

Cluster Editing: Kernelization based on Edge Cuts ^{*}

YIXIN CAO

JIANER CHEN

Department of Computer Science and Engineering
Texas A&M University
{yixin, chen}@cse.tamu.edu

October 29, 2018

Abstract

Kernelization algorithms for the CLUSTER EDITING problem have been a popular topic in the recent research in parameterized computation. Thus far most kernelization algorithms for this problem are based on the concept of *critical cliques*. In this paper, we present new observations and new techniques for the study of kernelization algorithms for the CLUSTER EDITING problem. Our techniques are based on the study of the relationship between CLUSTER EDITING and graph edge-cuts. As an application, we present an $\mathcal{O}(n^2)$ -time algorithm that constructs a $2k$ kernel for the *weighted* version of the CLUSTER EDITING problem. Our result meets the best kernel size for the unweighted version for the CLUSTER EDITING problem, and significantly improves the previous best kernel of quadratic size for the weighted version of the problem.

1 Introduction

Errors are ubiquitous in most experiments, and we have to find out the true information buried behind them, that is, to remove the inconsistencies in data of experiment results. In most cases, we want to make the data consistent with the least amount of modifications, i.e., we assume the errors are not too much. This is an *everyday* problem in real life. Indeed, the problem has been studied by researchers in different areas [3, 25]. A graph theoretical formulation of the problem is called the CLUSTER EDITING problem that seeks a collection of edge insertion/deletion operations of minimum cost that transforms a given graph into a union of disjoint cliques. The CLUSTER EDITING problem has applications in many areas, including machine learning [3], world wide web [12], data-mining [4], information retrieval [19], and computational biology [10]. The problem is also closely related to another interesting and important problem in algorithmic research, CLUSTERING AGGREGATION [1], which, given a set of clusterings on the same set of vertices, asks for a single clustering that agrees as much as possible with the input clusterings.

Let $G = (V, E)$ be an undirected graph, and let V^2 be the set of all unordered pairs of vertices in G (thus, for two vertices v and w , $\{v, w\}$ and $\{w, v\}$ will be regarded as the same pair). Let $\pi : V^2 \mapsto \mathbb{N} \cup \{+\infty\}$ be a *weight function*, where \mathbb{N} is the set of positive integers. The *weight* of an edge $[v, w]$ in G is defined to be $\pi(v, w)$. If vertices v and w are not adjacent, and we add an edge between v and w , then we say that we *insert an edge* $[v, w]$ of *weight* $\pi(v, w)$.

The weighted CLUSTER EDITING problem is formally defined as follows:

(Weighted) CLUSTER EDITING: Given (G, π, k) , where $G = (V, E)$ is an undirected graph, $\pi : V^2 \mapsto \mathbb{N} \cup \{+\infty\}$ is a weight function, and k is an integer, is it possible to transform G into a union of disjoint cliques by edge deletions and/or edge insertions such that the weight sum of the inserted edges and deleted edges is bounded by k ?

^{*}Supported in part by the US National Science Foundation under the Grants CCF-0830455 and CCF-0917288. Work partially done when both authors were visiting Central South University, Changsha, China.

The problem is NP-complete even in its unweighted version [25]. Polynomial-time approximation algorithms for the problem have been studied. The best result is a randomized approximation algorithm of expected approximation ratio 3 by Ailon, Charikar, and Newman [1], which was later derandomized by van Zuylen and Williamson [27]. The problem has also been shown to be APX-hard by Charikar, Guruswami, and Wirth [8].

Recently, some researchers have turned their attention to exact solutions, and to the study of parameterized algorithms for the problem. A closely related problem is to study *kernelization algorithms* for the problem, which, on an instance (G, π, k) of CLUSTER EDITING, produces an “equivalent” instance (G', π, k') such that $k' \leq f(k)^1$ and that the *kernel size* (i.e., the number of vertices in the graph G') is small. For the unweighted version of the problem (i.e., assuming that for each pair v and w of vertices, $\pi(v, w) = 1$), Gramm et al. [17] presented the first parameterized algorithm running in time $\mathcal{O}(2.27^k + n^3)$ and a kernelization algorithm that produces a kernel of $\mathcal{O}(k^2)$ vertices. This result was immediately improved by a successive sequence of studies on kernelization algorithms that produce kernels of size $24k$ [15], of size $4k$ [18] and of size $2k$ [9]. The $24k$ kernel was obtained via *crown reduction*, while the later two results were both based on the concept of simple series module (*critical clique*), which is a restricted version of modular decomposition [11]. Basically, these algorithms iteratively construct the modular decomposition, find reducible simple series modules and apply reduction rules on them, until there are no any reducible modules found.

For the weighted version, to our best knowledge, the only non-trivial result on kernelization is the quadratic kernel developed by Böcker et al. [7].

The main result of this paper is the following theorem:

Theorem 1.1 *There is an $\mathcal{O}(n^2)$ -time kernelization algorithm for the weighted CLUSTER EDITING problem that produces a kernel which contains at most $2k$ vertices.*

Compared to all previous results, Theorem 1.1 is better not only in kernel size and running time, but also more importantly in conceptual simplicity.

A more general version of weighted CLUSTER EDITING problem is defined with real weights, that is, the weight function π is replaced by $\pi' : V^2 \mapsto \mathbb{R}_{\geq 1} \cup \{+\infty\}$ where $\mathbb{R}_{\geq 1}$ is the set of all real numbers larger than or equal to 1, and correspondingly k becomes a positive real number. Our result also works for this version, in the same running time, and with only a small relaxation in the constant of kernel size.

Our contribution. We report the first linear vertex kernel with very small constant, for the weighted version of the CLUSTER EDITING problem. Our contribution to this research includes:

1. the cutting lemmas (some of them are not used for our kernelization algorithm) are of potential use for future work on kernelizations and algorithms;
2. both the idea and the process are very simple with efficient implementations that run in time $\mathcal{O}(n^2)$. Indeed, we use only a single reduction rule, which works for both weighted and unweighted versions;
3. the reduction processes to obtain the above results are independent of k , and therefore are more general and applicable.

2 Cutting Lemmas

In this paper, graphs are always undirected and simple. A graph is a *complete graph* if each pair of vertices are connected by an edge. A *clique* in a graph G is a subgraph G' of G such that G' is a complete graph. By definition, a clique of h vertices contains $\binom{h}{2} = h(h-1)/2$ edges. If two vertices v and w are not adjacent, then we say that the edge $[v, w]$ is *missing*, and call the pair $\{v, w\}$ an *anti-edge*. The total number of anti-edges in a graph of n vertices is $n(n-1)/2 - |E(G)|$. The subgraph of the graph G induced by a vertex subset X is denoted by $G[X]$.

Let $G = (V, E)$ be a graph, and let $S \subseteq V^2$. Denote by $G \triangle S$ the graph obtained from G as follows: for each pair $\{v, w\}$ in S , if $[v, w]$ is an edge in G , then remove the edge $[v, w]$ in the graph, while if $\{v, w\}$ is an anti-edge, then insert the edge $[v, w]$ into the graph. A set $S \subseteq V^2$ is a *solution* to a graph $G = (V, E)$ if the graph $G \triangle S$ is a union of disjoint cliques.

¹ $f(\cdot)$ is a computable function.

For an instance (G, π, k) of CLUSTER EDITING, where $G = (V, E)$, the *weight* of a set $S \subseteq V^2$ is defined as $\pi(S) = \sum_{\{v,w\} \in S} \pi(v, w)$. Similarly, for a set E' of edges in G , the *weight* of E' is $\pi(E') = \sum_{[v,w] \in E'} \pi(v, w)$. Therefore, the instance (G, π, k) asks if there is a solution to G whose weight is bounded by k .

For a vertex v , denote by $N(v)$ the set of neighbors of v , and let $N[v] = N(v) \cup \{v\}$. For a vertex set X , $N[X] = \bigcup_{v \in X} N[v]$, and $N(X) = N[X] \setminus X$. For the vertex set X , define $\overline{X} = V \setminus X$. For two vertex subsets X and Y , denote by $E(X, Y)$ the set of edges that has one end in X and the other end in Y . For a vertex subset X , the edge set $E(X, \overline{X})$ is called the *cut of X* . The total cost of the cut of X is denoted by $\gamma(X) = \pi(E(X, \overline{X}))$. Obviously, $\gamma(X) = \gamma(\overline{X})$. For an instance (G, π, k) of the CLUSTER EDITING problem, denote by $\omega(G)$ the weight of an optimal (i.e., minimum weighted) solution to the graph G .

Behind all of the following lemmas is a very simple observation: in the objective graph $G \triangle S$ for any solution S to the graph G , each induced subgraph is also a union of disjoint cliques. Therefore, a solution S to the graph G restricted to an induced subgraph G' of G (i.e., the pairs of S in which both vertices are in G') is also a solution to the subgraph G' . This observation leads to the following *Cutting Lemma*.

Lemma 2.1 *Let $\mathcal{P} = \{V_1, V_2, \dots, V_p\}$ be a vertex partition of a graph G , and let $E_{\mathcal{P}}$ be the set of edges whose two ends belong to two different parts in \mathcal{P} . Then $\sum_{i=1}^p \omega(G[V_i]) \leq \omega(G) \leq \pi(E_{\mathcal{P}}) + \sum_{i=1}^p \omega(G[V_i])$.*

PROOF. Let S be an optimal solution to the graph G . For $1 \leq i \leq p$, let S_i be the subset of S such that each pair in S_i has both its vertices in V_i . As noted above, the set S_i is a solution to the graph $G[V_i]$, which implies $\omega(G[V_i]) \leq \pi(S_i)$. Thus,

$$\sum_{i=1}^p \omega(G[V_i]) \leq \sum_{i=1}^p \pi(S_i) \leq \pi(S) = \omega(G).$$

On the other hand, if we remove all edges in $E_{\mathcal{P}}$, and for each i , apply an optimal solution S'_i to the induced subgraph $G[V_i]$, we will obviously end up with a union of disjoint cliques. Therefore, these operations make a solution to the graph G whose weight is $\pi(E_{\mathcal{P}}) + \sum_{i=1}^p \pi(S'_i) = \pi(E_{\mathcal{P}}) + \sum_{i=1}^p \omega(G[V_i])$. This gives immediately $\omega(G) \leq \pi(E_{\mathcal{P}}) + \sum_{i=1}^p \omega(G[V_i])$. \square

Lemma 2.1 directly implies the following corollaries. First, if there is no edge between two different parts in the vertex partition \mathcal{P} , then Lemma 2.1 gives

Corollary 2.2 *Let G be a graph with connected components G_1, \dots, G_p , then $\omega(G) = \sum_{i=1}^p \omega(G_i)$, and every optimal solution to the graph G is a union of optimal solutions to the subgraphs G_1, \dots, G_p .*

When $p = 2$, i.e., the vertex partition is $\mathcal{P} = \{X, \overline{X}\}$, the edge set $E_{\mathcal{P}}$ becomes the cut $E(X, \overline{X})$, and $\pi(E(X, \overline{X})) = \gamma(X)$. Lemma 2.1 gives

Corollary 2.3 *Let $X \subseteq V$ be a vertex set, then $\omega(G[X]) + \omega(G[\overline{X}]) \leq \omega(G) \leq \omega(G[X]) + \omega(G[\overline{X}]) + \gamma(X)$.*

Corollary 2.4 *Let G be a graph, and let S^* be an optimal solution to G . For any subset X of vertices in G , if we let $S^*(X, \overline{X})$ be the subset of pairs in which one vertex is in X and the other vertex is in \overline{X} , then $\pi(S^*(X, \overline{X})) \leq \gamma(X)$.*

PROOF. The optimal solution S^* can be divided into three disjoint parts: the subset $S^*(X)$ of pairs in which both vertices are in X , the subset $S^*(\overline{X})$ of pairs in which both vertices are in \overline{X} , and the subset $S^*(X, \overline{X})$ of pairs in which one vertex is in X and the other vertex is in \overline{X} . By Corollary 2.3,

$$\omega(G) = \pi(S^*(X)) + \pi(S^*(\overline{X})) + \pi(S^*(X, \overline{X})) \leq \omega(G[X]) + \omega(G[\overline{X}]) + \gamma(X).$$

Since $\pi(S^*(X)) \geq \omega(G[X])$ and $\pi(S^*(\overline{X})) \geq \omega(G[\overline{X}])$, we get immediately $\pi(S^*(X, \overline{X})) \leq \gamma(X)$. \square

Corollary 2.4 can be informally described as “cut preferred” principle, which is fundamental for this problem. Similarly we have the following lemmas.

Lemma 2.5 *Let X be a subset of vertices in a graph G , and let S^* be any optimal solution to G . Let $S^*(V, \overline{X})$ be the set of pairs in S^* in which at least one vertex is in \overline{X} . Then $\omega(G) \geq \omega(G[X]) + \pi(S^*(V, \overline{X}))$.*

PROOF. The optimal solution S^* is divided into two disjoint parts: the subset $S^*(X)$ of pairs in which both vertices are in X , and the subset $S^*(V, \overline{X})$ of pairs in which at least one vertex is in \overline{X} . The set $S^*(X)$ is a solution to the induced subgraph $G[X]$. Therefore, $\pi(S^*(X)) \geq \omega(G[X])$. This gives

$$\omega(G) = \pi(S^*) = \pi(S^*(X)) + \pi(S^*(V, \overline{X})) \geq \omega(G[X]) + \pi(S^*(V, \overline{X})),$$

which proves the lemma. \square

Lemma 2.6 *Let X be a subset of vertices in a graph G , and let B_X be the set of vertices in X that are adjacent to vertices in \overline{X} . Then for any optimal solution S^* to G , if we let $S^*(B_X)$ be the set of pairs in S^* in which both vertices are in B_X , then $\omega(G) + \pi(S^*(B_X)) \geq \omega(G[X]) + \omega(G[\overline{X} \cup B_X])$.*

PROOF. Again, the optimal solution S^* can be divided into three disjoint parts: the subset $S^*(X)$ of pairs in which both vertices are in X , the subset $S^*(\overline{X})$ of pairs in which both vertices are in \overline{X} , and the subset $S^*(X, \overline{X})$ of pairs in which one vertex is in X and the other vertex is in \overline{X} . We also denote by $S^*(B_X, \overline{X})$ the subset of pairs in S^* in which one vertex is in B_X and the other vertex is in \overline{X} . Since $S^*(X)$ is a solution to the induced subgraph $G[X]$, we have

$$\begin{aligned} \omega(G) + \pi(S^*(B_X)) &= \pi(S^*(X)) + \pi(S^*(\overline{X})) + \pi(S^*(X, \overline{X})) + \pi(S^*(B_X)) \\ &\geq \omega(G[X]) + \pi(S^*(\overline{X})) + \pi(S^*(X, \overline{X})) + \pi(S^*(B_X)) \\ &\geq \omega(G[X]) + \pi(S^*(\overline{X})) + \pi(S^*(B_X, \overline{X})) + \pi(S^*(B_X)). \end{aligned}$$

The last inequality is because $B_X \subseteq X$, so $S^*(B_X, \overline{X}) \subseteq S^*(X, \overline{X})$. Since $S' = S^*(\overline{X}) \cup S^*(B_X, \overline{X}) \cup S^*(B_X)$ is the subset of pairs in S^* in which both vertices are in the induced subgraph $G[\overline{X} \cup B_X]$, S' is a solution to the induced subgraph $G[\overline{X} \cup B_X]$. This gives

$$\pi(S') = \pi(S^*(\overline{X})) + \pi(S^*(B_X, \overline{X})) + \pi(S^*(B_X)) \geq \omega(G[\overline{X} \cup B_X]),$$

which implies the lemma immediately. \square

The above results that reveal the relations between the structures of the CLUSTER EDITING problem and graph edge cuts not only form the basis for our kernelization results presented in the current paper, but also are of their own importance and interests.

3 The kernelization algorithm

Obviously, the number of different vertices included in a solution S of k vertex pairs to a graph G is upper bounded by $2k$. Thus, if we can also bound the number of vertices that are not included in S , we get a kernel. For such a vertex v , the clique containing v in $G \triangle S$ must be $G[N[v]]$. Inspired by this, our approach is to check the closed neighborhood $N[v]$ for each vertex v .

The observation is that if an induced subgraph (e.g. the closed neighborhood of a vertex) is very “dense inherently”, while is also “loosely connected to outside”, (*i.e.* there are very few edges in the cut of this subgraph), it might be cut off and solved separately. By the cutting lemmas, the size of a solution obtained as such should not be too far away from that of an optimal solution. Actually, we will figure out the conditions under which they are equal.

The subgraph we are considering is $N[v]$ for some vertex v . For the connection of $N[v]$ to outside, a good measurement is $\gamma(N[v])$. Thus, here we only need to define the density. A simple fact is that the fewer edges missing, the denser the subgraph is. Therefore, to measure the density of $N[v]$, we define the *deficiency* $\delta(v)$ of $N[v]$ as the total weight of anti-edges in $G[N[v]]$, which is formally given by $\delta(v) = \pi(\{\{x, y\} \mid x, y \in N(v), [x, y] \notin E\})$.

Suppose that $N[v]$ forms a single clique with no other vertices in the resulting graph $G \triangle S$. Then anti-edges of total weight $\delta(v)$ have to be added to make $N[v]$ a clique, and edges of total weight $\gamma(N[v])$ have to be deleted to make $N[v]$ disjoint. Based on this we define the *stable cost* of a vertex v as $\rho(v) = 2\delta(v) + \gamma(N[v])$, and we say $N[v]$ is *reducible* if $\rho(v) < |N[v]|$.

Lemma 3.1 *For any vertex v such that $N[v]$ is reducible, there is an optimal solution S^* to G such that the vertex set $N[v]$ is entirely contained in a single clique in the graph $G \triangle S^*$.*

PROOF. Let S be an optimal solution to the graph G , and pick any vertex v such that $N[v]$ is reducible, i.e., $\rho(v) < |N[v]|$. Suppose that $N[v]$ is not entirely contained in a single clique in $G \triangle S$, i.e., $N[v] = X \cup Y$, where $X \neq \emptyset$ and $Y \neq \emptyset$, such that Y is entirely contained in a clique C_1 in $G \triangle S$ while $X \cap C_1 = \emptyset$ (note that we do not assume that X is in a single clique in $G \triangle S$).

Inserting all missing edges between vertices in $N[v]$ will transform the induced subgraph $G[N[v]]$ into a clique. Therefore, $\omega(G[N[v]]) \leq \delta(v)$. Combining this with Corollary 2.3, we get

$$\begin{aligned} \omega(G) &\leq \omega(G[N[v]]) + \omega(G[\overline{N[v]}]) + \gamma(N[v]) \\ &\leq \delta(v) + \omega(G[\overline{N[v]}]) + \gamma(N[v]) \\ &= \omega(G[\overline{N[v]}]) + \rho(v) - \delta(v). \end{aligned} \tag{1}$$

Let $S(V, N[v])$ be the set of pairs in the solution S in which at least one vertex is in $N[v]$, and let $S(X, Y)$ be the set of pairs in S in which one vertex is in X and the other vertex is in Y . Also, let $P(X, Y)$ be the set of all pairs (x, y) such that $x \in X$ and $y \in Y$. Obviously, $\pi(S(V, N[v])) \geq \pi(S(X, Y))$ because $X \subseteq V$ and $Y \subseteq N[v]$. Moreover, since the solution S places the sets X and Y in different cliques, S must delete all edges between X and Y . Therefore $S(X, Y)$ is exactly the set of edges in G in which one end is in X and the other end is in Y . Also, by the definition of $\delta(v)$ and because both X and Y are subsets of $N[v]$, the sum of the weights of all anti-edges between X and Y is bounded by $\delta(v)$. Thus, we have $\pi(S(X, Y)) + \delta(v) \geq \pi(P(X, Y))$. Now by Lemma 2.5,

$$\begin{aligned} \omega(G) &\geq \omega(G[\overline{N[v]}]) + \pi(S(V, N[v])) \\ &\geq \omega(G[\overline{N[v]}]) + \pi(S(X, Y)) \\ &\geq \omega(G[\overline{N[v]}]) + \pi(P(X, Y)) - \delta(v). \end{aligned} \tag{2}$$

Combining (1) and (2), and noting that the weight of each vertex pair is at least 1, we get

$$|X||Y| \leq \pi(P(X, Y)) \leq \rho(v) < |N[v]| = |X| + |Y|. \tag{3}$$

This can hold true only when $|X| = 1$ or $|Y| = 1$. In both cases, we have $|X| \cdot |Y| = |X| + |Y| - 1$. Combining this with (3), and noting that all the quantities are integers, we must have

$$\pi(P(X, Y)) = \rho(v),$$

which, when combined with (1) and (2), gives

$$\omega(G) = \omega(G[\overline{N[v]}]) + \rho(v) - \delta(v) = \omega(G[\overline{N[v]}]) + \gamma(N[v]) + \delta(v). \tag{4}$$

Note that $\gamma(N[v]) + \delta(v)$ is the minimum cost to insert edges into and delete edges from the graph G to make $N[v]$ a disjoint clique. Therefore, Equality (4) shows that if we first apply edge insert/delete operations of minimum weight to make $N[v]$ a disjoint clique, then apply an optimal solution to the induced subgraph $G[\overline{N[v]}]$, then we have an optimal solution S^* to the graph G . This completes the proof of the lemma because the optimal solution S^* has the vertex set $N[v]$ entirely contained in a single clique in the graph $G \triangle S^*$. \square

Based on Lemma 3.1, we have the following reduction rule:

Step 1 *For a vertex v such that $N[v]$ is reducible, insert edges between anti-edges in $G[N[v]]$ to make $G[N[v]]$ a clique, and decrease k accordingly.*

After Step 1, the induced subgraph $G[N[v]]$ becomes a clique with $\delta(v) = 0$ and $\rho(v) = \gamma(N[v])$. Now we use the following rule to remove the vertices in $N(N[v])$ that are loosely connected to $N[v]$ (recall that $N(N[v])$ is the set of vertices that are not in $N[v]$ but adjacent to some vertices in $N[v]$, and that for two vertex subsets X and Y , $E(X, Y)$ denotes the set of edges that has one end in X and the other end in Y).

Step 2 *Let v be a vertex such that $N[v]$ is reducible on which Step 1 has been applied. For each vertex x in $N(N[v])$, if $\pi(E(x, N(v))) \leq |N[v]|/2$, then delete all edges in $E(x, N(v))$ and decrease k accordingly.*

We say that a reduction step R is *safe* if after edge operations of cost c_R by the step, we obtain a new graph G' such that the original graph G has a solution of weight bounded by k if and only if the new graph G' has a solution of weight bounded by $k - c_R$.

Lemma 3.2 *Step 2 is safe.*

PROOF. By Lemma 3.1, there is an optimal solution S to the graph G such that $N[v]$ is entirely contained in a single clique C in the graph $G \triangle S$. We first prove, by contradiction, that the clique C containing $N[v]$ in the graph $G \triangle S$ has at most one vertex in $\overline{N[v]}$. Suppose that there are r vertices u_1, \dots, u_r in $\overline{N[v]}$ that are in C , where $r \geq 2$. For $1 \leq i \leq r$, denote by c_i the total weight of all edges between u_i and $N[v]$, and by c'_i the total weight of all pairs (both edges and anti-edges) between u_i and $N[v]$. Note that $c'_i \geq |N[v]|$ and $\sum_{i=1}^r c_i \leq \gamma(N[v])$. Then in the optimal solution S to G , the total weight of the edges inserted between $N[v]$ and $\overline{N[v]}$ is at least

$$\begin{aligned} \sum_{i=1}^r (c'_i - c_i) &\geq \sum_{i=1}^r (|N[v]| - c_i) = r|N[v]| - \sum_{i=1}^r c_i \\ &\geq r|N[v]| - \gamma(N[v]) \geq 2|N[v]| - \gamma(N[v]) \\ &> 2|N[v]| - |N[v]| = |N[v]| > \gamma(N[v]), \end{aligned}$$

where we have used the fact $|N[v]| > \gamma(N[v])$ (this is because by the conditions of the step, $\rho(v) = 2\delta(v) + \gamma(N[v]) < |N[v]|$). But this contradicts Corollary 2.4.

Therefore, there is at most one vertex x in $N(N[v])$ that is in the clique C containing $N[v]$ in the graph $G \triangle S$. Such a vertex x must satisfy the condition $\pi(E(x, N(v))) > |N[v]|/2$: otherwise deleting all edges in $E(x, N(v))$ would result in a solution that is at least as good as the one that inserts all missing edges between x and $N[v]$ and makes $N[v] \cup \{x\}$ a clique. Thus, for a vertex x in $N(N[v])$ with $\pi(E(x, N(v))) \leq |N[v]|/2$, we can always assume that x is not in the clique containing $N[v]$ in the graph $G \triangle S$. In consequence, deleting all edges in $E(x, N(v))$ for such a vertex x is safe. \square

The structure of $N[v]$ changes after the above steps. The result can be in two possible cases: (1) no vertex in $N(N[v])$ survives, and $N[v]$ becomes an isolated clique – then by Corollary 2.2, we can simply delete the clique; and (2) there is one vertex x remaining in $N(N[v])$ (note that there cannot be more than one vertices surviving – otherwise it would contradict the assumption $\gamma(N[v]) \leq \rho(v) < |N[v]|$). In case (2), the vertex set $N[v]$ can be divided into two parts $X = N[v] \cap N(x)$ and $Y = N[v] \setminus X$. From the proofs of the above lemmas, we are left with only two options: either disconnecting X from x with edge cost c_X , or connecting Y and x with edge cost c_Y . Obviously $c_X > c_Y$. Since both options can be regarded as connection or disconnection between the vertex set $N[v]$ and the vertex x , we can further reduce the graph using the following reduction step:

Step 3 *Let v be a vertex such that $N[v]$ is reducible on which Steps 1 and 2 have been applied. If there still exists a vertex x in $N(N[v])$, then merge $N[v]$ into a single vertex v' , connect v' to x with weight $c_X - c_Y$, set weight of each anti-edge between v' and other vertex to $+\infty$, and decrease k by c_Y .*

The correctness of this step immediately follows from above argument.

Note that the conditions for all the above steps are only checked once. If they are satisfied, we apply all three steps one by one, or else we do nothing at all. So they are actually the parts of a single reduction rule presented as follows:

The Rule. Let v be a vertex satisfying $2\delta(v) + \gamma(N[v]) < |N[v]|$, then:

1. add edges to make $G[N[v]]$ a clique and decrease k accordingly;
2. for each vertex x in $N(N[v])$ with $\pi(E(x, N[v])) \leq |N[v]|/2$, remove all edges in $E(x, N[v])$ and decrease k accordingly;
3. if a vertex x in $N(N[v])$ survives, merge $N[v]$ into a single vertex (as described above) and decrease k accordingly.

Now the following lemma implies Theorem 1.1 directly.

Lemma 3.3 *If an instance of the weighted CLUSTER EDITING problem reduced by our reduction rule has more than $2k$ vertices, it has no solution of weight $\leq k$.*

PROOF. We divide the cost of inserting/deleting a pair $\{u, v\}$ into two halves and assign them to u and v equally. Thereafter we count the costs on all vertices.

For any two vertices with distance 2, at most one of them is not shown in a solution S : otherwise they would have to belong to the same clique in $G \triangle S$ because of their common neighbors but the edge between them is missing. Thus, if we let $\{v_1, v_2, \dots, v_r\}$ be the vertices not shown in S , then each two of their closed neighbors $\{N[v_1], N[v_2], \dots, N[v_r]\}$ are either the same (when they are in the same simple series module) or mutually disjoint. The cost in each $N[v_i]$ is $\delta(v_i) + \gamma(N[v_i])/2 = \rho(v_i)/2$, which is at least $|N[v_i]|/2$, because by our reduction rule, in the reduced instance we have $\rho(v) \geq |N[v]|$ for each vertex v . Each of the vertices not in any of $N[v_i]$ is contained in at least one pair of S and therefore bears cost at least $1/2$. Summing them up, we get a lower bound for the total cost at least $|V|/2$. Thus, if the solution S has a weight bounded by k , then $k \geq |V|/2$, i.e., the graph has at most $2k$ vertices. \square

4 On unweighted and real-weighted versions

We now show how to adapt the algorithm in the previous section to support unweighted and real-weighted versions. Only slight modifications are required. Therefore, the proof of the correctness of them is omitted for the lack of space.

Unweighted version. The kernelization algorithm presented does not work for unweighted version. The trouble arises in Step 3, where merging $N[v]$ is not a valid operation in an unweighted graph. Fortunately, this can be easily circumvented, by replacing Step 3 by the following new rule:

Step 3 (U) *Let v be a vertex such that $N[v]$ is reducible on which Steps 1 and 2 have been applied. If there still exists a vertex x in $N(N[v])$, then replace $N[v]$ by a complete subgraph $K_{|X|-|Y|}$, and connect x to all vertices of this subgraph.*

The correctness of this new rule is similar to the arguments in last section, and it is easy to check the first two rules apply for the unweighted version. Moreover, the proof of Lemma 3.3 can be easily adapted with the new rule.

Real-weighted version. There are even more troubles when weights are allowed to be real numbers, instead of only positive integers. The first problem is that, without the integrality, (3) cannot imply (4). This is fixable by changing the definition of reducible closed neighborhood from $\rho(v) < |N(v)|$ to $\rho(v) \leq |N(v)| - 1$ (they are equivalent for integers), then (3) becomes

$$|X||Y| \leq \pi(P(X, Y)) \leq \rho(v) \leq |N[v]| - 1 = |X| + |Y| - 1. \quad (5)$$

Formulated on reducible closed neighborhood, Steps 1 and 2 remain the same.

The second problem is Step 3, in which we need to maintain the validity of weights. Recall that we demand all weights be at least 1 for weight functions. This, although trivially holds for integral weight functions, will be problematic for real weight functions. More specifically, in Step 3, the edge $[x, v']$ could be assigned a weight $c_X - c_Y < 1$ when c_X and c_Y differ by less than 1. This can be fixed with an extension of Step 3:

Step 3 (R) *Let v be a vertex such that $N[v]$ is reducible and that on which Steps 1 and 2 have been applied. If there still exists a vertex x in $N(N[v])$, then*

- *if $c_X - c_Y \geq 1$, merge $N[v]$ into a single vertex v' , connect v' to x with weight $c_X - c_Y$, set weight of each anti-edge between v' and other vertex to $+\infty$, and decrease k by c_Y ;*
- *if $c_X - c_Y < 1$, merge $N[v]$ into two vertices v' and v'' , connect v' to x with weight 2, and v' to v'' with weight $2 - (c_X - c_Y)$, set weight of each anti-edge between v', v'' to other vertex to $+\infty$, and decrease k by $c_X - 2$.*

The new case is just to maintain the validity of the weight, and does not make a real difference from the original case. However, there does exist one subtlety we need to point out, that is, the second case might increase k slightly, and this happens when $c_X - 2 \leq 0$, then we are actually increase k by $2 - c_X$. We do not worry about this trouble due to both theoretical and practical reasons. Theoretically, the definition of kernelization does not forbid increasing k , and we refer readers who feel uncomfortable with this to the monographs [13, 16, 24]. Practically, 1) it will not really enlarge or complicate the graph, and therefore any reasonable algorithms will work as the same; 2) this case will not happen too much, otherwise the graph should be very similar to a star, and easy to solve; 3) even using the original value of k , our kernel size is bounded by $3k$.

The proof of Lemma 3.3 goes almost the same, with only the constant slightly enlarged. Due to the relaxation of the condition of reducible closed neighborhood from $\rho(v) < |N(v)|$ to $\rho(v) \leq |N(v)| - 1$, the number of vertices in the kernel for real-weighted version is bounded by $2.5k$.

5 Discussion

One very interesting observation is that for the unweighted version, by the definition of simple series modules, all of the following are exactly the same:

$$N[u] = N[M], \quad \delta(u) = \delta(M), \quad \text{and} \quad \gamma(N[u]) = \gamma(N[M]),$$

where M is the simple series module containing vertex u , and $\delta(M)$ is a natural generalization of definition $\delta(v)$. Thus it does not matter we use the module or any vertex in it, that is, every vertex is a full representative for the simple series module it lies in. Although there has been a long list of linear algorithms for finding modular decomposition for an undirected graph (see a comprehensive survey by de Montgolfier [23]), it is very time-consuming because the big constant hidden behind the big-O [26], and considering that the modular decomposition needs to be re-constructed after each iteration, this will be helpful. It is somehow surprising that the previous kernelization algorithms can be significantly simplified by avoiding modular decomposition. Being more surprising, this enables our approach to apply for the weighted version, because one major weakness of modular decomposition is its inability in handling weights.

One similar problem on inconsistent information is the FEEDBACK VERTEX SET ON TOURNAMENT (FAST) problem, which asks the reverse of minimum number of arcs to make a tournament transitive. Given the striking resemblances between CLUSTER EDITING and FAST, and a series of “one-stone-two-birds” approximation algorithms [1, 27] which only take advantage of the commonalities between them, we are strongly attempted to compare the results of these two problems from the parameterized aspect.

For the kernelization, our result already matches the best kernel, $(2 + \epsilon)k$ for weighted FAST of Bessy et al. [5], which is obtained based on a complicated PTAS [21].

For the algorithms, Alon et al. [2] managed to generalize the famous color coding approach to give a subexponential FPT algorithm for FAST. This is the first subexponential FPT algorithm out of bidimensionality theory, which was a systematic way to obtain subexponential algorithms, and has been intensively studied. This is an exciting work, and opens a new direction for further work. Indeed, immediately after the appearance of [2], for unweighted version, Feige reported an improved algorithm [14] that is far simpler and uses pure combinatorial approach. Recently, Karpinski and Schudy reached the same result for weighted version [20]. Based on their striking resemblances, we conjecture that there is also a subexponential algorithm for the CLUSTER EDITING problem.

References

- [1] Ailon, N., Charikar, M., Newman, A.: Aggregating inconsistent information: ranking and clustering. *J. ACM* 55(5), Article 23, 1-27 (2008)
- [2] Alon, N., Lokshtanov, D., Saurabh, S.: Fast FAST. In: *ICALP*, LNCS vol. 5555, pp. 49-58. Springer (2009)
- [3] Bansal, N., Blum, A., Chawla, S.: Correlation clustering. *Machine Learning* 56(1), 89-113 (2004)
- [4] Berkhin, P.: A survey of clustering data mining techniques. In: *Grouping Multidimensional Data*, Springer Berlin Heidelberg, pp. 25-71 (2006)
- [5] Bessy, S., Fomin, F. V., Gaspers, S., Paul, C., Perez, A., Saurabh, S., Thomassé, S.: Kernels for feedback arc set in tournaments. In: *CoRR*, abs/0907.2165 (2009)
- [6] Böcker, S., Briesemeister, S., Klau, G. W.: Exact algorithms for cluster editing: evaluation and experiments. *Algorithmica*, in press.
- [7] Böcker, S., Briesemeister, S., Bui, Q. B. A., Truss, A.: Going weighted: parameterized algorithms for cluster editing. *Theoretical Computer Science* 410, 5467-5480 (2009).
- [8] Charikar, M., Guruswami, V., Wirth, A.: Clustering with qualitative information. *Journal of Computer and System Sciences* 71(3), 360-383 (2005)
- [9] Chen, J., Meng, J.: A $2k$ kernel for the cluster editing problem. In: *COCOON 2010*, LNCS vol. 6196, pp. 459-468, Springer (2010)
- [10] Chen, Z.-Z., Jiang, T., Lin, G.: Computing phylogenetic roots with bounded degrees and errors. *Siam J. Comp.* 32(4), 864-879 (2003)
- [11] Cunningham, W. H., Edmonds, J.: A combinatorial decomposition theory. *Canad. J. Math.* 32(3), 734-765 (1980)
- [12] Dean, J., Henzinger, M. R.: Finding related pages in the World Wide Web. *Computer Networks* 31, 1467-1479 (1999)
- [13] Downey, R. G., Fellows, M. R.: *Parameterized Complexity*, Springer (1999)
- [14] Feige, U.: Faster FAST. In: *CoRR*, abs/0911.5094 (2009)
- [15] Fellows, M. R., Langston, M. A., Rosamond, F. A., Shaw, P.: Efficient parameterized preprocessing for cluster editing. In *FCT*, LNCS vol. 4639, pp. 312-321, Springer (2007)
- [16] Flum, J., Grohe, M.: *Parameterized Complexity Theory*, Springer (2006)
- [17] Gramm, J., Guo, J., Hüffner, F., Niedermeier, R.: Graph-modeled data clustering: exact algorithms for clique generation. *Theory of Computing Systems* 38(4), 373-392 (2005)
- [18] Guo J.: A more effective linear kernelization for cluster editing. *Theor. Comput. Sci.* 410(8-10), 718-726 (2009)
- [19] Hearst, M. A., Pedersen, J. O.: Reexamining the cluster hypothesis: scatter/gather on retrieval results. In: *Proceedings of SIGIR*, pp. 76-84 (1996)
- [20] Karpinski, M., Schudy, W.: Faster algorithms for feedback arc set tournament, Kemeny rank aggregation and betweenness tournament. In: *CoRR*, abs/1006.4396 (2010)
- [21] Kenyon-Mathieu, C., Schudy, W.: How to rank with few errors. In: *ACM Symposium on Theory of Computing (STOC)*, pp. 95-103 (2007)
- [22] Möhring, R. H., Radermacher, F. J.: Substitution decomposition for discrete structures and connections with combinatorial optimization. *Ann. Discrete Math.*, 19(95), 257-355, North-Holland mathematics studies, (1984)
- [23] de Montgolfier, F.: *Décomposition modulaire des graphes. Théorie, extensions et algorithmes*. Thèse de doctorat, Université Montpellier II (2003)
- [24] Niedermeier, R.: *Invitation to Fixed-Parameter Algorithms*, Oxford University Press (2006)
- [25] Shamir, R., Sharan, R., Tsur, D.: Cluster graph modification problems. *Discrete Appl. Math.* 144 (1-2), 173-182 (2004)
- [26] Tedder, M.; Corneil, D.; Habib, M.; Paul, C.: Simpler linear-time modular decomposition via recursive factorizing permutations. In: *ICALP*, LNCS vol. 5125, pp. 634-645. Springer-Verlag (2008)
- [27] van Zuylen, A., Williamson, D. P.: Deterministic pivoting algorithms for constrained ranking and clustering problems. *Mathematics of Operations Research* 34(3), 594-620 (2009)