A Polynomial Kernel for Paw-Free Editing

Eduard Eiben

Department of Informatics, University of Bergen, Bergen, Norway
eduard.eiben@uib.no

William Lochet

Department of Informatics, University of Bergen, Bergen, Norway
william.lochet@uib.no

Saket Saurabh

Institute of Mathematical Sciences, Chennai, India
Department of Informatics, University of Bergen, Bergen, Norway
saket@imsc.res.in

Abstract

For a fixed graph H, the H-free-Edge Editing problem asks whether we can modify a given graph G by adding or deleting at most k edges such that the resulting graph does not contain H as an induced subgraph. The problem is known to be NP-complete for all fixed H with at least 3 vertices and it admits a $2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$ algorithm. Cai and Cai [Algorithmica (2015) 71:731–757] showed that H-free-Edge Editing does not admit a polynomial kernel whenever H or its complement is a path or a cycle with at least 4 edges or a 3-connected graph with at least 1 edge missing. Their results suggest that if H is not independent set or a clique, then H-free-Edge Editing admits polynomial kernels only for few small graphs H, unless $\mathsf{coNP} \in \mathsf{NP/poly}$. Therefore, resolving the kernelization of H-free-Edge Editing for small graphs H plays a crucial role in obtaining a complete dichotomy for this problem. In this paper, we positively answer the question of compressibility for one of the last two unresolved graphs H on 4 vertices. Namely, we give the first polynomial kernel for Paw-free-Edge Editing with $\mathcal{O}\left(k^6\right)$ vertices.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Kernelization, Paw-free graph, H-free editing, graph modification problem

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

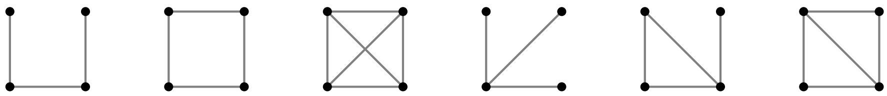
42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:12
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

For a family of graph $\mathcal{G}$, the general $\mathcal{G}$-Graph Modification problem ask whether we can modify a graph G into a graph in $\mathcal{G}$ by performing at most k simple operations. Typical examples of simple operations that are well-studied in the literature include vertex deletion, edge deletion, edge addition, or combination of edge deletion and addition. We call these problems $\mathcal{G}$-Vertex Deletion, $\mathcal{G}$-Edge Deletion, $\mathcal{G}$-Edge Addition, and $\mathcal{G}$-Edge Editing, respectively. By a classic result by Lewis and Yannakakis [16], $\mathcal{G}$-Vertex Deletion is NP-complete for all non-trivial hereditary graph classes. The situation is quite different for the edge modification problems. Earlier efforts for edge deletion problems [10, 19], though having produced fruitful concrete results, shed little light on a systematic answer, and it was noted that such a generalization is difficult to obtain.

$\mathcal{G}$-Graph Modification problems have been extensively investigated for graph classes $\mathcal{G}$ that can be characterized by a finite set of forbidden induced subgraphs. We say that a graph is $\mathcal{H}$-free, if it does not contain any graph in $\mathcal{H}$ as an induced subgraph. For this special case, the $\mathcal{H}$-free Vertex Deletion problem is well understood. If $\mathcal{H}$ contains a graph on at least two vertices, then all of these problems are NP-complete, but admit $c^k n^{\mathcal{O}(1)}$ algorithm [3], where c is the size of the largest graph in $\mathcal{H}$ (the algorithms with running time $f(k)n^{\mathcal{O}(1)}$ are called fixed-parameter tractable (FPT) algorithms [7, 9]). On the other hand, the NP-hardness proof of Lewis and Yannakakis [16] excludes algorithms with running time $2^{o(k)}n^{\mathcal{O}(1)}$ under Exponential Time Hypothesis (ETH) [14]. Finally, as observed by Flum and Grohe [12] a simple application of sunflower lemma [11] gives a *kernel* with $\mathcal{O}(k^c)$ vertices, where c is again the size of the largest graph in $\mathcal{H}$. A kernel is a polynomial time preprocessing algorithm which outputs an equivalent instance of the same problem such that the size of the reduced instance is bounded by some function $f(k)$ that depends only on k. We call the function $f(k)$ the size of the kernel. It is well-known that any problem that admits an FPT algorithm admits a kernel. Therefore, for problems with FPT algorithms one is interested in polynomial kernels, i.e., kernels where size upper bounded by a polynomial function.

For the edge modification problems, the situation is more complicated. While all of these problems also admit $c^k n^{\mathcal{O}(1)}$ time algorithm, where c is the maximum number of edges in a graph in $\mathcal{H}$ [3], the P vs NP dichotomy is still not known. Only recently Aravind et al. [1] gave the dichotomy for the special case when $\mathcal{H}$ contains precisely one graph H [1]. From the kernelization point of view, the situation is even more difficult. The reason is that deleting or adding an edge to a graph can introduce a new copy of H and this might further propagate. Hence, we cannot use the sunflower lemma to reduce the size of the instance. Cai asked the question whether H-free Edge Deletion admits a polynomial kernel for all graphs H [2]. Kratsch and Wahlström [15] showed that this is probably not the case and gave a graph H on 7 vertices such that H-free Edge Deletion and H-free Edge Editing does not admit a polynomial kernel unless coNP $\subseteq$ NP/poly. Consequently, it was shown that this is not an exception, but rather a rule [4, 13]. Indeed the result by Cai and Cai [4] shows that H-free Edge Deletion, H-free Edge Addition, and H-free-Edge Editing do not admit a polynomial kernel whenever H or its complement is a path or a cycle with at least 4 edges or a 3-connected graph with at least 2 edges missing. This suggests that actually the H-free modification problems with a polynomial kernels are rather rare and only for small graphs H. For the graphs on 4 vertices the kernelization of H-free edge modification problems was open for last two graphs and their complements (see Table 1), namely paw and claw, and Cao et al. [6] conjectured that all of these problems admit polynomial kernels.



(a) P_4 **(b)** C_4 **(c)** K_4 **(d)** claw **(e)** paw **(f)** diamond

Figure 1 Graphs on 4 vertices, their complements are omitted.

H	deletion	addition	editing
K_4	$\mathcal{O}\left(k^4\right)$ [5]	trivial	$\mathcal{O}\left(k^4\right)$ [5]
P_4	$\mathcal{O}\left(k^3\right)$ [13]	$\mathcal{O}\left(k^3\right)$ [13]	$\mathcal{O}\left(k^3\right)$ [13]
diamond	$\mathcal{O}\left(k^3\right)$ [18]	trivial	$\mathcal{O}\left(k^8\right)$ [6]
paw	$\mathcal{O}\left(k^3\right)$ [this paper]	$\mathcal{O}\left(k^3\right)$ [this paper]	$\mathcal{O}\left(k^6\right)$ [this paper]
claw	open	open	open
C_4	no [13]	no [13]	no [13]

Table 1 The kernelization results of H-free edge modification problems for H being 4-vertex graphs. Note that for a complement of H, the rows with deletion and addition are swapped, but otherwise the same results hold.

In this paper, we give kernels for the first of the two remaining graphs, namely the paw.

1.1 Brief Overview of the Algorithm

Our main result is a polynomial kernel for Paw-free-Edge Editing. The key to obtain the kernel is a structural theorem by Olariu [17] that states that every connected paw-free graph is either triangle-free or complete multipartite graph. We start our kernelization algorithm by finding a greedy edge-disjoint packing of paws in G. This clearly contains at most k paws and hence at most $4k$ vertices. Let us denote the set of these vertices by S. The goal now is to bound the vertices in $G - S$. Bounding the vertices belonging to the complete multipartite components of $G - S$ is rather simple. We show that every vertex in S is adjacent to at most 1 complete multipartite component and for each multipartite component, we can reduce the size of each part as well as the number of these parts to $\mathcal{O}(k)$, else we can always find an irrelevant vertex that does not appear in any solution. The triangle-free part is more tricky. The difficulty comes from the fact that actually instead of keeping this part of the graph triangle-free, the optimal solution might want to add some edges to make it complete multipartite. We are however able to show that there is always optimal solution that keeps the vertices at distance at least 5 from S in a triangle-free component. This structural claim helps us in looking for solution which are not too far away from S "in some sense". Moreover, after some preprocessing of the instance, we can also show that the vertices with more than $4k + 6$ neighbors inside the triangle-free components of $G - S$ cannot end up inside a complete multipartite component. It means that we can mark the relevant vertices in triangle-free components as follows. Set $S_0 := S$ and for every $i < 5$, let S_{i+1} be the set obtained by marking for each vertex of S_{i+1}, $4k + 6$ neighbors at distance $i + 1$ from S. The set of vertices marked is then $\mathcal{O}\left(k^6\right)$. Finally, we can remove the vertices of triangle-free components which have not been marked. This is safe because these vertices are either too far from S to belong to a complete multipartite component, or every way to connect these vertices to S use vertices that can't end up in a complete multipartite component of the reduce instance because of the degree condition. This gives us the desired kernel.

2 Preliminaries

We assume familiarity with the basic notations and terminologies in graph theory. We refer the reader to the standard book by Diestel [8] for more information. Given a graph G and a set of pairs of vertices $A \in V(G)^2$, we denote by $G\Delta A$ the graph whose set of vertices is $V(G)$ and set of edges is the symmetric difference of $E(G)$ and A.

Parameterized Algorithms and Kernelization: For a detailed illustration of the following facts the reader is referred to [7, 9]. A *parameterized problem* is a language $\Pi \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet; the second component k of instances $(I, k) \in \Sigma^* \times \mathbb{N}$ is called the *parameter*. A parameterized problem Π is *fixed-parameter tractable* if it admits a *fixed-parameter algorithm*, which decides instances (I, k) of Π in time $f(k) \cdot |I|^{\mathcal{O}(1)}$ for some computable function f.

A *kernelization* for a parameterized problem Π is a polynomial-time algorithm that given any instance (I, k) returns an instance (I', k') such that $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi$ and such that $|I'| + k' \leq f(k)$ for some computable function f. The function f is called the *size* of the kernelization, and we have a polynomial kernelization if $f(k)$ is polynomially bounded in k. It is known that a parameterized problem is fixed-parameter tractable if and only if it is decidable and has a kernelization. However, the kernels implied by this fact are usually of superpolynomial size.

A *reduction rule* is an algorithm that takes as input an instance (I, k) of a parameterized problem Π and outputs an instance (I', k') of the same problem. We say that the reduction rule is *safe* if (I, k) is a *yes*-instance if and only if (I', k') is a *yes*-instance. In order to describe our kernelization algorithm, we present a series of reduction rules.

We will need the following result describing the structure of paw-free graphs [17].

▶ **Theorem 1.** *G is a paw-free graph if and only if each connected component of G is triangle-free or complete multipartite.*

To make a clear distinction between these two cases, we will say that a graph is a complete multipartite graph if it contains at least three parts. In particular, it contains a triangle.

3 Reduction Rules

From now on (G, k) will be an instance of paw-free editing and we assume $k > 0$. Let us first describe two rules which can be safely applied.

▶ **Reduction Rule 1.** *If X is an independent set of $k+3$ vertices with the same neighborhood, remove a vertex $x \in X$ from the graph.*

Proof of Safeness. Suppose (G, k) is an instance of the paw-free editing problem and X is an independent set of $k + 3$ vertices with the same neighborhood. Let G' be the graph obtained by removing a vertex of X. We need to show that (G', k) has a solution if and only if (G, k) has one. Since G' is a subgraph of G, it is clear that if (G, k) has a solution, then so does (G', k). Let A be a solution to (G', k) and assume $G\Delta A$ contains a paw x_1, x_2, x_3, x_4 with x_1, x_2, x_3 being a triangle and x_4 being adjacent to x_3. Because A is a solution to (G', k), it means that one of the x_i must be the vertex x that we removed from G. Moreover, at most two of the other vertices of X belong to the paw, as x is adjacent to at least one vertex and X is an independent set. If only one other vertex of X belongs to it, consider the other $k + 1$ vertices of X which are not in the paw. They all have the same neighborhood in the paw as x, so A must contain for each of them at least one edge with the paw, or we could replace x with this vertex in the paw, which contradicts the fact that A is a solution of (G', k). However, since A is smaller than $k+1$ we reach a contradiction. If two other vertices of X belong to the paw, then it means that $x = x_4$ and these vertices are x_1 and x_3. Moreover it means that the edge x_1x_3 must be edited as X is an independent set. In that case, consider the other k vertices of X which are not in the paw. Again, for each of them, the solution must contains an edge with the paw, but since $|A \setminus (x_1x_3)| < k$, we also reach a contradiction. Overall this implies that Rule 1 is safe. ◀

Following analogous arguments for the case when X induces a complete multipartite graph with at least $k+5$ parts, we also obtain safeness of the following rule.

▶ **Reduction Rule 2.** *If X is a complete multipartite subgraph with $k+5$ parts having the same neighborhood outside of X, then remove the smallest part of X from the graph.*

Proof of Safeness. Suppose (G, k) is an instance of the paw-free editing problem and X is a complete multipartite subgraph with $k+5$ parts having the same neighborhood outside of X. Let G' be the graph obtained by removing the smallest part P of X. We need to show that (G', k) has a solution if and only if (G, k) has one. Let A be a solution to (G', k) and assume $G\Delta A$ contains a paw x_1, x_2, x_3, x_4 with x_1, x_2, x_3 being a triangle and x_4 being adjacent to x_3. Because A is a solution to (G', k), it means that one of the x_i must belong to P. Moreover, since the vertices in P have exactly the same neighborhood in G and they form an independent set, this paw can contain at most one vertex from P. Let us call x this vertex. Since X consists of $k+5$ parts, it means that there exists $k+1$ parts different from P and without a vertex in this paw. However we know that any vertex in these parts has the exact same neighborhood as x inside the paw. This means that each of these vertices must be adjacent in A to the paw, or we can replace x with a vertex belonging to G', which is a contradiction. However, since there is at least $k+1$ of these vertices and $|A| = k$, we reach a contradiction. ◀

Note that if there exists a set X for which Reduction Rule 1 can be applied, then this set can be found in polynomial time. Therefore from now on we assume that (G, k) is an instance where Reduction Rule 1 cannot be applied. Let $\mathcal{H}$ be a maximal packing of edge-disjoint paws and S the set of vertices appearing in $\mathcal{H}$.

We will now introduce two new rules.

▶ **Reduction Rule 3.** *If there is a pair of adjacent vertices s_1, s_2 with $4k+6$ common neighbors in the triangle-free components of $G - S$, then remove the edge s_1, s_2 and set $k := k - 1$.*

The soundness of Reduction Rule 3 is implied by the following Lemma:

▶ **Lemma 2.** *Suppose Reduction Rule 1 cannot be applied anymore and let s_1, s_2 be two adjacent vertices. If there are more than $4k+6$ vertices belonging to the triangle-free components of $G - S$ adjacent to both s_1 and s_2, then either (G, k) is a* no*-instance, or any solution uses the edge s_1s_2.*

Proof. Suppose there is a solution A not using the edge s_1s_2. Because s_1 and s_2 have $4k+6$ common neighbors in G, it means that they belong to a triangle and thus to a complete multipartite component of $G\Delta A$. Because $|A| = k$, we know that at least $2k+6$ of the common neighbors of s_1 and s_2 are not adjacent to any edge in A. This means that these vertices belong to the same component in $G\Delta A$, and moreover they can only be in two different parts as they belong to the triangle-free components of $G - S$. This means that $k+3$ of these vertices belong to the same part of a complete multipartite component of

$G\Delta A$ and since they are not incident to any edge in A, they have the same neighborhood in G. Therefore, we could have applied Reduction Rule 1. ◀

▶ **Reduction Rule 4.** *If C is a complete multipartite component of $G - S$ and P_1 is a part of C with more than $3k + 3$ vertices, then remove all the edges between the other parts of C and decrease k by the amount of edges removed. If this amount is greater than k, answer* no.

The fact that Reduction Rule 4 is safe is implied by the following Lemma:

▶ **Lemma 3.** *Suppose Reduction Rule 1 cannot be applied anymore and assume C is a complete multipartite component of $G - S$. If one part of C is larger than $3k + 3$, then either (G, k) is a* no-*instance, or any solution will remove all the edges between the other parts of C.*

Proof. Let P_1 be a part of C of size greater than $3k+3$ and let s_1, s_2 be two adjacent vertices of $C - P_1$. Let A be a solution of size at most k which does not use the edge s_1s_2. A is incident to at most $2k$ vertices, so it means that at least $k + 3$ vertices of P_1 are not incident to any edge of A. Moreover, since s_1s_2 is not in A, these $k + 3$ vertices belong to the same part of a complete multipartite component of $G\Delta A$ and thus have the same neighborhood in G. This is a contradiction, as Reduction Rule 1 cannot be applied anymore. ◀

Note also that if Reduction Rules 3 and 4 can be applied, then it is possible to do it in polynomial time. From now on assume that none of these rules can be applied.

4 Bounding the Complete Multipartite Components

The next two lemmas allow us to bound the number of vertices belonging to complete multipartite components of $G - S$.

▶ **Lemma 4.** *Let C denote a complete multipartite component of $G - S$. If $|C| \geq (3k + 3)(3k + 5)$, then either Reduction Rule 2 can be applied or (G, k) is a* no-*instance. Moreover, if Reduction Rule 2 can be applied, then it can be done in polynomial time.*

Proof. Because Reduction Rule 4 cannot be applied, we have that every part of C contains at most $(3k + 3)$ vertices. Suppose now that C consists of more than $3k + 5$ parts. If (G, k) is a *yes*-instance, then the solution can only be adjacent to at most $2k$ of these parts. The complete multipartite graph consisting of the $k + 5$ parts not adjacent to the solution is then a candidate to apply Reduction Rule 2.

Note that to find the multipartite subgraph to apply Reduction Rule 2, we only have to check for each part if the vertices in this part have the same neighborhood outside of C, and for the part that do, find a maximum set of parts with the same neighborhood. ◀

▶ **Lemma 5.** *For any $s \in S$, s is adjacent to at most one complete multipartite component of $G - S$.*

Proof. Suppose $s \in S$ is adjacent to two complete multipartite components C_1 and C_2. Let x be a vertex of C_1 adjacent to s. By definition of C_1, there exist vertices y and z in C_1 such that x, y, z is a triangle. This implies that one of y and z has to be adjacent to s or it would yield a paw without any edge in S which is not possible by definition of $\mathcal{H}$.

Suppose now that y is adjacent to s (the case x is adjacent to s is identical). Now let c_2 be a vertex of C_2 adjacent to s. Because C_1 and C_2 are two different components, c_2 cannot be adjacent to either c_1 or y, which means that s, c_1, c_2 and y form paw without any edge in S, a contradiction. ◀

The next section is devoted to proving that, if there exists a solution A, then we can assume that any complete multipartite component of $G\Delta A$ only contains vertices at distance 5 from S.

5 Bounding the Diameter of Relevant Vertices

Let A denote a solution such that the sizes of the multipartite components in $G\Delta A$ are minimal. In this section, C will denote a complete multipartite component of $G\Delta A$, and $C_1, C_2, \ldots, C_r$ the parts of C. For any $i \in [r]$ and j, let $C_{i,j}$ denote the set of vertices of C_i which are at distance j of S and $\overline{C_{i,j}} = \bigcup_{t \neq i} C_{t,j}$.

▶ **Lemma 6.** *For any $j \geq 4$, and any $i \in [r]$, if $C_{i,0} \cup C_{i,1}$ is non empty, then $C_{i,j}$ is.*

Proof. Suppose $C_{i,0} \cup C_{i,1}$ and $C_{i,j}$ are non empty.

Because $j \geq 4$, we know that $E(C_{i,j}, \overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}})$ is empty. This implies that A contains all the pairs in $C_{i,j} \times (\overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}})$. However, vertices in $C_{i,j}$ can only be adjacent to vertices at distance $i, i-1$ and $i+1$ from S, thus replacing all the edges in $C_{i,j} \times (\overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}})$ by the pairs in $E(C_{i,j}, \overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}})$ would also give a solution by disconnecting the vertices in $C_{i,j}$ from C. However, since A is chosen such that $|C|$ is minimal, it implies that: $|C_{i,j}| \times |\overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}}| \leq |E(C_{i,j}, \overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}})|$. However, $|E(C_{i,j}, \overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}})| \leq |C_{i,j}| \times |\overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}}|$ and thus:

$$|\overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}}| \geq |\overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}}|$$

.

Now setting

$$A' := \left(A \cup E(C_{i,0} \cup C_{i,1}, \overline{C_{i,0}} \cup \overline{C_{i,1}} \cup \overline{C_{i,2}})\right) \setminus \left((C_{i,0} \cup C_{i,1}) \times (\overline{C_{i,j-1}} \cup \overline{C_{i,j}} \cup \overline{C_{i,j+1}})\right)$$

gives an optimal solution where C doesn't contain $C_{i,0} \cup C_{i,1}$ and whose value is as good as A, a contradiction. ◀

For any j, let $S_j = \bigcup_{i \in [r]} C_{i,j}$. In other word, S_j is the set of vertices of C at distance j from S. The main implication of Lemma 6 is that, if S_j is not empty for $j \geq 4$, then A contains all the pair $S_i \times (S_0 \cup S_1)$. Indeed, it shows that vertices in S_j and $S_0 \cup S_1$ belongs to different parts and thus must be adjacent in $G\Delta A$. However, just by considering the distance to S in G, these vertices cannot be adjacent in G, and thus these pairs must be in A. This allows us to prove the following lemma.

▶ **Lemma 7.** *For any $j \geq 5$, S_j is empty.*

Proof. Suppose S_4 and S_5 are non empty. By Lemma 6, we know that the vertices in S_5 and $S_0 \cup S_1$ belong to different parts of the complete multipartite component. This implies that A contains $S_5 \times (S_0 \cup S_1)$. However, removing these pairs from A, as well as all pairs containing a vertex of C at distance more than 6 from S, and adding $E_G(S_5, S_4)$ also yields a solution by disconnecting S_5 from the multipartite component. By optimality of A, this implies that $E_G(S_5, S_4) \geq |S_5||S_0 \cup S_1|$ and thus $|S_4| \geq |S_1 \cup S_0|$. Now again by Lemma 6, we have that A contains $S_4 \times (S_0 \cup S_1)$. However, $|S_4| \geq |S_1 \cup S_0|$ so it means that $|S_1 \cup S_0|^2 \leq |S_4||S_1 \cup S_0|$. Let A' be the solution obtained from A by disconnecting S_1 from S_0 and removing all pairs adjacent to the sets S_j for $j \geq 2$. Because $|S_1 \cup S_0|^2 \leq |S_4||S_1 \cup S_0|$, we have that $|A'| \leq |A|$ and the multipartite component containing S_0 is strictly smaller in $G\Delta A'$ than in $G\Delta A$ while the other remain exactly the same, which is a contradiction. ◀

6 Triangle-Free Components

Before proving our main result let us prove the following lemma, which will be useful in bounding the number of vertices outside of S.

▶ **Lemma 8.** *If $x \in G$ has at least $4k+6$ neighbors belonging to triangle-free components of $G - S$, then there is no solution A such that x belongs to a complete multipartite component of $G\Delta A$.*

Proof. Let T denote the set of neighbors of x belonging to triangle-free components of $G - S$. Suppose x belongs to a complete multipartite component C of $G\Delta A$. First note that at least $2k+6$ of the vertices of T will not be adjacent to any edge of A, which means that their neighborhood in G and $G\Delta A$ are the same and they belong to C in $G\Delta A$. Now because the vertices of T belong to triangle-free components, it means that these $2k+6$ vertices can only belong to two different parts of this multipartite component. In particular, at least $k+3$ of those belong to the same part and thus have the exact same neighborhood in $G\Delta A$ and thus in G. This means that Reduction Rule 1 can be applied, which is a contradiction. ◀

▶ **Lemma 9.** *Suppose (G, k) is a* yes*-instance. Then there exists a set S' of at most $(4k+6)4k$ vertices such that if $x \notin S'$ belongs to a triangle-free component of $G - S$, then x doesn't belong to any triangle in G using only one vertex of S. Moreover, there is a polynomial time algorithm that either find this set or concludes that (G, k) is a* no*-instance.*

Proof. Let x be a vertex belonging to a triangle-free component C of $G - S$. Suppose that x belongs to a triangle using only one vertex s of S and another vertex y of C. Note first that C is the only component of $G - S$ adjacent to s or we would have a paw using edges not in S. Suppose now that $t \in C$ is adjacent to x. Then t must be adjacent to either y or s or it would yield a paw using no edge in S. Thus, since C is triangle free, t must be adjacent to s. The same argument would show that any vertex adjacent to t in C must be adjacent to s and thus the whole component C is adjacent to x.

Let $\mathcal{M}$ be a maximal matching in C. If $\mathcal{M}$ consists of more than k edges, then it means that any solution A to the instance (G, k) puts s in a complete multipartite component. In particular if $|C| \geq 4k+6$, as $C \subseteq N(x)$ and $|A| \geq k$, we have that $2k+6$ of the vertices of C are not adjacent to any edge of A and belong to the same complete multipartite component as s. Moreover, these vertices can only belong to two different parts of this complete multipartite component (or we would have a triangle in C), and thus $k+3$ of them belong to the same part. However, since their neighborhood in G and $G\Delta A$ are identical, it means we could have applied Reduction Rule 1, so (G, k) is a *no*-instance. So let C' be defined as the vertices of $\mathcal{M}$ if $|\mathcal{M}| \leq k$ and the full set C if $\mathcal{M}$ if $|\mathcal{M}| \geq k$. Note that in the case where $|\mathcal{M}| \leq k$, the vertices in $C \setminus C'$ only have neighbors in $S \cup C'$.

Let S' be the union of the C' for every such component C where there exists a vertex which belong to a triangle using one vertex from $s \in S$. Note that the number of those components C is bounded by $|S|$. Indeed, s cannot be adjacent to any other component of $G - S$ or we have a paw using no edge from S which is not possible. This implies that $|S'| \leq |S|(4k+6)$. ◀

7 Main Result

We are now ready to prove our main theorem.

▶ **Theorem 10.** *The paw-free editing problem has a kernel on $\mathcal{O}\left(k^6\right)$ vertices*

Proof. Let (G,k) be an instance of paw-free editing. The algorithm first apply Reduction Rule 1 repetitively. Once Reduction Rule 1 cannot be applied anymore, the algorithm computes $\mathcal{H}$ a maximal packing of edge-disjoint paws. If $\mathcal{H}$ consists of more than k paws, answer *no*. If this is not the case, let S be the set of vertices belonging to a paw of $\mathcal{H}$. $|S| \leq 4k$. Then the algorithm apply Reduction Rules 3 and 4 until either $k < 0$, in which case it answers *no*, or they cannot be applied anymore.

Because $\mathcal{H}$ is maximal, Theorem 1 implies that the components $G - S$ are either triangle-free or complete multipartite. Let C be a complete multipartite component. If $|C| \geq (3k+3)(3k+5)$, then Lemma 4 implies that the algorithm can apply Reduction Rule 2 or answer *no*. Moreover Lemma 5 implies that the number of complete multipartite components adjacent to S is bounded by $|S|$. Overall this implies that the number of vertices contained in complete multipartite components of $G - S$ adjacent to S is bounded by $4k(3k+3)(3k+5)$, or it is possible to apply Reduction Rule 2.

By applying Lemma 9, we either find out that (G,k) is a *no*-instance or find a set S' of at most $(4k+6)4k$ vertices such that if $x \notin S'$ belongs to a triangle-free component of $G - S$, then x doesn't belong to any triangle in G using only one vertex of S.

Because Reduction Rule 3 cannot be applied anymore, it means that for every pair of adjacent vertices s_1, s_2 in S, the number of vertices in triangle-free components adjacent to both s_1 and s_2 is bounded by $4k+6$. This means that, if S'' denotes the set of vertices in a triangle-free component forming a triangle with 2 vertices of S, then $|S''| \leq |S|^2(4k+6)$.

Then we construct recursively sets $S_0, S_1, \dots S_6$ such that S_i is a subset of vertices of G at distance i from S as follows: First we set $S_0 := S$ and then, for every $i \in \{0, \dots, 5\}$, we define S_{i+1} by picking, for every vertex x of S_i, $4k+6$ neighbors of x at distance $i+1$ from S in G and belonging to a triangle-free component of $G - S$. Note that $|\bigcup S_i| = \mathcal{O}\left(k^6\right)$.

Let G' be the graph induced on G by S, S', S'' the S_i and all the complete multipartite components of $G - S$ adjancent to S. Note that, by construction of S' and S'', there is no triangle in G using a vertex which is not in G'. We claim that (G',k) has a solution if and only if (G,k) has a solution. As G' is a subgraph of G, it is clear that if (G,k) has a solution, then so does (G',k). Suppose now that (G',k) has a solution A, but (G,k) does not have a solution. In particular, it implies that $G\Delta A$ is not paw-free. Because of Lemma 7, we can assume that no complete multipartite component of $G'\Delta A$ has a vertex at distance 5 from S and that A is minimal. Let x_1, x_2, x_3, x_4 form a paw in $G\Delta A$, with x_1, x_2, x_3 being the triangle. One of the x_i must be a vertex which has not been marked during the construction of the S_i. Moreover, since G' contains all the triangles of G, it means that x_1, x_2 and x_3 belong to G' and x_4 doesn't. It also means that x_1, x_2 and x_3 belong to a complete multipartite component of $G'\Delta A$ and x_4 is adjacent to one of these vertices, say x_1. Since x_1 is at distance less than 5 from S, it means that during the marking process x_4 was not marked for x_1. But this means that x_1 has more than $4k+6$ neighbors in triangle-free components of $G' - S$. However, Lemma 8 implies that x_1 cannot belong to a complete multipartite component of $G'\Delta A$, which is a contradiction. ◀

8 Better Bounds for Deletion and Addition

In this section, we provide better kernels for paw-deletion and paw-addition. Let us start with the deletion problem, where the proof is quite similar to the one of Theorem 10, with the difference that we only keep vertices of the triangle-free components which are at distance one from S.

▶ **Theorem 11.** *The paw-free deletion problem admits a kernel of size $\mathcal{O}\left(k^3\right)$.*

Proof. Let (G, k) be an instance of paw-free deletion. First note that Reduction Rules 1–4 are still safe in this context, and Lemma 8 still applies. Therefore the algorithm applies Reduction Rule 1 until it cannot be applied anymore. It then computes $\mathcal{H}$ a maximal packing of edge-disjoint paws. If $\mathcal{H}$ consists of more than k paws, answer *no*. If this is not the case, let S be the set of vertices belonging to a paw of $\mathcal{H}$. $|S| \leq 4k$. Then the algorithm apply Reduction Rules 3 and 4 until either $k < 0$, in which case it answers *no*, or they cannot be applied anymore.

Again, by possibly applying Reduction Rule 2, we can assume that the set of vertices in all the multipartite components of $G - S$ adjacent to S is smaller than $4k(3k + 3)(3k + 5)$. By applying Lemma 9, we either find out that (G, k) is a *no*-instance or find a set S' of at most $(4k + 6)4k$ vertices such that if $x \notin S'$ belongs to a triangle-free component of $G - S$, then x doesn't belong to any triangle in G using only one vertex of S.

Because Reduction Rule 3 cannot be applied anymore, it means that for every pair of adjacent vertices s_1, s_2 in S, the number of vertices in triangle-free components adjacent to both s_1 and s_2 is bounded by $4k + 6$. This means that, if S'' denote the set of vertices in a triangle-free component, forming a triangle with 2 vertices of S, then $|S''| \leq |S|^2(4k + 6)$.

Note also that Lemma 8 still applies, and let S_1 be the set obtained by picking for every vertex s in S, $4k + 6$ neighbors in triangle-free components of $G - S$.

Let G' be the graph induced on G by S, S', S'', S_1, as well as all the vertices on complete multipartite components of $G - S$. We want to show that (G, k) has a solution if and only if (G', k) has a solution. Let A be a solution of (G', k) and suppose $G\Delta A$ has a paw x_1, x_2, x_3, x_4, with x_1, x_2, x_3 being a triangle and x_4 being adjacent to x_3. Because of the choice of the sets S' and S'', all the triangle of G are contained in G'. Note also that, since the solution can only remove edges, x_1, x_2, x_3 is a triangle in G. This implies that $x_3 \in S$ and x_4 was not picked for the $4k + 6$ neighbors of x_3. In particular, this means that x_3 has $4k + 6$ neighbors which belong to a triangle-free component of $G' - S$ in G' and thus by Lemma 8, x_3 cannot belong to a complete multipartite component of $G'\Delta A$. However, since x_1, x_2 and x_3 form a triangle in $G'\Delta A$, we reach a contradiction. ◀

▶ **Theorem 12.** *The paw-free addition problem admits a kernel of size* $\mathcal{O}\left(k^3\right)$.

Proof. Again, Reduction Rules 1–4 are still safe in this context, with the difference for Rules 3 and 4 that, instead of removing edges and decreasing k, we can directly conclude that (G, k) is a *no*-instance. Note also that a paw-free connected component can safely be removed from the graph.

So the algorithm start by removing all the paw-free components of G and applying Reduction Rule 1 until it cannot be applied anymore. It then computes $\mathcal{H}$ a maximal packing of edge-disjoint paws. If $\mathcal{H}$ consists of more than k paws, answer *no*. If this is not the case, let S be the set of vertices belonging to a paw of $\mathcal{H}$. $|S| \leq 4k$. From now on we can assume that Rules 3 and 4 cannot be applied.

Again, by possibly applying Reduction Rule 2, we can assume that the set of vertices in all the multipartite components of $G - S$ adjacent to S is smaller than $4k(3k + 3)(3k + 5)$.

Consider a connected component C_1 of G. This component cannot be paw-free, or the algorithm would have removed it from the graph. So let $S_1 = C_1 \cap S$ and R_1 the vertices of C_1 contained in triangle-free component of $G - S$. Because C_1 is not triangle-free, it means that any solution A to (G, k) leaves C_1 as a complete multipartite component. In particular, it implies that R_1 is smaller than $4k + 6$. Indeed, if R_1 is bigger than $4k + 6$, then $2k + 6$ vertices will have the same neighborhood in $G\Delta A$ as in G. Moreover, since R_1 is triangle-free, it means that these vertices belong to at most 2 parts of the complete multipartite component. This implies that at least $k+3$ of these vertices belong to the same part and Rule 1 applies. Moreover, since G has at most k connected component which are not paw-free, it implies that the set of vertices contained in triangle-free components of $G-S$ is smaller than $(4k+6)k$.

Overall, it implies that our reduced instance has size at most $4k(3k+3)(3k+5)+(4k+6)k+4k=\mathcal{O}\left(k^3\right)$, which ends the proof. ◀

9 Conclusion

In this paper we studied Paw-free-Edge Editing and gave a polynomial kernel of size $\mathcal{O}\left(k^6\right)$. The only unresolved graphs H on 4 vertices, for which the kernelization complexity of H-free-Edge Editing problem remains open is claw. In fact, for this problem even the kernelization complexity of H-Edge Deletion and H-Edge Addition remain open. Settling the kernelization complexity might require using the power of structure theorem of claw free graphs. Thus, a natural start here could be looking at editing/deletion/addition to basic graphs, on which structure theorem of claw free graphs is built. We leave these as natural directions to pursue.

References

1 N. R. Aravind, R. B. Sandeep, and Naveen Sivadasan. Dichotomy results on the hardness of h-free edge modification problems. *SIAM J. Discrete Math.*, 31(1):542–561, 2017. doi:10.1137/16M1055797.

2 Hans L Bodlaender, Leizhen Cai, Jianer Chen, Michael R Fellows, Jan Arne Telle, and Dániel Marx. Open problems in parameterized and exact computation-iwpec 2006. *UU-CS*, 2006, 2006.

3 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.

4 Leizhen Cai and Yufei Cai. Incompressibility of h-free edge modification problems. *Algorithmica*, 71(3):731–757, 2015. doi:10.1007/s00453-014-9937-x.

5 Yufei Cai. Polynomial kernelisation of H-free edge modification problems. Mphil thesis, Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China, 2012.

6 Yixin Cao, Ashutosh Rai, R. B. Sandeep, and Junjie Ye. A polynomial kernel for diamond-free editing. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 10:1–10:13, 2018. doi:10.4230/LIPIcs.ESA.2018.10.

7 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

8 R. Diestel. *Graph Theory, 4th Edition*. Springer, 2012.

9 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

10 Ehab S. El-Mallah and Charles J. Colbourn. The complexity of some edge deletion problems. *IEEE Transactions on Circuits and Systems*, 35(3):354–362, 1988. doi:10.1109/31.1748.

11 Paul Erdös and Richard Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 1(1):85–90, 1960.

12 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

13 Sylvain Guillemot, Frédéric Havet, Christophe Paul, and Anthony Perez. On the (non-)existence of polynomial kernels for P_l-free edge modification problems. *Algorithmica*, 65(4):900–926, 2013. doi:10.1007/s00453-012-9619-5.

14 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

15 Stefan Kratsch and Magnus Wahlström. Two edge modification problems without polynomial kernels. *Discrete Optimization*, 10(3):193–199, 2013. doi:10.1016/j.disopt.2013.02.001.

16 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.

17 Stephan Olariu. Paw-free graphs. *Information Processing Letters*, 28(1):53 – 54, 1988. URL: http://www.sciencedirect.com/science/article/pii/0020019088901433, doi:https://doi.org/10.1016/0020-0190(88)90143-3.

18 R. B. Sandeep and Naveen Sivadasan. Parameterized lower bound and improved kernel for diamond-free edge deletion. In *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, pages 365–376, 2015. doi:10.4230/LIPIcs.IPEC.2015.365.

19 Mihalis Yannakakis. Edge-deletion problems. *SIAM J. Comput.*, 10(2):297–309, 1981. doi:10.1137/0210021.