G[V_2]$. From Lemmata 5, 8 and 9 we know that there is only one red vertex w connected to P. We further know that w must be connected to some component of $G[V_2]$ other than P, otherwise no P_5 could be formed. Assume that x is some vertex to which w connects outside P and let y be a neighbor of x in $G[V_2]$. This rule is split into four subrules (R5.1), (R5.2), (R5.3) and (R5.4) based on how w is connected to P.

▶ **Branching rule (R5.1).** Vertex w is connected only to v in P. Then branch on $\langle v \mid x \rangle$.

▶ **Branching rule (R5.2).** Vertex w is connected only to u, v in P. Then branch on $\langle u \mid v \mid x \rangle$.

▶ **Branching rule (R5.3).** Vertex w is connected only to u in P. Then branch on $\langle u \mid x \mid y \rangle$.

▶ **Branching rule (R5.4).** Vertex w is connected to t, v in P and w can be also connected to u in P. Then branch on $\langle u \mid v \mid x \rangle$.

▶ **Lemma 14.** *Assume that Rules (R0) – (R5) are not applicable. Then there are no isolated P_3 paths in $G[V_2]$.*

3.6 Dealing with isolated triangles in $G[V_2]$

▶ **Context rule (R6).** Let T be a $K_3 = \{t, u, v\}$ that forms a connected component in $G[V_2]$. From Lemmata 5 and 8 we know that there is only one red vertex w connected to T. We further know that w must be connected to some component of $G[V_2]$ other than T, otherwise no P_5 could be formed. Assume that x is some vertex to which w connects outside T and let y be a neighbor of x in $G[V_2]$. This rule is split into three subrules (R6.1), (R6.2) and (R6.3) based on how w is connected to T.

▶ **Branching rule (R6.1).** Vertex w is connected only to one vertex in T, let that vertex be v. Then branch on $\langle\, v \mid x \,\rangle$.

▶ **Branching rule (R6.2).** Vertex w is connected to exactly two vertices in T, let those vertices be u, v. Then branch on $\langle\, t \mid v \mid x \,\rangle$.

▶ **Branching rule (R6.3).** Vertex w is connected to all vertices in T. Then branch on $\langle\, v \mid x \,\rangle$.

▶ **Lemma 15.** *Assume that Rules (R0) – (R6) are not applicable. Then there are no isolated triangles in $G[V_2]$.*

3.7 Dealing with 4-cycles in $G[V_2]$

▶ **Lemma 16.** *Let C be a connected component of $G[V_2]$ and $X = V(C) \cap N(V_1)$. Let F be a solution that deletes at least $|X|$ vertices in C. Then $F' = (F \setminus V(C)) \cup X$ is also a solution and $|F'| \leq |F|$.*

▶ **Context rule (R7).** Let Q be a connected component in $G[V_2]$ such that Q is a subgraph of K_4 and a 4-cycle is a subgraph of Q, label the vertices of the 4-cycle (v_1, v_2, v_3, v_4). We will call pairs of vertices $\{v_1, v_3\}$ and $\{v_2, v_4\}$ *diagonal*, all other pairs will be called *non-diagonal*. Edges corresponding to diagonal (non-diagonal) pairs are called diagonal (non-diagonal) edges, respectively. This rule is split into two subrules (R7.1), (R7.2) based on the number of red vertices connected to Q.

▶ **Reduction rule (R7.1).** Assume that there are at least two red vertices connected to Q. Then delete any vertex v_i in Q and add it to the solution F.

▶ **Context rule (R7.2).** Assume that there is only one red vertex w connected to Q and $X = V(Q) \cap N(w)$. This rule is split into five subrules (R7.2a), (R7.2b), (R7.2c), (R7.2d) and (R7.2e) based on how w is connected to Q and whether w is connected to other components.

▶ **Reduction rule (R7.2a).** Vertex w is connected only to one vertex in Q, let it be v_1. Then delete v_1 and add it to the solution F.

▶ **Branching rule (R7.2b).** Set X contains at least one diagonal pair, let that pair be $\{v_1, v_3\}$. Then branch on $\langle\, v_1 \mid v_2 \mid v_4 \,\rangle$.

▶ **Branching rule (R7.2c).** Set X contains exactly one non-diagonal pair, let that pair be $\{v_1, v_2\}$, and case *(a)* either both diagonal edges are in Q , or case *(b)* none of them is. Then branch on $\langle\, v_1 \mid v_3 \mid v_4 \,\rangle$

▶ **Reduction rule (R7.2d).** Set X contains exactly one non-diagonal pair, let that pair be $\{v_1, v_2\}$ and exactly one diagonal edge is in Q, let that edge be $\{v_1, v_3\}$. Furthermore, w is connected only to Q, i.e. $N(w) \subseteq V(Q)$. Then delete any vertex v_i in Q and add it to the solution F.

▶ **Branching rule (R7.2e).** Set X contains exactly one non-diagonal pair, let that pair be $\{v_1, v_2\}$ and exactly one diagonal edge is in Q, let that edge be $\{v_1, v_3\}$. Furthermore, w is connected to at least one more component of $G[V_2]$ other than Q, label the vertex to which w connects outside Q as x and let y be a neighbor of x in $G[V_2]$. Then branch on $\langle \{v_1, v_2\} \mid x \mid y \rangle$.

▶ **Lemma 17.** *Assume that Rules (R0) – (R7) are not applicable. Then there is no component of $G[V_2]$ that contains a 4-cycle as a subgraph.*

3.8 Dealing with stars in $G[V_2]$

In this subsection we consider a connected component of $G[V_2]$ which is isomorphic to a star. Through this subsection we denote it by S and label its vertices as in Definition 1. We remark that, as none of the Rules (R0) – (R7) is applicable, there is exactly one red vertex connected to S.

▶ **Context rule (R8).** Let S be a star in $G[V_2]$ and let w be a red vertex connected to S. This rule is divided into three subrules (R8.1), (R8.2) and (R8.3) based on how w is connected to S.

▶ **Branching rule (R8.1).** A red vertex w is connected to at least two leaves of S, let those two leaves be l_1, l_2. Then branch on $\langle l_1 \mid s \mid L \setminus \{l_1, l_2\} \rangle$.

▶ **Branching rule (R8.2).** A red vertex w is connected only to s in S and w is connected to some other vertex x in $G[V_2]$ outside S and y is a neighbor of x in $G[V_2]$. Then branch on $\langle s \mid x \mid y \rangle$.

▶ **Branching rule (R8.3).** A red vertex w is connected to l_1 in S, w can be connected also to s in S, and w is connected to some other vertex x in $G[V_2]$ outside S. Then branch on $\langle s \mid l_1 \mid x \rangle$.

▶ **Lemma 18.** *Assume that Rules (R0) – (R8) are not applicable. Then there are no stars in $G[V_2]$.*

Proof. For contradiction assume that Rules (R0) – (R8) are not applicable and there is a star S in $G[V_2]$.

If there is no P_5 that uses vertices from S, then Rule (R1) is applicable on S. Suppose there are at least two red vertices connected to S. If the red vertices are not connected to a single vertex or a single edge in S, then Rule (R2) is applicable, since there is a P_5 that uses at least two red vertices. So suppose the red vertices are connected to a single vertex or a single edge in S. Then from Lemmata 8 and 9 we know that those red vertices are not connected to any other vertices outside S. Consequently, there cannot be a P_5 that uses vertices from S and again Rule (R1) is applicable on S.

So suppose that there is a P_5 that uses vertices from S and there is only one red vertex w connected to S. If w is connected to two leaves, then Rule (R8.1) is applicable. So suppose that w is not connected to two leaves. There are three not mutually isomorphic possibilities how w can be connected to S: $\{l_1\}$, $\{s\}$, and $\{l_1, s\}$. In those cases we apply Rules (R8.3), (R8.2), and (R8.3), respectively. ◀

3.9 Dealing with stars with a triangle in $G[V_2]$

In this subsection we consider a connected component of $G[V_2]$ which is isomorphic to a star with a triangle. Through this subsection we denote it by $S^{\triangle}$ and label its vertices as in Definition 2.

▶ **Context rule (R9).** Let $S^{\triangle}$ be a star with a triangle in $G[V_2]$ and let w be a red vertex connected to $S^{\triangle}$. This rule is divided into four subrules (R9.1), (R9.2), (R9.3) and (R9.4) based on how w is connected to $S^{\triangle}$.

▶ **Branching rule (R9.1).** There is a red vertex w such that $\{t_1, t_2\} \subseteq N(w)$. Then branch on $\langle\, t_1 \mid s \mid L \,\rangle$.

▶ **Branching rule (R9.2).** There is a red vertex w such that $|\{t_1, t_2\} \cap N(w)| = 1$, assume that w is connected to t_1. Then branch on $\langle\, t_1 \mid s \mid L \,\rangle$.

▶ **Lemma 19.** *Assume that Rules (R0) – (R9.1) are not applicable and the assumptions of Rule (R9.2) are satisfied. If F is a solution that contains t_2, then at least one of the following holds:*
(1) $t_1 \in F$,
(2) $F' = (F \setminus \{t_2\}) \cup \{t_1\}$ *is a solution.*

▶ **Branching rule (R9.3).** There is a red vertex w connected to a leaf of $S^{\triangle}$, let that leaf be l_1. Then branch on $\langle\, l_1 \mid s \,\rangle$.

▶ **Branching rule (R9.4).** A red vertex w is connected only to s in $S^{\triangle}$. Then w must be also connected to some component of $G[V_2]$ other than $S^{\triangle}$, otherwise no P_5 would occur in the component containing $S^{\triangle}$. Label the vertex to which w connects outside $S^{\triangle}$ as x. Then branch on $\langle\, s \mid x \,\rangle$.

▶ **Lemma 20.** *Assume that Rules (R0) – (R9) are not applicable. Then there are no stars with a triangle in $G[V_2]$.*

3.10 Dealing with di-stars in $G[V_2]$

In this subsection we consider a connected component of $G[V_2]$ which is isomorphic to a di-star. Through this subsection we denote it by D and label its vertices as in Definition 3.

▶ **Branching rule (R10).** Let D be a di-star in $G[V_2]$ and let there be a red vertex w connected to at least two leaves on the same side of the di-star, i.e. $|N(w) \cap L| \geq 2$ or $|N(w) \cap L'| \geq 2$. Assume that those leaves are from L and l_1, l_2 are among them. Observe that there is no other red vertex connected to l_1, l_2. Then branch on $\langle\, l_1 \mid s \mid s' \,\rangle$.

Proof of correctness. We have to delete something in $\{l_1, l_2, s, s'\}$ and since l_1, l_2 are twins, from Lemma 12 we know that we have to try only one of them, thus branching on $\langle\, l_1 \mid s \mid s' \,\rangle$ is correct. ◀

▶ **Context rule (R11).** Let D be a di-star in $G[V_2]$ and let w be the only red vertex connected to D such that $\{s, s'\} \subseteq N(w)$. This rule is split into three subrules (R11.1), (R11.2), and (R11.3) based on the degrees of s and s'.

▶ **Context rule (R11.1).** Assume that both s, s' have degree two in $G[V_2]$, i.e. the di-star D is actually a P_4. This rule is split into four subrules (R11.1a), (R11.1b), (R11.1c), and (R11.1d) based on how w is connected to D and whether w is connected to other components.

▶ **Branching rule (R11.1a).** Vertex w is connected only to s, s' in D. Then branch on $\langle\, s \mid s' \,\rangle$.

▶ **Branching rule (R11.1b).** Vertex w is connected to s, s' and to one leaf in D, let that leaf be l_1. Then branch on $\langle\, l_1 \mid s \mid s' \,\rangle$.

▶ **Branching rule (R11.1c).** Vertex w is connected to l_1, l'_1, s, s' in D and to at least one other component of $G[V_2]$, label the vertex w connects to outside D as x and the neighbor of x in $G[V_2]$ as y. Then branch on $\langle\, x \mid y \mid \{l_1, s'\} \mid \{s, l'_1\} \mid \{s, s'\} \,\rangle$.

▶ **Reduction rule (R11.1d).** Vertex w is connected only to l_1, l'_1, s, s' in D and to no other component of $G[V_2]$. Then delete any vertex v in D and add it to the solution F.

▶ **Context rule (R11.2).** Assume that exactly one of s, s' has degree at least 3 in $G[V_2]$, let it be s. This rule is split into four subrules (R11.2a), (R11.2b), (R11.2c), and (R11.2d) based on how w is connected to D.

▶ **Branching rule (R11.2a).** Vertex w is connected only to s, s' in D. Then branch on $\langle\, s \mid s' \,\rangle$.

▶ **Branching rule (R11.2b).** Vertex w is connected to s, s' and exactly one leaf from L in D, let that leaf be l_1. Then branch on $\langle\, l_1 \mid s \mid s' \,\rangle$.

▶ **Branching rule (R11.2c).** Vertex w is connected to s, s', l'_1 in D. Then branch on $\langle\, l'_1 \mid s \mid s' \,\rangle$.

▶ **Branching rule (R11.2d).** Vertex w is connected to s, s', l'_1 and exactly one leaf from L in D, let that leaf be l_1. Then branch on $\langle\, l_1 \mid s \mid s' \,\rangle$.

▶ **Branching rule (R11.3).** Assume that both s, s' in D have degree at least 3 in $G[V_2]$. Then branch on $\langle\, L \mid s \mid s' \mid L' \,\rangle$.

▶ **Context rule (R12).** Let D be a di-star in $G[V_2]$ and let w be the only red vertex connected to D such that w is connected to D by exactly two edges. We know that w is connected to a subset of $\{l_1, s, s', l'_1\}$, but not to both s and s'. This rule is split into three subrules (R12.1), (R12.2), and (R12.3) based on how w is connected to D.

▶ **Branching rule (R12.1).** Vertex w is connected to a center and its leaf in D, let them be s and l_1. Then branch on $\langle\, l_1 \mid s \,\rangle$.

▶ **Branching rule (R12.2).** Vertex w is connected to a center and to a leaf of the other center in D, let them be s' and l_1. Then branch on $\langle\, l_1 \mid s \mid s' \,\rangle$.

▶ **Context rule (R12.3).** Vertex w is connected to two opposite leaves in D, let them be l_1 and l'_1. This rule is split into four subrules (R12.3a), (R12.3b), (R12.3c), and (R12.3d) based on the degrees of s and s' and whether w is connected to other components.

▶ **Branching rule (R12.3a).** Both s, s' in D have degree 2 in $G[V_2]$ and w is connected to a component of $G[V_2]$ other than D, let x be the vertex w connects to outside D and let y be a neighbor of x in $G[V_2]$. Then branch on $\langle\, x \mid y \mid \{l_1, l'_1\} \,\rangle$.

▶ **Reduction rule (R12.3b).** Both s, s' in D have degree 2 in $G[V_2]$ and w is not connected to a component of $G[V_2]$ other than D. Then delete any vertex v in D and add it to the solution F.

▶ **Branching rule (R12.3c).** Exactly one of s, s' in D has degree at least 3 in $G[V_2]$, let it be s. Then branch on $\langle\, l_1 \mid s \mid l'_1 \,\rangle$.

▶ **Branching rule (R12.3d).** Both s, s' in D have degree at least 3 in $G[V_2]$. Then branch on $\langle\, s \mid s' \mid \{l_1, l'_1\} \,\rangle$.

▶ **Context rule (R13).** Let D be a di-star in $G[V_2]$ and let w be the only red vertex connected to D such that w is connected to D by exactly three edges. We know that w is connected to a subset of $\{l_1, s, s', l'_1\}$, but not to both s and s'. Assume that w is connected to l_1, s, l'_1. This rule is split into four subrules (R13.1), (R13.2), (R13.3) and (R13.4) based on the degrees of s and s' and whether w is connected to other components.

▶ **Branching rule (R13.1).** Both s, s' in D have degree 2 in $G[V_2]$ and w is connected to at least one other component of $G[V_2]$, label the vertex w connects to outside D as x and the neighbor of x in $G[V_2]$ as y. Then branch on $\langle\, x \mid y \mid \{l_1, s'\} \mid \{s, l'_1\} \,\rangle$.

Proof of correctness. If none of the vertices x, y is deleted, then we have to delete at least two and, by Lemma 16, at most three vertices in D. Suppose we wanted to delete exactly two vertices. Out of six possible pairs, only $\{l_1, s'\}, \{s, s'\}, \{s, l'_1\}$ lead to a solution. We do not have to try $\{s, s'\}$, since if we delete s, then Lemma 10 becomes applicable and we may delete l'_1 instead of s'. Finally, if we wanted to delete three vertices, then by Lemma 16 those vertices would be $\{l_1, s, l'_1\}$, but this is already covered by branching on $\{s, l'_1\}$. Thus branching on $\langle\, x \mid y \mid \{l_1, s'\} \mid \{s, l'_1\} \,\rangle$ is correct. ◀

▶ **Reduction rule (R13.2).** Both s, s' in D have degree 2 in $G[V_2]$ and w is not connected to other component of $G[V_2]$. Then delete any vertex v in D and add it to the solution F.

▶ **Branching rule (R13.3).** Vertex s in D has degree at least 3 in $G[V_2]$. Then branch on $\langle\, l_1 \mid s \mid l'_1 \,\rangle$.

▶ **Branching rule (R13.4).** Vertex s' in D has degree at least 3 in $G[V_2]$. Then branch on $\langle\, l_1 \mid s' \mid l'_1 \,\rangle$.

▶ **Context rule (R14).** There is exactly one red vertex w connected to D by one edge. This rule is split into two subrules (R14.1) and (R14.2) based on how w is connected to D.

▶ **Reduction rule (R14.1).** Vertex w is connected to a leaf in D, let it be l_1. Then delete l_1 and add it to the solution F.

▶ **Branching rule (R14.2).** Vertex w is connected to a center in D, let it be s, and w is connected to at least one component of $G[V_2]$ other than D, label the vertex w connects to outside D as x. Then branch on $\langle\, s \mid x \,\rangle$.

▶ **Reduction rule (R15).** There are at least two red vertices connected to D by exactly one edge and they are connected to a single vertex. From Lemma 8 we know, that the red vertices are not connected to a component of $G[V_2]$ other than D and hence the single vertex must be a leaf, let it be l_1, otherwise no P_5 would be formed and Rule (R1) would be applicable. Then delete l_1 and add it to the solution F.

▶ **Branching rule (R16).** There are at least two red vertices connected to D by exactly one edge and they are connected to two opposite leaves, let those leaves be l_1, l'_1. Assume that there is at least one red vertex connected to each one of them. Further assume that the red vertices connected to l_1 are not connected to a component of $G[V_2]$ other than D. Then branch on $\langle\, s' \mid l'_1 \,\rangle$.

▶ **Branching rule (R17).** Let there be a di-star D and the two red vertices w, w' connected to D are connected to leaves l_1, l'_1, respectively, and at least one of the centers has degree at least three, let it be s. Then branch on $\langle\, s \mid s' \mid l'_1 \,\rangle$.

▶ **Branching rule (R18).** Let there be a di-star D and the two red vertices w, w' connected to D are connected to leaves l_1, l'_1, respectively, and both centers have degree exactly two. Then branch on $\langle l_1 \mid l'_1 \rangle$.

▶ **Lemma 21.** *Assume that Rules (R0) – (R9) are not applicable. Then at least one of Rules (R10) – (R18) is applicable.*

Proof. From Lemma 4 together with Lemmata 7, 13, 14, 15, 17, 18 and 20 we are now in the situation in which all components of $G[V_2]$ are di-stars and there must be a di-star D in $G[V_2]$ such that there is a P_5 that uses the vertices of D which implies there is at least one red vertex connected to D. For contradiction assume that Rules (R10) – (R18) are not applicable, i.e. no rules are applicable.

Let w be some red vertex connected to D. If $|N(w) \cap L| \geq 2$ or $|N(w) \cap L'| \geq 2$, then Rule (R10) is applicable. If there there is a red vertex w connected to l_i and a red vertex w' connected to l_j, $j \neq i$, then Rule (R2) applies. Similarly with vertices connected to l'_i and l'_j. So for the rest of this proof assume that each red vertex can be connected only to vertices l_1, s, s', or l'_1.

Firstly, assume that there is only one red vertex w connected to D. In Table 1 we list all possibilities (omitting several isomorphic cases) based on how w is connected to D, on the degrees of s and s', and whether w is connected only to D ($N(w) \subseteq V(D)$) or w is also connected outside D ($N(w) \not\subseteq V(D)$).

Observe that if there were at least two red vertices connected to D and w was connected to D by at least two edges, then Rule (R2) would be applicable with the only exception in case where w is connected to $\{l_1, s\}$ or $\{s', l'_1\}$ and the other red vertices to s or s', respectively. But this exception is resolved by Rule (R1) since vertices connected only to s or s' in this configuration are not used by any P_5. With this in mind, if there are at least two red vertices connected to D, then they are connected to D by only one edge.

Secondly, assume that there are at least two red vertices connected to D by exactly one edge. Let $X \subseteq V(D)$ be the vertices to which the red vertices are connected in D. Again, if $|X \cap L| \geq 2$ or $|X \cap L'| \geq 2$, then Rule (R2) applies. So suppose that $X \subseteq \{l_1, s, s', l'_1\}$. If $\{l_1, s'\} \subseteq X$ or $\{s, l'_1\} \subseteq X$ (which covers also cases where $|X| \geq 3$), then again Rule (R2) is applicable. If the vertices are connected to a single edge, then at least one of the vertices of such edge is a center, the vertices connected to it are not used by any P_5, and Rule (R1) applies. We conclude that the red vertices may be connected only to a single vertex or to two opposite leaves in D.

Thirdly, assume that the red vertices are connected to a single vertex. If that vertex is a leaf, then Rule (R15) is applicable, otherwise Rule (R1) is applicable.

Fourthly, assume that the red vertices are connected to two opposite leaves, let them be l_1 and l'_1, and let W be the set of red vertices connected to l_1 and W' be the set of red vertices connected to l'_1. If the vertices in W or in W' (or both) are not connected to any component other than D, then Rule (R16) is applicable. This is the case whenever $|W| \geq 2$ or $|W'| \geq 2$ by Lemma 8.

Observe that now we are in situation in which there are exactly two red vertices w and w' connected to D by exactly one edge and these vertices are connected to l_1 and l'_1, assume that w is connected to l_1 and w' is connected to l'_1. Furthermore, vertices w and w' are connected to at least one other di-star in $G[V_2]$. If at least one of L, L' has size at least two, then Rule (R17) is applicable, otherwise all di-stars in $G[V_2]$ are actually a P_4 paths and Rule (R18) is applicable.

Finally, there is no di-star remaining in $G[V_2]$ which together with Lemmata 4, 7, 13, 14, 15, 17, 18 and 20 implies that $G[V_2] = \emptyset$ and since V_1, V_2 is a P_5-free bipartition, there is no P_5 path remaining in G and Rule (R0) is applicable. ◀

Table 1 Possible configurations of single red vertex w and D in Lemma 21.

$N(w) \cap V(D)$	$N(w) \not\subseteq V(D)$			$N(w) \subseteq V(D)$		
	$\lvert L\rvert = 1$, $\lvert L'\rvert = 1$	$\lvert L\rvert > 1$, $\lvert L'\rvert = 1$	$\lvert L\rvert > 1$, $\lvert L'\rvert > 1$	$\lvert L\rvert = 1$, $\lvert L'\rvert = 1$	$\lvert L\rvert > 1$, $\lvert L'\rvert = 1$	$\lvert L\rvert > 1$, $\lvert L'\rvert > 1$
$\{l_1\}$	(R14.1)	(R14.1)	(R14.1)	(R14.1)	(R14.1)	(R14.1)
$\{s\}$	(R14.2)	(R14.2)	(R14.2)	(R1)	(R1)	(R1)
$\{s'\}$	(R14.2)	(R14.2)	(R14.2)	(R1)	(R1)	(R1)
$\{l'_1\}$	(R14.1)	(R14.1)	(R14.1)	(R14.1)	(R14.1)	(R14.1)
$\{l_1, s\}$	(R12.1)	(R12.1)	(R12.1)	(R12.1)	(R12.1)	(R12.1)
$\{l_1, s'\}$	(R12.2)	(R12.2)	(R12.2)	(R12.2)	(R12.2)	(R12.2)
$\{l_1, l'_1\}$	(R12.3a)	(R12.3c)	(R12.3d)	(R12.3b)	(R12.3c)	(R12.3d)
$\{s, s'\}$	(R11.1a)	(R11.2a)	(R11.3)	(R11.1a)	(R11.2a)	(R11.3)
$\{s, l'_1\}$	(R12.2)	(R12.2)	(R12.2)	(R12.2)	(R12.2)	(R12.2)
$\{s', l'_1\}$	(R12.1)	(R12.1)	(R12.1)	(R12.1)	(R12.1)	(R12.1)
$\{l_1, s, s'\}$	(R11.1b)	(R11.2b)	(R11.3)	(R11.1b)	(R11.2b)	(R11.3)
$\{l_1, s, l'_1\}$	(R13.1)	(R13.3)	(R13.3)	(R13.2)	(R13.3)	(R13.3)
$\{l_1, s', l'_1\}$	(R13.1)	(R13.4)	(R13.3)	(R13.2)	(R13.4)	(R13.3)
$\{s, s', l'_1\}$	(R11.1b)	(R11.2c)	(R11.3)	(R11.1b)	(R11.2c)	(R11.3)
$\{l_1, s, s', l'_1\}$	(R11.1c)	(R11.2d)	(R11.3)	(R11.1d)	(R11.2d)	(R11.3)

3.11 Final remarks

From Lemma 21 we know that there is always at least one rule applicable. It remains to analyze the running time of the disjoint compression routine DISJOINT_R.

▶ **Theorem 22.** *The* DISJOINT_R *procedure solves the* 5-PVCwB *problem in* $\mathcal{O}^*(3^k)$ *time.*

By standard arguments (see Cygan et al. [3, pages 80–81]) we get the following corollary.

▶ **Corollary 23.** *The iterative compression algorithm solves the* 5-PVC *problem and runs in* $\mathcal{O}^*(4^k)$ *time.*

4 Conclusion

We conclude this paper with a few open questions.

Firstly, we see the trend of solving 3-PVC, 4-PVC and now 5-PVC with the iterative compression technique, so it is natural to ask whether this approach can be further used for 6-PVC or even to d-PVC in general. However, given the complexity (number of rules) of the algorithm presented in this paper, it seems more reasonable to first try to find a simpler algorithm for 5-PVC.

Secondly, motivated by the work of Orenstein et al. [13], we ask whether known algorithms for 3-PVC, 4-PVC, 5-PVC can be generalized to work with directed graphs.

Finally, due to Fafianie and Kratsch [5] we know that d-PVC problem has a kernel with $\mathcal{O}(k^d)$ vertices and edges. Dell and van Melkebeek [4] showed that there is no $\mathcal{O}(k^{d-\epsilon})$ kernel for any $\epsilon > 0$ for general d-HITTING SET unless coNP is in NP/poly, which would imply a collapse of the polynomial-time hierarchy. However, for 3-PVC problem, Xiao and Kou [19] presented a kernel with $5k$ vertices. To our knowledge, it is not known whether there exists a linear kernel for 4-PVC or any d-PVC with $d \geq 5$.

References

1 Boštjan Brešar, František Kardoš, Ján Katrenič, and Gabriel Semanišin. Minimum k-path vertex cover. *Discrete Applied Mathematics*, 159(12):1189–1195, 2011. doi:10.1016/j.dam.2011.04.008.

2 Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theor. Comput. Sci.*, 411(40-42):3736–3756, 2010. doi:10.1016/j.tcs.2010.06.026.

3 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.

4 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. doi:10.1145/2629620.

5 Stefan Fafianie and Stefan Kratsch. A Shortcut to (Sun)Flowers: Kernels in Logarithmic Space or Linear Time. In *Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II*, pages 299–310, 2015. doi:10.1007/978-3-662-48054-0_25.

6 Fedor V. Fomin, Serge Gaspers, Dieter Kratsch, Mathieu Liedloff, and Saket Saurabh. Iterative compression and exact algorithms. *Theor. Comput. Sci.*, 411(7-9):1045–1053, 2010. doi:10.1016/j.tcs.2009.11.012.

7 Fedor V. Fomin, Serge Gaspers, Daniel Lokshtanov, and Saket Saurabh. Exact Algorithms via Monotone Local Search. *J. ACM*, 66(2):8:1–8:23, 2019. doi:10.1145/3284176.

8 Fedor V. Fomin and Dieter Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2010. doi:10.1007/978-3-642-16533-7.

9 Stefan Funke, André Nusser, and Sabine Storandt. On k-Path Covers and their applications. *VLDB J.*, 25(1):103–123, 2016. doi:10.1007/s00778-015-0392-3.

10 Ján Katrenič. A faster FPT algorithm for 3-path vertex cover. *Inf. Process. Lett.*, 116(4):273–278, 2016. doi:10.1016/j.ipl.2015.12.002.

11 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.

12 Marián Novotný. Design and Analysis of a Generalized Canvas Protocol. In *Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices, 4th IFIP WG 11.2 International Workshop, WISTP 2010, Passau, Germany, April 12-14, 2010. Proceedings*, pages 106–121, 2010. doi:10.1007/978-3-642-12368-9_8.

13 Yaron Orenstein, David Pellow, Guillaume Marçais, Ron Shamir, and Carl Kingsford. Designing small universal k-mer hitting sets for improved analysis of high-throughput sequencing. *PLoS Computational Biology*, 13(10), 2017. doi:10.1371/journal.pcbi.1005777.

14 Dekel Tsur. An $O^*(2.619^k)$ algorithm for 4-path vertex cover. *CoRR*, abs/1811.03592, 2018. arXiv:1811.03592.

15 Dekel Tsur. Parameterized algorithm for 3-path vertex cover. *Theoretical Computer Science*, 783:1–8, 2019. doi:10.1016/j.tcs.2019.03.013.

16 Jianhua Tu. A fixed-parameter algorithm for the vertex cover P_3 problem. *Inf. Process. Lett.*, 115(2):96–99, 2015. doi:10.1016/j.ipl.2014.06.018.

17 Jianhua Tu and Zemin Jin. An FPT algorithm for the vertex cover P_4 problem. *Discrete Applied Mathematics*, 200:186–190, 2016. doi:10.1016/j.dam.2015.06.032.

18 Mingyu Xiao and Shaowei Kou. Exact algorithms for the maximum dissociation set and minimum 3-path vertex cover problems. *Theor. Comput. Sci.*, 657:86–97, 2017. doi:10.1016/j.tcs.2016.04.043.

19 Mingyu Xiao and Shaowei Kou. Kernelization and Parameterized Algorithms for 3-Path Vertex Cover. In *Theory and Applications of Models of Computation - 14th Annual Conference, TAMC 2017, Bern, Switzerland, April 20-22, 2017, Proceedings*, pages 654–668, 2017 . doi:10.1007/978-3-319-55911-7_47.

20 Mingyu Xiao and Hiroshi Nagamochi. Exact algorithms for maximum independent set. *Inf. Comput.*, 255:126–146, 2017. doi:10.1016/j.ic.2017.06.001.