Kernelization Algorithms for the Eigenvalue Deletion Problems

Ajinkya Gaikwad, Hitendra Kumar and Soumen Maity

Indian Institute of Science Education and Research, Pune, India
ajinkya.gaikwad@students.iiserpune.ac.in; soumen@iiserpune.ac.in

Abstract. Given a graph $G = (V, E)$ and an integer $k \in \mathbb{N}$, we study 2-Eigenvalue Vertex Deletion (2-EVD), where the goal is to remove at most k vertices such that the adjacency matrix of the resulting graph has at most 2 eigenvalues. It is known that the adjacency matrix of a graph has at most 2 eigenvalues if and only if the graph is a collection of equal sized cliques. So 2-Eigenvalue Vertex Deletion amounts to removing a set of at most k vertices such that the resulting graph is a collection of equal sized cliques. The 2-Eigenvalue Edge Editing (2-EEE), 2-Eigenvalue Edge Deletion (2-EED) and 2-Eigenvalue Edge Addition (2-EEA) problems are defined analogously. We provide a kernel of size $\mathcal{O}(k^3)$ for 2-EVD. For the problems 2-EEE and 2-EED, we provide kernels of size $\mathcal{O}(k^2)$. Finally, we provide a linear kernel of size $6k$ for 2-EEA. We thereby resolve three open questions listed by Misra et al. (ISAAC 2023) concerning the complexity of these problems parameterized by the solution size.

Keywords: Parameterized Complexity · FPT · Eigenvalue · Uniform Cluster Graphs

1 Introduction

Numerous algorithmic challenges involving graphs can be framed as tasks of modifying a graph to meet specific criteria. Notably, over the last three decades, these graph modification tasks have been a significant source of inspiration for innovating new methodologies in parameterized algorithms and complexity theory. This paper delves into a particular graph modification problem known as Uniform Cluster Vertex Deletion, along with several related variations of the same problem. Previously explored in [8], the complexity of reducing the count of distinct eigenvalues of a graph by either removing vertices or editing edges was examined. Notably, it can be observed that an adjacency matrix of graph possesses at most two distinct eigenvalues if and only if it comprises of disjoint unions of equally sized cliques. Our notation aligns with that of the aforementioned paper. Let $G = (V, E)$ be a simple undirected graph, where V denotes the vertex set of G and E denotes the edge set of G. We use n to denote $|V|$. For $u \in V$, we define $N(u) = \{v \in V : (u, v) \in E\}$ and $N[u] = N(u) \cup \{u\}$. The *degree* of $u \in V$ is $|N(u)|$ and denoted by $d_G(v)$. A *clique* C in an undirected graph $G = (V, E)$ is a subset of the vertices $C \subseteq V$ such that every two distinct vertices are adjacent. A *cluster graph* is a graph where every component is a clique. Observe that a graph is a cluster graph if and only if it does not have an induced P_3, that is, an induced path on three vertices. Let $U \subseteq V$ be a subset of vertices of G and $F \subseteq \binom{V}{2}$ be a subset of pairs of vertices of G. The subgraph induced by $U \subseteq V$ is denoted by $G[U]$. We define $G - U = G[V \setminus U]$, $G - F = (V, E \setminus F)$, $G + F = (V, E \cup F)$ and $G \triangle F = (V, E \triangle F)$. Here, $E \triangle F$ is the symmetric difference between E and F. If $U = \{u\}$ or $F = \{e\}$ then we simply write $G - u$, $G - e$ and $G + e$ for $G - U$, $G - F$ and $G + F$, respectively. It is known that the spectrum of the adjacency matrix of a graph can be computed in polynomial time. The following lemma is crucial to our discussions:

Lemma 1. *[2,4] The adjacency matrix of a graph G has at most two distinct eigenvalues if and only if G is a disjoint union of equal sized cliques.*

We refer to Appendix A and [1,3] for details on parameterized complexity. Our goal is to present the parameterized complexity landscape of the following problems:

2-Eigenvalue Vertex Deletion (2-EVD)
Input: An undirected graph $G = (V, E)$, and a positive integer k.
Question: Is there a subset $S \subseteq V$ with $|S| \leq k$ such that $G - S$ is a collection of equal sized cliques?

2-Eigenvalue Edge Editing (2-EEE)
Input: An undirected graph $G = (V, E)$, and a positive integer k.
Question: Is there a subset $F \subseteq \binom{V(G)}{2}$ with $|F| \leq k$ such that $G \triangle F = (V, E \Delta F)$ is a collection of equal sized cliques?

2-Eigenvalue Edge Deletion (2-EED)
Input: An undirected graph $G = (V, E)$, and a positive integer k.
Question: Is there a subset $F \subseteq E$ with $|F| \leq k$ such that $G - F = (V, E \setminus F)$ is a collection of equal sized cliques?

2-Eigenvalue Edge Addition (2-EEA)
Input: An undirected graph $G = (V, E)$, and a positive integer k.
Question: Is there a subset $F \subseteq \binom{V(G)}{2}$ with $|F| \leq k$ such that $G + F = (V, E \cup F)$ is a collection of equal sized cliques?

1.1 Our results

In this paper, we study 2-EVD, 2-EEE, 2-EED, and 2-EEA from the parameterized complexity point of view. Clearly, one can study the above problems for any fixed number of distinct eigenvalues. We give a kernel of size $\mathcal{O}(k^3)$ for 2-EVD. For the problems 2-EEE and 2-EED (see Appendix D), we provide kernels of size $\mathcal{O}(k^2)$. Finally, we provide a linear kernel of size $6k$ for the 2-EEA problem (see Appendix C). We thereby resolve three open questions listed by Misra et al. [8] concerning the complexity of these problems parameterized by solution size.

1.2 Review of previous work

As mentioned in [8], the problem of modifying the graph to reduce the count of distinct eigenvalues to r of the corresponding adjacency matrix was first brought up in [7]. In [8], Misra et al. considered classical and parameterized complexity of this problem. The paper studied mainly four possible operations which are vertex deletion, edge deletion, edge addition and edge editing. For the special case of $r = 2$, the vertex deletion variant was shown to be NP-complete even on triangle-free and $3d$-regular graphs for any $d \geq 2$, and also NP-complete on d-regular graphs for any $d \geq 8$. Moreover, the edge deletion, addition, and editing variants were proved to be NP-complete for $r = 2$ case. Furthermore, for any fixed $r \geq 3$, they showed that r-EVD is NP-complete on bipartite graphs. Also, the 2-EEA was shown to be NP-complete when the input is either a cluster graph, a forest, or a collection of cycles. Apart from studying the classical complexity, the authors also studied and provided numerous results in the realm of parameterized complexity. They gave single exponential FPT algorithms for the 2-EVD and 2- EED problems. The paper also presented a quadratic kernel for 2-EEA.

The clustering problems hold high importance in the complexity theory as they can model multiple scenarios. One more way to view this problem is to modify graph in such a way that the components of the resulting graph have smaller diameter. This is called as s-Club Cluster Vertex Deletion problem where the input is a graph with two integers $s \geq 2$ and $k \geq 1$ and the goal is to decide whether it is possible to remove at most k vertices from G such that each connected component of the resulting graph has diameter at most s.

We note that Madathil and Meeks [6] recently explored a generalization of this problem, termed Balanced Cluster Completion, in which, given a graph G, and two integers $0 \leq \eta \leq n$ and k, we are permitted to "edit" up to k edges. This editing should result in a cluster graph where the size difference between any two connected components of the resulting cluster graph does not exceed η. One can see that when $\eta = 0$, this problem reduces to the Uniform Cluster Editing problem. The authors presented polynomial kernels for Balanced Cluster Completion, Balanced Cluster Deletion, and Balanced Cluster Addition. While they provided polynomial kernels for the general cases, we have achieved better bounds in all considered variants when $\eta = 0$. We obtained our results simultaneously with and independent from those by Madathil and Meeks [6].

2 Kernelization algorithm for 2-EVD parameterized by solution size

In this section, we prove the following theorem.

Theorem 1. 2-EVD *parameterized by solution size admits a kernel of size* $\mathcal{O}(k^3)$.

A cluster graph is a graph where every connected component is a clique. Observe that a graph is a cluster graph if and only if it does not have an induced P_3. This observation serves as a cornerstone for our kernelization technique. The first step of the kernel is to compute a maximal set $\mathcal{P}_3$ of vertex-disjoint induced P_3s in G. This can be done by a greedy algorithm:

Algorithm 1

Require: A graph $G = (V, E)$.
Ensure: A maximal set $\mathcal{P}_3$ of vertex-disjoint induced P_3 in G.

```
P_3 = ∅
while G has an induced P_3 do
    Identify an induced P_3 = (u, v, w) in G
    P_3 = P_3 ∪ {P_3}
    G = G − {u, v, w}
end while
return S
```

At the end, if $|\mathcal{P}_3| > k$, we have a no-instance. So we assume that $|\mathcal{P}_3| \leq k$, and let S be the vertices of $\mathcal{P}_3$. We have $|S| \leq 3k$. Let us denote the set of cliques of $G - S$ by $\mathcal{C}$. We have the following simple rules:

Reduction EVD 1 *If there exists $s \in S$ with neighbours in more than $k+1$ cliques of $\mathcal{C}$, then delete s (and its incident edges from G) and decrement the parameter k by 1. The new instance is $(G - s, k - 1)$*

Lemma 2. *Reduction Rule EVD 1 is safe.*

Proof. Let X be an uniform cluster vertex deletion set of G of size at most k. For the sake of contradiction, let us assume that $s \in S$ is not in X. Therefore s must be in a clique of $G - X$. That clique can have vertices from at most one clique, say, C_i in $\mathcal{C}$. Since s has neighbours in more than or equal to $k+2$ cliques, we would need to include neighbours of s in at least $k+1$ cliques except C_i in the solution. This contradicts that the solution size X is at most k. Hence if (G, k) is a yes-instance, then s must be in X. □

Next, we partition the set of cliques of $\mathcal{C}$ based on the condition of whether they have neighbours in S or not. We define

$$\mathcal{C}_0 = \{C \in \mathcal{C} \ : \ \text{no vertex in } C \text{ has a neighbour in } S\}$$

and

$$\mathcal{C}_1 = \{C \in \mathcal{C} : \text{some vertex in } C \text{ has a neighbour in } S\}.$$

Observe that exhaustive application of reduction rule EVD 1 completely removes the vertices of S which are having neighbours in more than $k+1$ cliques of $\mathcal{C}_1$. In other words, after exhaustive application of reduction rule EVD 1, each vertex of S can have neighbours in at most $k+1$ cliques of $\mathcal{C}_1$. As $|S| \leq 3k$ and each vertex of S can have neighbours in at most $k+1$ cliques, there are at most $3k(k+1)$ cliques in $\mathcal{C}_1$. Therefore we have

$$|\mathcal{C}_1| \leq 3k(k+1).$$

A r-clique is a clique of size r. Now we have the following simple rule.

Reduction EVD 2 *If for some $r \in \{1, 2, \ldots, n\}$ there are more than $k+1$ r-cliques in $\mathcal{C}_0$, then remove all but $k+1$ r-cliques from G.*

Lemma 3. *Reduction rule EVD 2 is safe.*

Proof. Suppose (G, k) is a yes-instance of 2-EVD and X is an uniform cluster vertex deletion set of G of size at most k. If for some $r \in \{1, 2, \ldots, n\}$ there are more than $k+1$ r-cliques in $\mathcal{C}_0$, then we claim that r is the size of all cliques in $G - X$. For the sake of contradiction, let us assume that $r' < r$ is the size of all cliques in $G - X$. In that case, we need to delete at least one vertex from each r-clique. That means we have to add at least $k+1$ vertices in X, which contradicts that X is of size at most k. So, we conclude that r is the size of all cliques in $G - X$. As the solution size is at most k, this allows us to remove all but $k+1$ r-cliques from G. □

To bound the number of cliques in $\mathcal{C}_0$, we make the following observation.

Observation 1 *$\mathcal{C}_0$ contains cliques of at most $k+1$ distinct sizes.*

The above observation is correct because by deleting at most k vertices we can edit the sizes of at most k cliques. Due to this observation $\mathcal{C}_0$ can contain cliques of at most $k+1$ distinct sizes and by Reduction Rule EVD 2 there can be at most $k+1$ cliques of the same size. Therefore, we have

$$|\mathcal{C}_0| \leq (k+1)(k+1).$$

Therefore, we get

$$|\mathcal{C}| = |\mathcal{C}_0| + |\mathcal{C}_1| \leq (k+1)(k+1) + 3k(k+1) = 4k^2 + 5k + 1.$$

Now, we make two cases based on the maximum size of a clique in $\mathcal{C}$. Let $\omega(\mathcal{C}) = \max_{C \in \mathcal{C}} \{|C|\}$.

Case 1. We assume that $\omega(\mathcal{C}) < 8k$. In this case, we get a kernel of size $|S| + \sum_{C \in \mathcal{C}} |C| \leq 3k + 8k(4k^2 + 5k + 1) = \mathcal{O}(k^3)$.

Case 2. We assume that $\omega(\mathcal{C}) \geq 8k$. Let C_0 be a clique in $\mathcal{C}$ such that $\omega(\mathcal{C}) = |C_0|$. Then C_0 becomes a clique of size at least $|C_0| - k \geq 7k$ in $G - X$, where X is a solution of size at most k. In this case, the resulting graph $G - X$ must be a collection of r-cliques, where $r \geq |C_0| - k \geq 7k$.

Reduction EVD 3 *If there exists $C \in \mathcal{C}$ such that $|C| < 4k$, then remove C from G and decrement the parameter k by $|C|$. The new instance is $(G - C, k - |C|)$.*

Lemma 4. *Reduction rule EVD 3 is safe.*

Proof. For the sake of contradiction, let us assume that $v \in C$ does not belong to some solution X of size at most k. First, since C is a clique and $|C| \leq 4k - 1$, v has at most $4k - 2$ neighbours in C. Second, v can have at most $3k$ neighbours in S. Thus v can be part of a clique of size at most $(4k - 2) + 3k + 1 = 7k - 1$. This contradicts the fact that $G - X$ is a collection of r-cliques, where $r \geq 7k$. Therefore we must include all vertices of C in every uniform cluster vertex deletion set X of G of size at most k. □

We now introduce the concept of a *heavy neighbour* of a vertex $s \in S$ in $\mathcal{C}$, which is a key element for explaining the forthcoming reduction rules.

Definition 1. *A clique $C \in \mathcal{C}$ is called a heavy neighbour of $s \in S$ if s is adjacent to at least* $\max\{|C| - 4k, k + 1\}$ *vertices in C.*

We observe that if $s \in S$ has no heavy neighbour, then this vertex should belong to every solution of size at most k. Our next reduction rule is the following:

Reduction EVD 4 *If $s \in S$ has no heavy neighbour, remove s from G, and decrement the parameter k by 1. The new instance is $(G - s, k - 1)$.*

Lemma 5. *Reduction rule EVD 4 is safe.*

Proof. For the sake of contradiction, let us assume that $s \in S$ has no heavy neighbour and it does not belong to some solution X of size at most k. It means s must belong to some clique in $G - X$. As s has no heavy neighbour in $\mathcal{C}$, s is adjacent to at most $\max\{|C| - 4k, k + 1\} - 1$ vertices in some $C \in \mathcal{C}$. On the other hand, s can have at most $3k - 1$ neighbours in S. Suppose $\max\{|C| - 4k, k + 1\} - 1 = |C| - 4k - 1$. Then s can be part of a clique of size at most $(|C| - 4k - 1) + (3k - 1) + 1 = |C| - k - 1$. This contradicts the fact that $G - X$ is a collection of r-cliques, where $r \geq |C| - k$. Suppose $\max\{|C| - 4k, k + 1\} - 1 = k$. Then s can be part of a clique of size at most $k + (3k - 1) + 1 = 4k$. This contradicts the fact that $G - X$ is a collection of r-cliques, where $r \geq 7k$. Therefore, s must be part of every solution of size at most k. □

We observe that if $s \in S$ has at least $k + 1$ neighbours each in two cliques, then this vertex should belong to every solution of size at most k. Our next reduction rule is the following:

Reduction EVD 5 *If $s \in S$ has at least $k+1$ neighbours each in two cliques, remove s from G, and decrement the parameter k by 1. The new instance is $(G-s, k-1)$.*

Lemma 6. *Reduction rule 5 is safe.*

Proof. Suppose s is adjacent to at least $k+1$ vertices in C and C' each. For the sake of contradiction, let us assume that $s \in S$ does not belong to some solution X of size at most k. It means s must belong to a clique in $G - X$. This clique can have vertices either from C or C' but not both. Therefore, we must include at least $k+1$ neighbours of s from C or C' in X. This is a contradiction as X is of size at most k. $\square$

Observation 2 *After applying reduction rules 4 and 5 exhaustively, each $s \in S$ has exactly one heavy neighbour in $\mathcal{C}$.*

Suppose C is the heavy neighbour of $s_1^C, \ldots, s_r^C \in S$. For each $C \in \mathcal{C}$, we define

$$\mathcal{N}(C) = \bigcup_{i=1}^{r} \Big(C \setminus N(s_i^C)\Big) \cup \bigcup_{s \in S \setminus \{s_1^C, \ldots, s_r^C\}} \Big(N(s) \cap C\Big).$$

We also define

$$\mathcal{N}(\mathcal{C}) = \bigcup_{C \in \mathcal{C}} \mathcal{N}(C)$$

Lemma 7. *We have $|\mathcal{N}(\mathcal{C})| = \mathcal{O}(k^3)$.*

Proof. We have

$$\begin{aligned}
\mathcal{N}(\mathcal{C}) &= \bigcup_{C \in \mathcal{C}} \bigcup_{i=1}^{r} \Big(C \setminus N(s_i^C)\Big) \cup \bigcup_{C \in \mathcal{C}} \bigcup_{s \in S \setminus \{s_1^C \ldots, s_r^C\}} \Big(N(s) \cap C\Big) \\
&= \bigcup_{i=1}^{r} \bigcup_{C \in \mathcal{C}} \Big(C \setminus N(s_i^C)\Big) \cup \bigcup_{s \in S \setminus \{s_1^C \ldots, s_r^C\}} \bigcup_{C \in \mathcal{C}} \Big(N(s) \cap C\Big)
\end{aligned}$$

Note that by Reduction Rule EVD 1 each $s \in S$ has neighbours in at most $k+1$ cliques in $\mathcal{C}$ and by Reduction Rule 5, for each $s \in S$ can have at most $k+1$ neighbours in a clique. Therefore, we have

$$\left| \bigcup_{C \in \mathcal{C}} \Big(N(s) \cap C\Big) \right| < (k+1)(k+1).$$

Also, for each $s_i^C \in S$, we have

$$\begin{aligned}
\left|C \setminus N(s_i^C)\right| &\leq |C| - (|C| - 4k) = 4k \text{ if } \max\{|C| - 4k, k+1\} = |C| - 4k \\
&\leq |C| - (k+1) \leq 4k \text{ if } \max\{|C| - 4k, k+1\} = k+1
\end{aligned}$$

Recall that C is the heavy neighbour of s_i^C. As there are at most $3k$ vertices in S, we get that $|\mathcal{N}(\mathcal{C})| \leq 3k[(k+1)(k+1) + 4k] = 3k^3 + 9k^2 + 3k = \mathcal{O}(k^3)$. $\square$

Lemma 8. *For each $C \in \mathcal{C}$, every pair of adjacent vertices x and y in $C \setminus \mathcal{N}(C)$ are true twins, that is, $N[x] = N[y]$.*

Proof. By the definition of $\mathcal{N}(C)$, it is clear that for a vertex $x \in C \setminus \mathcal{N}(C)$, we have $N(x) \cap S = \{s_1^C, \ldots, s_r^C\}$. Recall that C is the heavy neighbour of $s_1^C \cdots s_r^C$. Since $v \in C$ and C is a clique, v is adjacent to all other vertices of C. Therefore, for every $v \in C \setminus \mathcal{N}(C)$, we have $N[v] = C \cup \{s_1^C, \ldots, s_r^C\}$. □

Reduction EVD 6 *If* $\min_{C \in \mathcal{C}}\{|C \setminus \mathcal{N}(C)|\} > k+1$, *then arbitrarily delete exactly one vertex from each $C \setminus \mathcal{N}(C)$, $C \in \mathcal{C}$. The new instance is $(G - W, k)$ where W is the set of vertices deleted from G.*

Lemma 9. *Reduction Rule EVD 6 is safe.*

A proof of Lemma 6 can be found in Appendix B. After exhaustive application of Reduction Rule 6, we know that there exists a clique, say $C_q \in \mathcal{C}$ such that $|C_q - \mathcal{N}(C_q)| \leq k+1$. Note that $\max_{C \in \mathcal{C}} |C| - |C_q| \leq 4k$ due to Reduction Rule EVD 3. Therefore, we have

$$
\begin{aligned}
|V(G)| &\leq |S| + |\mathcal{N}(\mathcal{C})| + \left| \bigcup_{C \in \mathcal{C}} (C - \mathcal{N}(C)) \right| \\
&\leq (3k) + (3k^3 + 9k^2 + 3k) + (5k+1)(4k^2 + 5k + 1) \\
&= 23k^3 + 38k^2 + 16k + 1 = \mathcal{O}(k^3)
\end{aligned}
$$

3 Kernelization algorithm for 2-EEE parameterized by solution size

In this section, we study the following problem: For a given graph G, can we transform G into an uniform cluster graph by "editing" at most k adjacencies, that is, by adding or deleting at most k edges? More formally, let $G = (V, E)$ be a graph. Then $F \subseteq V \times V$ is called an *uniform cluster editing set* for G if $G \triangle F$ is a uniform cluster graph. In this section, we prove the following theorem.

Theorem 2. 2-EEE *parameterized by solution size admits a kernel of size $\mathcal{O}(k^2)$.*

Note that many preprocessing rules applied in the case of 2-EEE will be applicable in the cases of 2-EEA and 2-EED as well. To make the writing concise, we will refer the following pre processing step as *preparation* step. The *preparation* step begins here. A graph is a cluster graph if and only if it does not have an induced P_3. It is easy to compute a maximal set $\mathcal{P}_3$ of vertex-disjoint induced P_3s in G using Algorithm 1. If $|\mathcal{P}_3| > k$, we have a no-instance. So we assume that $|\mathcal{P}_3| \leq k$, and let S be the vertices of $\mathcal{P}_3$. We have $|S| \leq 3k$. Let us denote the set of cliques of $G - S$ by $\mathcal{C}$. Let F be an uniform cluster editing set of size at most k for G. Let V_F be the vertices of F. Then we have $|V_F| \leq 2k$.

Next, we observe that if there is indeed an uniform cluster vertex deletion set F of size at most k, then the vertices in $V \setminus V_F$ have equal degree in both G and $G \triangle F$. In other words, for every $x, y \in V \setminus V_F$, we have $d_G(x) = d_G(y) = d_{G \triangle F}(x) = d_{G \triangle F}(y)$. We assume that G has at least $4k{+}1$ vertices; otherwise, we have a kernel of size $4k$. We know that if the input is a yes-instance, then at least $2k + 1$ vertices have the same degree say d and at most $2k$ vertices have degree not equal to d. Therefore, we first check if the input instance (G, k) satisfies this condition. If the input instance does not satisfy this condition, then conclude that we are dealing with a no-instance. If the input instance satisfies this condition, we can calculate the exact value of d in polynomial time. Assuming that the input is a yes-instance, let c be the size of equal sized cliques in $G \triangle F$. One can observe that c must be equal to $d + 1$ in $G \triangle F$. This implies that the minimum degree of G is at least $d - k$ because by executing k edge-editing operations, the degree of a vertex increases at most by k. Similarly, the maximum degree of G is at most $d + k$ because by executing k editing operations, the degree of a vertex decreases at most by k. The preparation step ends here. The following two rules are safe.

Reduction EEE 1 *If $\delta(G) < d - k$ or $\Delta(G) > d + k$ then conclude that we are dealing with a no-instance.*

Reduction EEE 2 *If $|\{v \in V \mid d_G(v) \neq d\}| > 2k$ then conclude that we are dealing with a no-instance.*

We make two cases based on the value of d.

Case 1: Let us assume that $d \geq 6k$.

Lemma 10. *If (G, k) is a yes-instance then for each $C \in \mathcal{C}$, we have $|C| > 2k$.*

Proof. If there is a clique $C \in \mathcal{C}$ such that $|C| \leq 2k$ then every vertex $v \in C$ has degree at most $5k - 1$ because v has at most $2k - 1$ neighbours in C and at most $3k$ neighbours in S. As $d \geq 6k$, we have $5k - 1 < d - k$. This implies $\delta(G) < d - k$. Due to Reduction Rule 1, if $\delta(G) < d - k$ then conclude that we are dealing with a no-instance. Therefore, we must have $|C| > 2k$. □

Reduction EEE 3 *If $s \in S$ is adjacent to at least $k + 1$ vertices of a clique $C \in \mathcal{C}$, then add all missing edges between s and C and decrement the parameter k by $|W|$. Here, W is the set of all missing edges between s and C. The resulting instance is $(G + W, k - |W|)$. If $|W| \geq k + 1$, then conclude that we are dealing with a no-instance.*

Lemma 11. *Reduction rule 3 is safe.*

Proof. The vertices of C must belong to the same connected component in the edited graph. As s has at least $k + 1$ neighbours in C, s must be in the same connected component as the vertices in C. Otherwise, we need to delete at least $k{+}1$ edges between s and C to place s in a separate component which contradicts our assumption that $|F| \leq k$. Therefore, we should add all the missing edges between s and C. □

Reduction EEE 4 *If $s \in S$ is not adjacent to at least $k+1$ vertices of a clique $C \in \mathcal{C}$, then delete all edges between s and C, and decrement the parameter k by $|W|$. Here, W is the set of edges between s and C. The new resulting instance is $(G \triangle W, k - |W|)$. If $|W| \geq k+1$, then conclude that we are dealing with a no-instance.*

Lemma 12. *Reduction rule 4 is safe.*

Proof. All the vertices of C must belong to the same connected component in the edited graph. Note that s cannot be in the same connected component as the vertices in C, as it requires adding at least $k+1$ edges. Therefore, we must delete all the edges between s and C. □

Observe that exhaustive application of reductions EEE 3 and EEE 4 ensure that for each $s \in S$ and each $C \in \mathcal{C}$, either s is adjacent to every vertex of $C \in \mathcal{C}$ or s is adjacent to no vertex of C.

Reduction EEE 5 *If $s \in S$ has neighbours in two distinct cliques of $\mathcal{C}$, then conclude that we are dealing with a no-instance.*

Lemma 13. *Reduction rule 5 is safe.*

Proof. Suppose s has neighbours in cliques C and C'. Then due to Reduction rules 3 and 4, s must be adjacent to every vertex of C and C'. This implies that the graph contains at least $2k+1$ edge disjoint induced P_3s. Therefore the input instance is a no-instance. □

Reduction EEE 6 *If s_1 and s_2 are two non-adjacent vertices of S and have neighbour in $C \in \mathcal{C}$, then add the edge (s_1, s_2) and decrement the parameter k by 1. The new instance is $(G + (s_1, s_2), k-1)$.*

Lemma 14. *Reduction rule 6 is safe.*

Proof. Due to reductions EEE 3 and EEE 4, s_1 and s_2 are adjacent to every vertex of C. Thus, s_1 and s_2 must be in the same connected component as the vertices in C in the edited graph. Therefore, we must add the edge (s_1, s_2) to the solution. □

Reduction EEE 7 *If s_1 and s_2 are two adjacent vertices of S and have neighbour in $C_1 \in \mathcal{C}$ and $C_2 \in \mathcal{C}$ respectively, then delete the edge (s_1, s_2) and decrement the parameter k by 1. The new instance is $(G - (s_1, s_2), k-1)$.*

Lemma 15. *Reduction rule 7 is safe.*

Proof. The vertices of C_1 and vertices of C_2 must be in the different connected components in the edited graph. Therefore, s_1 and s_2 as well must be in the different connected components in the edited graph as well. Therefore, we must delete the edge (s_1, s_2) and add it it to solution. □

Lemma 16. *If (G,k) is a yes-instance and none of the reduction rules EEE 1 to EEE 4, is applicable to G then G is disjoint union of cliques of size $d+1$.*

Proof. When none of the reduction rules EEE 1 to EEE 7, is applicable to G then clearly G is disjoint union of cliques. Now we show that the cliques are of size $d+1$. If there exists a clique of size more than $d+1$ then there exists more than $2k$ vertices whose degree is not equal to d. Similarly, If there exists a clique of size less than $d+1$ then there exists more than $2k$ vertices whose degree is not equal to d because as shown in Lemma 10 all the cliques in $C \in \mathcal{C}$ have size at least $2k+1$. This is not possible due to Reduction Rule 2. □

In Case 1, we get a kernel of size $\mathcal{O}(1)$.

Case 2: Let us assume that $d \leq 6k-1$.

We partition $\mathcal{C} = G - S$ into four parts as follows: $\mathcal{C}_{<} = \{C \in \mathcal{C} : |C| < d+1\}$, $\mathcal{C}_{0,d+1} = \{C \in \mathcal{C} : |C| = d+1$ and no vertex in C has a neighbour in $S\}$, $\mathcal{C}_{1,d+1} = \{C \in \mathcal{C} : |C| = d+1$ and some vertex in C has a neighbour in $S\}$, and $\mathcal{C}_{>} = \{C \in \mathcal{C} : |C| > d+1\}$.

Lemma 17. *If (G,k) is a yes-instance and reduction rule EEE 2 is not applicable to G then $\bigcup\limits_{C \in \mathcal{C}_{>}} |V(C)| \leq 2k$.*

Proof. Note that every vertex in $\bigcup\limits_{C \in \mathcal{C}_{>}} V(C)$ has degree more than $d+1$. Due to Reduction rule 2, the number of vertices with degree more than $d+1$ is bounded by $2k$.

Reduction EEE 8 *If $|\mathcal{C}_{0,d+1}| > 2k$, then remove all but $2k+1$ cliques from $\mathcal{C}_{0,d+1}$.*

Lemma 18. *Reduction Rule 8 is safe.*

Proof. An edge editing set of size at most k can change degrees of at most $2k$ vertices in $\mathcal{C}_{0,d+1}$. In other words, an edge editing set of size at most k can effect at most $2k$ cliques in $\mathcal{C}_{0,d+1}$. There will still exist at least one clique of size $d+1$ in the final graph. Therefore, we can delete all but $2k+1$ cliques. □

Lemma 19. $|\mathcal{C}_{1,d+1}| \leq 2k$.

Proof. Each clique $C \in \mathcal{C}_{1,d+1}$ by definition does contain a vertex which has a neighbour in S. This implies every clique in $\mathcal{C}_{1,d+1}$ contains at least one vertex whose degree is more than d. Due to Reduction Rule 2, there are at most $2k$ such vertices. This implies that $|\mathcal{C}_{1,d+1}| \leq 2k$. □

Therefore, we get $\left|\bigcup\limits_{C \in \mathcal{C}_{d+1}} V(C)\right| \leq (4k+1)(6k-1) = 24k^2 + 2k - 1$, where $\mathcal{C}_{d+1} = \mathcal{C}_{0,d+1} \cup \mathcal{C}_{1,d+1}$.

Lemma 20. *If (G,k) is a yes-instance and reduction rule EEE 1 is not applicable to G then we have $\bigcup_{C\in\mathcal{C}_<} |V(C)| < 21k^2+5k$.*

Proof. Since Reduction rule 1 is not applicable to G, every vertex in G has degree at most $d+k \leq 7k-1$. For the sake of contradiction assume that $\bigcup_{C\in\mathcal{C}_<} |V(C)| \geq 21k^2+5k$. Each clique $C \in \mathcal{C}_<$ is of size $< d+1$, hence the vertices in C have degree $< d$. As (G,k) is a yes-instance, G has an uniform cluster vertex editing set of size at most k that can increase the degree of at most $2k$ vertices. Therefore at least $21k^2+5k-2k$ vertices in $\bigcup_{C\in\mathcal{C}_<} V(C)$ have neighbours in S. As $|S| \leq 3k$ and S has at least $21k^2+53k$ neighbours in $G-S$, by Pigeonhole principle there is a vertex $s \in S$ with at least $7k+1$ neighbours in $G-S$, a contradiction to the fact that $d_G(v) \leq 7k-1$ for all $v \in V$. Therefore, we have $\bigcup_{C\in\mathcal{C}_<} |V(C)| < 21k^2+5k$. ☐

Applying the above reduction rules, we get that $|V(G)| \leq |S| + |\mathcal{C}_<| + |\mathcal{C}_{d+1}| + |\mathcal{C}_>| \leq 3k + (21k^2+5k) + (24k^2+2k-1) + (2k) = 45k^2+12k-1$.

4 Conclusion and Open Problems

In this paper, we stuided the problem of modifying a given graph such that the resulting graph becomes a collection of equal-sized cliques. We gave polynomial kernels for different types of modification problems. It would be interesting to see if one could improve the kernelization algorithms with better bounds.

References

1. M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.
2. M. Doob. On characterizing certain graphs with four eigenvalues by their spectra. *Linear Algebra and its Applications*, 3:461–482, 1970.
3. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 2012.
4. F. Goldberg, S. Kirkland, A. Varghese, and A. Vijayakumar. On split graphs with four distinct eigenvalues. *Discrete Appl. Math.*, 277(C):163–171, apr 2020.
5. R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.
6. J. Madathil and K. Meeks. Parameterized algorithms for balanced cluster edge modification problems, 2024.
7. S. Meesum, P. Misra, and S. Saurabh. Reducing rank of the adjacency matrix by graph modification. *Theoretical Computer Science*, 9198, 02 2016.
8. N. Misra, H. Mittal, S. Saurabh, and D. Thakkar. On the Complexity of the Eigenvalue Deletion Problem. In S. Iwata and N. Kakimura, editors, *34th International Symposium on Algorithms and Computation (ISAAC 2023)*, volume 283 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:17, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

Appendix A

5 Parameterized Complexity

A *parameterized problem* is a language $L \subseteq \Sigma^\star \times \mathbb{N}$, where Σ is a fixed, finite alphabet. For an instance $(x, k) \in \Sigma^\star \times \mathbb{N}$, k is called the *parameter*. A parameterized problem $\mathcal{P}$ is *fixed-parameter tractable* (FPT in short) if a given instance (x, k) can be solved in time $f(k) \cdot |(x, k)|^c$ where f is some (usually computable) function, and c is a constant. Parameterized complexity classes are defined with respect to *fpt-reducibility*. A parameterized problem $\mathcal{P}$ is *fpt-reducible* to $\mathcal{Q}$ if in time $f(k) \cdot |(x, k)|^c$, one can transform an instance (x, k) of $\mathcal{P}$ into an instance (x', k') of Q such that $(x, k) \in \mathcal{P}$ if and only if $(x', k') \in \mathcal{Q}$, and $k' \leq g(k)$, where f and g are computable functions depending only on k. Owing to the definition, if $\mathcal{P}$ *fpt-reduces* to $\mathcal{Q}$ and $\mathcal{Q}$ is fixed-parameter tractable then $\mathcal{P}$ is fixed-parameter tractable as well.

What makes the theory more interesting is a hierarchy of intractable parameterized problem classes above FPT which helps in distinguishing those problems that are not fixed parameter tractable. Central to parameterized complexity is the following hierarchy of complexity classes, defined by the closure of canonical problems under *fpt-reductions*: FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq \ldots \subseteq$ XP. All inclusions are believed to be strict. In particular, FPT $\neq$ W[1] under the Exponential Time Hypothesis [5]. The class W[1] is the analog of NP in parameterized complexity. A major goal in parameterized complexity is to distinguish between parameterized problems which are in FPT and those which are *W[1]-hard*, i.e., those to which every problem in W[1] is *fpt-reducible*. There are many problems shown to be complete for W[1], or equivalently *W[1]-complete*, including the Multi-Colored Clique (MCC) problem [3].

Closely related to fixed-parameter tractability is the notion of preprocessing. A *reduction to a problem kernel*, or equivalently, *problem kernelization* means to apply a data reduction process in polynomial time to an instance (x, k) such that for the reduced instance (x', k') it holds that (x', k') is equivalent to (x, k), $|x'| \leq g(k)$ and $k' \leq g(k)$ for some computable function g only depending on k. Such a reduced instance is called a *problem kernel*. It is easy to show that a parameterized problem is in FPT if and only if there is kernelization algorithm. A *polynomial kernel* is a kernel, whose size can be bounded by a polynomial in the parameter. We refer to [1,3] for further details on parameterized complexity.

Appendix B

6 Proof of Lemma 9

Proof. Let $I' = (G', k)$ be an instance of 2-EVD obtained from $I = (G, k)$ by exhaustively applying Reduction Rule EVD 6. We will show that I is a yes-instance if and only if I' is a yes-instance. Let $X \subseteq V(G)$ be a solution of size at most k for the instance I. We will construct a solution X' of the same size as X for the instance I'. We know $G - X$ is a collection of equal sized cliques $Q_1, Q_2, \ldots, Q_r$. We observe that there is no clique Q_i such that $Q_i \subseteq S$. This is true because otherwise $|Q_i| \leq 3k$ as $|S| \leq 3k$, and we have already observed that the size of cliques must be at least $7k$. We construct X' from X as follows: if $u \in X \cap (C \setminus \mathcal{N}(C))$ for some $C \in \mathcal{C}$ and u is deleted during the execution of Reduction Rule 6, then remove u from X and include u's true twin $v \notin X$.

We have to show that $G' - X'$ is a collection of equal sized cliques. It is easy to see that $G' - X'$ is a collection of cliques $Q'_i = Q_i \setminus u$ where $u \in Q_i$ is a vertex deleted by Reduction Rule 6. As $|Q_i| = |Q_j|$, we get $|Q'_i| = |Q'_j|$ for all $i \neq j$. This shows that I' is a yes-instance.

In the other direction, let us assume that X' is a solution of size at most k for the instance I'. Let us denote the equal sized cliques in $G' - X'$ by $Q'_1, Q'_2, \ldots, Q'_r$. We claim that $X := X'$ is a solution for the instance I. Note that $G - X$ is again a collection of cliques $Q_1, Q_2, \ldots, Q_r$ where $Q_i = Q'_i \cup \{u\}$ where u is a vertex deleted by Reduction Rule 6. Clearly, we have $|Q_i| = |Q_j|$ as we know that $|Q'_i| = |Q'_j|$ for all $i \neq j$. □

Appendix C

7 Kernelization algorithm for 2-EEA parameterized by solution size

In this subsection, we provide a proof of the following theorem.

Theorem 3. 2-EEA *admits a kernel with at most* $6k$ *vertices.*

We begin with the same *preparation step* designed for 2-EEE.

Reduction EEA 1 *If* $\delta(G) < d - k$ *or* $\Delta(G) > d$ *then conclude that we are dealing with a no-instance.*

Reduction EEA 2 *If we have* $|\{v \mid d_G(v) \neq d\}| > 2k$ *then conclude that we are dealing with a no-instance.*

Reduction EEA 3 *If there is a path between two nonadjacent vertices* u *and* v *then make* u *adjacent to* v*, and decrease* k *by 1. The new instance is* $(G + (u, v), k - 1)$.

Lemma 21. *Reduction rule 3 is safe.*

Proof. As there is a path between u and v, they must be in the same clique in the edited graph. Therefore, we must join u and v by an edge. □

Observe that exhaustive application of Reduction rule EEA 3 ensures that the new graph is a collection of disjoint cliques.

Reduction EEA 4 *If there are more than one clique of size* $d+1$*, then we keep only* x *cliques of size* $d+1$ *where* $x(d+1) \geq 2k+1$ *and remove the remaining cliques of size* $d+1$.

Lemma 22. *Reduction Rule 4 is safe.*

Proof. Note that the Reduction Rule EEA 4 ensures that at least $2k+1$ vertices have degree exactly d which in turn ensures that the size of cliques after adding at most k remains exactly $d+1$. Therefore deleting the extra cliques does not change the size of a solution. □

Case 1: Let us assume that $d \geq k+1$.

Reduction EEA 5 *If there is a clique of size less than* $d+1$*, then conclude that we are dealing with a no-instance.*

Proof. Due to reduction rule EEA 1, there cannot be a clique of size more than $d+1$. Let us assume that we have a clique C of size $x < d+1$. Then to get a $d+1$ sized clique from C, we will have to add at least $x(d+1-x)$ edges, which is not possible as $x(d+1-x) > k$. □

In the case 1, we get a kernel of size $\mathcal{O}(1)$.

Case 2: Let us assume that $d \leq k$.
After applying reduction rule EEA 4, the cliques of size $d+1$ contain at most $4k$ vertices. By reduction rule 2 there are at most $2k$ vertices with degree not equal to d. Therefore, we have $V(G) \leq 6k$.

This completes the proof of Theorem 3.

Appendix D

8 Kernelization algorithm for 2-EED parameterized by solution size

In this subsection, we provide a proof of the following theorem.

Theorem 4. 2-EED *parameterized by solution size admits a kernel of size at most* $\mathcal{O}(k^2)$.

We begin with the same *preparation step* designed for 2-EEE.

Reduction EED 1 *If* $\delta(G) < d$ *or* $\Delta(G) > d+k$ *then conclude that we are dealing with a no-instance.*

Reduction EED 2 *If we have* $|\{v \mid d_G(v) \neq d\}| > 2k$ *then conclude that we are dealing with a no-instance.*

We make two cases based on the values of d.

Case 1: Let us assume that $d \geq 5k$.

Lemma 23. *For each $C \in \mathcal{C}$, we have $|C| > 2k$.*

Proof. If there is a clique $C \in \mathcal{C}$ such that $|C| \leq 2k$ then every vertex $v \in C$ has degree at most $5k - 1$ because v has $2k - 1$ neighbours in C and at most $3k$ neighbours in S. As $d \geq 5k$, we have $5k - 1 < d$. This implies $\delta(G) < d$. Due to Reduction Rule EED 1, if $\delta(G) < d$ then conclude that we are dealing with a no-instance. Therefore, we must have $|C| > 2k$. □

Reduction EED 3 *If $s \in S$ is not adjacent to at least one vertex of $C \in C_S$, then delete all edges between s and C. The new instance is $(G - W, k - |W|)$, where W is the set of edges between s and C. In the case $|W| \geq k + 1$, conclude that we are dealing with a no-instance.*

Lemma 24. *Reduction rule 3 is safe.*

Proof. All of the vertices of C must belong to the same connected component in the edited graph. As s cannot be in the same connected component as the vertices in C, we must delete all the edges between s and C. □

Reduction EED 4 *If $s \in S$ has neighbours in two distinct cliques of $\mathcal{C}$, then conclude that we are dealing with a no-instance.*

Lemma 25. *Reduction rule 4 is safe.*

Proof. Suppose $s \in S$ has neighbours in cliques C and C'. Then due to Reduction rules 3, s must be adjacent to all vertices of C and C'. This implies that the graph contains at least $2k + 1$ edge disjoint induced P_3's. Therefore the input instance is a no-instance. □

Reduction EED 5 *If u and v are two nonadjacent vertices of S such that both have neighbours in $C \in \mathcal{C}$, then conclude that we are dealing with a no-instance.*

Lemma 26. *Reduction rule 6 is safe.*

Proof. Note that vertices u and v must be in the same connected component as the vertices in C in the edited graph. Therefore, since the edge (u, v) is not there in the graph, we must return a no-instance. □

Lemma 27. *After applying all the reduction rules exhaustively, the graph becomes disjoint union of cliques.*

Proof. Note that $u \in S$ and $v \in S$ are in the same connected component if and only if they have neighbours in the same clique $C \in \mathcal{C}$. Due to Reduction rule 5, we have shown that the edge (u, v) must in the graph. Also due to Reduction rule 3, u and v are adjacent to all the vertices in C. This shows that all the connected components are cliques. □

Lemma 28. *After applying Reduction Rule 2 exhaustively, we must be left with cliques of size exactly equal to $d + 1$.*

Proof. Note that if there exists a clique of size more than $d+1$ or less than $d+1$, then there exists more than $2k$ vertices whose degree is not equal to d. This is not possible due to Reduction Rule 2. □

In Case 1, thus we get a kernel of size $\mathcal{O}(1)$.

Case 2: Let us assume that $d \leq 5k-1$.
We partition $\mathcal{C} = G - S$ into four parts as follows: $\mathcal{C}_< = \{C \in \mathcal{C} : |C| < d+1\}$, $\mathcal{C}_{0,d+1} = \{C \in \mathcal{C} : |C| = d+1$ and no vertex in C has a neighbour in $S\}$, $\mathcal{C}_{1,d+1} = \{C \in \mathcal{C} : |C| = d+1$ and some vertex in C has a neighbour in $S\}$, and $\mathcal{C}_> = \{C \in \mathcal{C} : |C| > d+1\}$.

Lemma 29. *After applying Reduction Rule 2, we have* $\bigcup_{C \in \mathcal{C}_>} |V(C)| \leq 2k$.

Proof. Note that every vertex in $\bigcup_{C \in \mathcal{C}_>} V(C)$ has degree more than $d+1$. Due to Reduction Rule 2, the number of such vertices must be bounded by $2k$. □

Reduction EED 6 *If* $|\mathcal{C}_{0,d+1}| > k+1$*, then remove all but* $k+1$ *cliques from* $\mathcal{C}_{0,d+1}$.

Lemma 30. *Reduction Rule 6 is safe.*

Proof. An edge deleting set of size at most k can change degrees of at most $2k$ vertices in $\mathcal{C}_{0,d+1}$. In other words, an edge deleting set of size at most k can effect at most k cliques in $\mathcal{C}_{0,d+1}$. There will still exist at least one clique of size $d+1$ in the final graph. Therefore, we can delete all but $k+1$ cliques. □

Lemma 31. $|\mathcal{C}_{1,d+1}| \leq 2k$.

Proof. Each clique $C \in \mathcal{C}_{1,d+1}$ by definition does contain a vertex which has a neighbour in S. This implies every clique in $\mathcal{C}_{1,d+1}$ contains at least one vertex whose degree is more than d. Due to Reduction Rule 2, there are at most $2k$ such vertices. This implies that $|\mathcal{C}_{1,d+1}| \leq 2k$. □

Therefore, we get $\left| \bigcup_{C \in \mathcal{C}_{d+1}} V(C) \right| \leq (3k+1)(5k-1) = 15k^2 + 2k - 1$, where $\mathcal{C}_{d+1} = \mathcal{C}_{0,d+1} \cup \mathcal{C}_{1,d+1}$.

Lemma 32. *After applying Reduction Rule 1 exhaustively, we have* $\mathcal{C}_< = \emptyset$.

Applying the above reduction rules, we get that $|V(G)| \leq |S| + |\mathcal{C}_<| + |\mathcal{C}_{d+1}| + |\mathcal{C}_>| \leq 3k + (15k^2 + 2k - 1) + (2k) = 15k^2 + 7k - 1$.

Theorem 5. *2-EED parameterized by solution size admits a kernel of size* $\mathcal{O}(k^2)$.