# Clustering with partial information*

Hans L. Bodlaender† Michael R. Fellows‡ Pinar Heggernes§
Federico Mancini§ Charis Papadopoulos§ Frances Rosamond‡

**Abstract**

The Correlation Clustering problem, also known as the Cluster Editing problem, seeks to edit a given graph by adding and deleting edges to obtain a collection of vertex-disjoint cliques, such that the editing cost is minimized. The Edge Clique Partitioning problem seeks to partition the edges of a given graph into edge-disjoint cliques, such that the number of cliques is minimized. Both problems are known to be NP-hard, and they have been previously studied with respect to approximation and fixed parameter tractability. In this paper we study these two problems in a more general setting that we term *fuzzy graphs*, where the input graphs may have missing information, meaning that whether or not there is an edge between some pairs of vertices of the input graph can be *undecided*.

For fuzzy graphs the Correlation Clustering and Edge Clique Partitioning problems have previously been studied only with respect to approximation. Here we give parameterized algorithms based on kernelization for both problems. We prove that the Correlation Clustering problem is fixed-parameter tractable on fuzzy graphs when parameterized by $(k, r)$, where $k$ is the editing cost and $r$ is the minimum number of vertices required to cover the undecided edges. In particular we show that it has a polynomial-time reduction to a problem kernel on $O(k^2 + r)$ vertices. We provide an analogous result for the Edge Clique Partitioning problem on fuzzy graphs. Using $(k, r)$ as parameters, where $k$ bounds the size of the partition, and $r$ is the minimum number of vertices required to cover the undecided edges, we describe a polynomial-time kernelization to a problem kernel on $O(k^4 \cdot 3^r)$ vertices. This implies fixed-parameter tractability for this parameterization. Furthermore we also show that parameterizing only by the number of cliques $k$, is not enough to obtain fixed-parameter tractability. The problem remains, in fact, NP-hard for each fixed $k > 2$.

## 1 Introduction

Given a complete graph with labels $\langle + \rangle$ or $\langle - \rangle$ on each edge, the Correlation Clustering problem is to partition the vertices into clusters so that the number of $\langle - \rangle$ edges inside each

*This work is supported by the Research Council of Norway. A preliminary version of this paper was presented at MFCS 2008 [6].

†Department of Information and Computing Sciences, Utrecht University, The Netherlands. hansb@cs.uu.nl

‡PCRU, Office of DVC (Research), University of Newcastle, Australia. michael.fellows@newcastle.edu.au, frances.rosamond@newcastle.edu.au

§Department of Informatics, University of Bergen, Norway. pinar@ii.uib.no, federico@ii.uib.no, charis@cs.uoi.gr

cluster plus the number of $\langle + \rangle$ edges between the clusters, is minimized. Taking $\langle + \rangle$ edges as *edges* and $\langle - \rangle$ edges as *non-edges*, this problem is equivalent to the Cluster Editing problem, where we are given an ordinary graph and asked to add and delete the total minimum number of edges so that the resulting graph is a collection of disconnected (i.e., vertex-disjoint) cliques. The Correlation Clustering (equivalent to Cluster Editing) problem has been proven NP-hard several times [2, 3], as it has been discovered and rediscovered in various applications areas, such as hierarchical tree clustering [22], computational biology [4, 29], and phylogenetic trees [10]. General versions of the Correlation Clustering problem have been defined [1, 5] and studied from the point of view of approximation [9, 11, 12, 14]. The second problem that we study in this paper is the Edge Clique Partitioning problem, which asks us to partition the edges of a given graph into edge-disjoint cliques such that the number of cliques in the partitioned graph is minimized. This problem is NP-hard [27] for general graphs, but also for $K_4$-free and even chordal graphs [23].

A key point of what we offer here is to expand the investigation of both problems to inputs consisting of *fuzzy graphs*, where some pairs of vertices of the input may have an *undetermined*, *unknown* or *undecided* relation. Fuzzy graphs are also called *incomplete graphs* in the literature [9, 11, 12, 14]. For many applications, such a notion clearly adds to the realism of the modeling in an important way. To mention one application area where a similar idea has been considered before, in bioinformatics the notion of "sandwich graph problems" has played a useful role [18].

NP-hard problems remain hard also on fuzzy graphs, as they are a generalization of ordinary graphs. Hence we investigate their tractability from a parameterized complexity point of view, and we try to understand which structural parameters are more suitable to attack problems on fuzzy graphs.

A problem is fixed parameter tractable (FPT) if its input can be partitioned into a main part of size $n$ and a parameter (usually an integer) $k$ so that there is an algorithm that solves the problem in time $O(n^c \cdot f(k))$, where $f$ is a computable function and $c$ is a fixed constant [13]. A *kernel* is an instance of the problem smaller than the input, such that the problem has a solution on the input if and only if it has a solution on the kernel. It is well known that a problem is FPT if and only if a kernel of size $g(k)$ can be computed from the input in polynomial time, for a computable function $g$ [13, 26].

The fixed-parameter tractability of the Cluster Editing problem (for ordinary non-fuzzy graphs) has been shown, with a series of improvements in [8, 19, 28], when using the editing cost $k$ as parameter. The problem has also been shown to admit a linear kernelization [16, 20]. On fuzzy graphs, it is not known whether using only $k$ as parameter, ensures fixed parameter tractability. Here we relax the problem by introducing a new parameter $r$, that represents the minimum number of vertices required to cover the undecided edges. By parameterizing the Cluster Editing problem by $(k, r)$, we show that the problem admits a quadratic kernel, specifically on $O(k^2 + r)$ vertices, and therefore the problem is FPT also for fuzzy graphs. Furthermore the result holds also when the fuzzy graph is weighted.

We continue by showing that the parameter $r$ might indeed be a good choice to capture the structure of *fuzzy* problems in general, not only for the specific case of Cluster Editing. For that purpose we study the Edge Clique Partitioning problem. When the natural parameter alone is not enough any more to handle the complexity added by the fuzzy edges, using $r$ as second parameter yields again an FPT algorithm.

The Edge Clique Partitioning problem has been recently shown to be FPT in [25],

when parameterized by the number $k$ of cliques that the edges can be partitioned into. In their work the authors give a quadratic kernel for it. The corresponding parameterization on fuzzy graphs asks, given a fixed $k$, whether the fuzzy edges can be turned into edges and non-edges so that the resulting set of edges can be partitioned into at most $k$ edge-disjoint cliques. We prove that the problem becomes hard when the input is a fuzzy graph, namely NP-complete for any fixed $k \geq 3$. Parameterizing only by $k$ is thus not enough to ensure fixed-parameter tractability. However, if we parameterize by $(k, r)$, then the problem becomes FPT, and admits a polynomial time kernelization to a kernel on $O(k^4 \cdot 3^r)$ vertices.

## 2 Notation and definitions

For an undirected graph $G = (V, E)$, we denote its vertex set by $V(G) = V$ and edge set by $E(G) = E$ with $n = |V|$. The set of *neighbors* of $v \in V$ is $N_G(v) = \{u \mid uv \in E\}$, and the *degree* of $v$ is $d_G(v) = |N_G(v)|$. In addition, $N_G[v] = N_G(v) \cup \{v\}$. Analogously, for a set $S \subseteq V$, $N_G[S] = \cup_{x \in S} N_G[x]$ and $N_G(S) = N_G[S] \setminus S$. We omit subscripts when there is no ambiguity. An *induced subgraph* of $G$ by $U \subseteq V$ is the graph $G[U] = (U, E_U)$, where $E_U = \{xv \in E \mid x, v \in U\}$. Given a vertex $x$ of $G$, we denote the graph $G[V \setminus \{x\}]$ by $G - x$. In addition, for a set of edges $M \subset E$, we define $G(M) = (\{x \mid \exists u, xu \in M\},\ M)$.

A graph is *complete* if every pair of vertices are adjacent. If a subgraph is complete then it is called a *clique*. If $G[K]$ is a clique for $K \subset V$, we also say that $K$ is a clique. If $G(M)$ is a clique for $M \subset E$, we also say that $M$ is a clique. A vertex subset $S \subseteq V$ is a *vertex cover* if every edge of $G$ has at least one endpoint in $S$. A *connected component* is a maximal connected subgraph.

We define a *fuzzy graph* (also known as *incomplete graph*) $G = (V, E, F)$ to be a graph with two types of edges:

- $E$ is the set of *real edges* (also known as *positive edges*),
- $F$ is the set of *fuzzy edges* (also known as *unweighted edges*), and
- for all other pairs of vertices in the graph we say that we have *non-edges* (also known as *negative edges*).

When we decide for each fuzzy edge whether it should become a real edge or a non-edge, we say that we *realize* the fuzzy edges. The resulting graph is called a *normalization* of the fuzzy graph. Formally we say that $(R^+, R^-)$ with $F = R^+ \cup R^-$ is a *realization* of $F$ into real edges $R^+$ and non-edges $R^-$ such that $G' = (V,\ E \cup R^+)$ is the corresponding normalization of $G = (V, E, F)$. When we speak about the *connected components of a fuzzy graph*, we mean the connected components of the graph obtained by turning all fuzzy edges into non-edges. So a *connected fuzzy graph* is a fuzzy graph where between any two vertices there is a path of real edges.

## 3 Parameterized cluster editing with partial information

A *cluster graph* is a graph where each connected component is a clique. In this section we study the problem of editing a weighted fuzzy graph $G = (V, E, F)$ to obtain a cluster graph. *Editing* means turning some real edges into non-edges (*deleting*), turning some non-edges into

real edges (*adding*), and turning all fuzzy edges into either real edges or non-edges. Each edge and non-edge is associated with a positive weight, whereas each fuzzy edge has weight 0. The *cost* of an edit is the sum of the weights of the deleted and added edges, and the goal is to minimize the cost. The problem is formally defined as follows.

Weighted Fuzzy Cluster Editing (WFCE)
*Instance:* A fuzzy graph $G = (V, E, F)$, a weight function $w : V \times V \to \mathbb{N}$ such that $w(uv) = 0$ if $uv \in F$ and $w(uv) > 0$ if $uv \notin F$, and a natural number $k \geq 0$.
*Question:* Is there a set $M \subseteq V \times V$ such that: $G' = (V, (E \setminus M) \cup (M \setminus E))$ is a cluster graph and $\sum_{uv \in M} w(uv) \leq k$?

First we characterize the fuzzy graphs that can be turned into a cluster graph just by realizing the fuzzy edges, that is, without any editing cost. As it was already noted in [14] and [9], these graphs can be defined by a family of forbidden induced (fuzzy) subgraphs, which they called *erroneous cycles* and *negative polygon* respectively. Since they are, in practice, a fuzzy version of an induced path, we denote them as $P_l^f = \{v_1, v_2, ..., v_l\}$, where: $l \geq 3$; $v_i v_{i+1}$ is a real edge for every $1 \leq i \leq l-1$; $v_1 v_l$ is a non-edge; and all the other pairs of vertices are joined by fuzzy edges (see Figure 1). In terms of our notation the results of [9, 14] imply the following theorem.

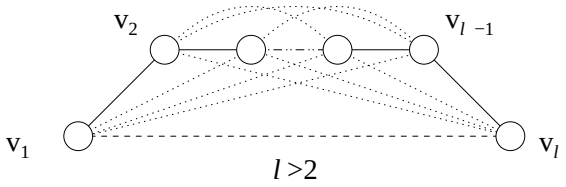


Figure 1: Real edges are represented by continuous lines, non-edges by dashed lines, and fuzzy edges by dotted lines.

**Theorem 3.1** *Let $G$ be a fuzzy graph. Then there exists a realization of the fuzzy edges that results in a cluster graph without editing any real edge or non-edge if and only if $G$ does not contain any induced subgraph isomorphic to a $P_l^f$ for $l \geq 3$.*

The $k$-Weighted Fuzzy Cluster Editing problem ($k$-WFCE) is the WFCE problem where we choose $k$ as the parameter of the problem. The complexity of $k$-WFCE is open even for the unweighted case. The characterization given in Theorem 3.1 is through an infinite set of forbidden induced subgraphs, and hence an FPT algorithm for $k$-WFCE does not follow from the results of Cai [8].

In order to give an FPT algorithm, we introduce an additional parameter. We define a *fuzzy vertex cover* of a fuzzy graph to be a vertex subset $S$ such that each fuzzy edge has an endpoint in $S$. The new parameter is $r = |S|$ where $S$ is a smallest fuzzy vertex cover of $G$. We call the corresponding new problem the $(k, r)$-Weighted Fuzzy Cluster Editing, or $(k, r)$-WFCE, problem.

Observe that checking whether a fuzzy graph $G = (V, E, F)$ has a fuzzy vertex cover of size at most $r$ is FPT when parameterized by $r$. To do this we create a non-fuzzy graph $G'$ from $G(F)$ by turning all real edges of $G(F)$ into non-edges and all fuzzy edges into real edges. It is easy to see that $G$ has a fuzzy vertex cover with at most $r$ vertices if and only if $G'$ has a

vertex cover of at most $r$ vertices. Since the $r$-VERTEX COVER problem is well known to be FPT, our claim follows.

### 3.1 Kernel for the $(k, r)$-Weighted Fuzzy Cluster Editing problem

We show fixed-parameter tractability by giving a set of rules that either enable us to answer NO, or produce a kernel of size $O(k^2 + r)$ in polynomial time, for the $(k, r)$-WFCE problem. First we give a general observation based on the fact cluster graphs are hereditary, and that, by definition, only fuzzy edges and non-edges are present among connected components of a fuzzy graph.

**Observation 3.2** *Let $G$ be a weighted fuzzy graph with connected components $C_1, \ldots, C_l$. Then $G$ can be made into a cluster graph with editing cost at most $k$ if and only if each connected component $C_i$ can be made into a cluster graph with editing cost at most $k_i$, such that $\sum_{1 \leq i \leq l} k_i \leq k$.*

Now we start presenting the rules, that are mostly self-explanatory. We will not give sharp bounds on the running time of each rule; instead, we will limit the explanation to why they can be executed in polynomial time. Our main goal is to prove that there exists a quadratic kernel that can be computed in polynomial time, and therefore the $(k, r)$-WFCE problem is FPT.

**Rule 3.1.1** *If there is a connected component $C$ with no non-edges, remove $C$.*

**Lemma 3.3** *Rule 3.1.1 is correct and can be applied in $O(|V| + |E| + |F|)$ time.*

**Proof.** Since such a connected component can be made into a clique by only realizing fuzzy edges, removing it will not affect the final result by Observation 3.2. Finding the connected components of a graph and checking its edges can be clearly done in linear time. ∎

**Rule 3.1.2** *If Rule 3.1.1 does not apply and there are more than $k+1$ connected components, then answer* NO.

**Lemma 3.4** *Rule 3.1.2 is correct and can be applied in $O(|V| + |E| + |F|)$ time.*

**Proof.** If Rule 3.1.1 does not apply, then in every connected component there is at least one non-edge, and at least one path of real edges connecting its endpoints. This means that there exists an erroneous cycle in each connected component, and each of them requires at least one editing to be destroyed. As there are more than $k$ disjoint fuzzy paths, the result follows. Checking the number of connected components of a graph can be obtained by a linear-time graph traversal. ∎

For the following rule, note that a *minimum cut* between two vertices $u$ and $v$ is the minimum total weight of a collection of real edges that must be deleted so that between $u$ and $v$ there is no path of real edges.

**Rule 3.1.3** *If there are vertices $u$ and $v$ such that the value of a minimum cut between them is at least $k + 1$, then replace $u$ and $v$ with a unique vertex $x$, and for every other vertex $z$ create the edge $xz$ as following, depending on the edges $e = uz$ and for $f = vz$:*

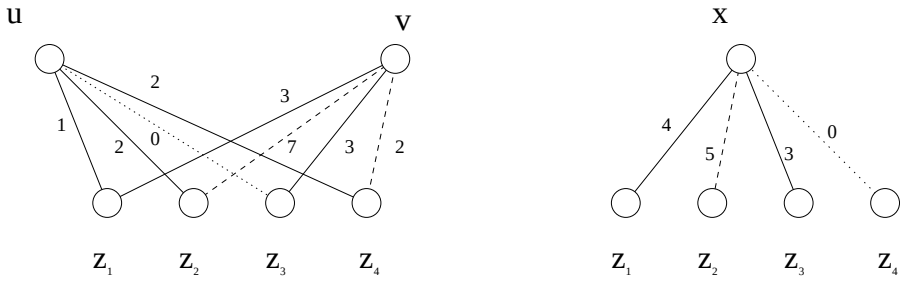


Figure 2: An example of the application of Rule 3.1.3. Real edges are represented by continuous lines, non-edges by dashed lines, and fuzzy edges by dotted lines.

1. *If $e$ is fuzzy then let $xz = f$, and vice-versa.*
2. *If $e$ and $f$ are both real edges or both non-edges, then let $xz$ be a real edge or non-edge, respectively, with $w(xz) = w(e) + w(f)$.*
3. *If, without loss of generality, $e$ is a real edge and $f$ is non-edge, then:*
   (a) *If $w(e) > w(f)$, then let $xz$ be a real edge with $w(xz) = w(e) - w(f)$ and let $k = k - w(f)$.*
   (b) *If $w(e) < w(f)$, then let $xz$ be a non-edge with $w(xz) = w(f) - w(e)$ and let $k = k - w(e)$.*
   (c) *If $w(e) = w(f)$, then let $xz$ be a fuzzy edge and let $k = k - w(e)$.*
4. *If $uv$ was a non-edge then let $k = k - w(uv)$.*
5. *If now $k < 0$, answer* NO.

**Lemma 3.5** *Rule 3.1.3 is correct and can be applied in polynomial time.*

**Proof.** The idea of the rule is basically that, given an input weighted graph $G$, if two vertices $u$ and $v$ cannot be separated by removing edges of total weight less than $k + 1$, then they must belong to the same cluster in every possible solution. This implies also that in every possible solution, $u$ and $v$ must be adjacent and every other vertex in the graph will have to be either adjacent to both $u$ and $v$, or to none of them. With this in mind, we can pay the cost for making $u$ and $v$ adjacent (point 4) and construct a new graph $H$ where $u$ and $v$ are replaced by a unique vertex $x$ such that, given any other vertex $z$ of $G$, making $z$ incident to $x$ in $H$ will cost the the same as making $z$ incident to both $u$ and $v$ in $G$, and similarly, disconnecting $z$ from $x$ will have the same editing cost as disconnecting $z$ from both $u$ and $v$. Once we prove that these editing costs are equivalent, since $G - \{u, v\}$ is identical to $H - x$, it will be straightforward to see that $G$ has a solution of cost at most $k$ if and only if $H$ has a solution of cost at most $k'$, where $k'$ is the new $k$ obtained after applying Rule 3.1.3.

We will now show that the editing costs in $G$ and $H$ are equivalent.

If the two edges $vz$ and $uz$ are of different type, there is always a minimum editing cost (possibly 0) to be paid to make $z$ adjacent either to both or to none of them, i.e., the $\min\{w(uz), w(vz)\}$, because for every possible editing choice at least one of them must be turned into a different type (points 1, 3a, 3b and 3c). Once we decrease the parameter $k$

by this fixed editing cost, if the new $k$ is non-negative (point 5), we can create the new edge $xz$ of the same type of the edge of maximum weight between $uz$ and $vz$, and with $w(xz) = |w(uz) - w(vz)|$. We explain the reason by taking as an example the case where $uz$ is a real edge and $vz$ a non-edge, and $w(uz) > w(vz)$. If we pay $w(vz)$, we can imagine that either we have paid to make $vz$ into a real edge, so that we can consider $z$ adjacent to both $u$ and $v$, or that we have already paid some of the editing cost to make $uz$ into a non-edge, so that only $w(uz) - w(vz)$ is left to pay to disconnect completely $z$ from both $u$ and $v$. This is clearly equivalent to having a real edge $xz$ with weight $w(uz) - w(vz)$ in $H$. All other cases are identical to this one, with the exception of the case when $w(uz) - w(vz) = 0$, in which case both the possible editions have completely been paid for and $xz$ is clearly equivalent to a fuzzy edge.

If the edges are of the same type, instead, the minimum editing cost is 0 because there might be a solution where no editing is necessary for these edges (Rule 2). In this situation, the new edge $xz$ will be simply an edge of the same type of both $uz$ and $vz$, with weight $w(xz) = w(uz) + w(vz)$. The equivalence of the editing costs in $G$ and $H$ in this case is immediate.

Finally, the running time is polynomial because the minimum cut of two vertices of a graph can be found in polynomial time [17], and $H$ can clearly be constructed in polynomial time as well. ∎

**Theorem 3.6** *If Rules 3.1.1, 3.1.2 and 3.1.3 do not apply, and we have not answered* NO *yet, then either the current graph has at most $k^2 + 4k + r$ vertices, or the answer is* NO.

**Proof.** Let us consider a fuzzy graph $H = (V, E, F)$ obtained from a fuzzy input graph $G$ by applying Rules 3.1.1, 3.1.2 and 3.1.3 until it is possible, such that we did not answer NO in the process. Let $k$ and $r$ the parameters associated to $H$, and $k'$ and $r'$ the parameter associated to $H$. We first prove that if $H$ is a YES instance to the fuzzy cluster editing problem, then it has at most $k'^2 + 3k' + r'$ vertices.

If Rule 3.1.1 and Rule 3.1.2 do not apply, it means that $H$ has at most $k'$ connected components, and each of them must be edited. If Rule 3.1.3 does not apply, then there cannot be cliques of size greater than $k'+1$. Hence $H$ has no cliques of size greater than $k'+1$ and we assume that there exists a set $M \subseteq V \times V$ such that $H' = (V, (E \setminus M) \cup (M \setminus E))$ is a cluster graph and $\sum_{uv \in M} w(uv) \leq k'$. First we consider the case when $H$ is connected. Let us define the following:

- $S \subseteq V$ is the set of vertices that are incident to some real edge or non-edge in $M$ (the edited edges);
- $R$ is a minimum fuzzy vertex cover of $H$, so that $r' = |R|$;
- $X = V \setminus R$, so that $H[X]$ does not contain any fuzzy edge;
- $X' = S \cap X$.

Then it is easy to see that $|X'| \leq |S| \leq 2k'$. Let us focus on the graph $H[X \setminus X']$. It does not contain fuzzy edges, and none of its vertices is incident to an edited real edge or non-edge. We can conclude that it must be a union of disjoint cliques. In particular we show that it must be the union of at most $k'+1$ disjoint cliques, and that each of them has specific neighbors in

the rest of the graph. Since no vertex of these cliques is incident to an edited edge, and there are no fuzzy edges in between them, each of the cliques must belong to a different cluster in the solution. However, to create more than $k'+1$ connected components from a connected graph, we need to remove at least $k'+1$ edges. Hence the first part of the claim is proved.

¿From the previous argument, it also follows that all vertices in $X'$ connected to a clique in $H[X \setminus X']$, must end up in the same cluster of the solution as the clique they are adjacent to. Assume they do not, then we should delete an edge incident to a vertex in $H[X \setminus X']$, getting a contradiction. Therefore every vertex in $X'$ can be connected to at most one clique in $H[X \setminus X']$, and furthermore it must be adjacent to all vertices of this clique. This means that every clique in $H[X \setminus X']$ has either some neighbors in $X'$ and size at most $k'$, or it has neighbors only in $R$ and size at most $k'+1$. Going back to $H$, if we define $N$ as the number of cliques in $H[X \setminus X']$ that have neighbors only in $R$, we have the following bound:

$$|V| \leq (k'+1) \cdot N + k' \cdot (k'+1-N) + (2k'-N) + |R| = k'^2 + 3k' + r'.$$

The first two terms give a bound on the number of vertices in $H[X \setminus X']$ according to the previous discussion, while the term $(2k'-N)$ represents a tighter bound on $|X'|$. In fact, for every clique with neighbors only in $R$, there must be at least one distinct vertex in $R$ incident to an edited edge. This is because we need to disconnect the clique from the rest of the graph, but we cannot touch edges incident to its vertices. Besides at least one endpoint of the edge we have to remove will belong to the same cluster as the clique.

Consider now the case when $H$ has $l$ connected components $C_1, \ldots, C_l$, where $1 < l \leq k'$. By Observation 3.2 we know that there is a solution for $H$ that edits at most $k'$ edges if and only if there is a solution for each $H[C_i]$ that edits at most $k'_i$ edges, such that $\sum_{i=1}^{l} k'_i \leq k'$. This means that, by what we just proved for connected fuzzy graphs, if there is a solution for $H$ then $|V(H[C_i])| \leq k_i'^2 + 3k'_i + r'_i$ for $1 \leq i \leq l$, where $r'_i$ is the size of a minimum vertex cover of $H[C_i]$. Hence $|V| \leq \sum_{i=1}^{l} k_i'^2 + 3k'_i + r'_i$, that is

$$|V| \leq \sum_{i=1}^{l} k_i'^2 + 3\sum_{i=1}^{l} k'_i + \sum_{i=1}^{l} r'_i \leq (\sum_{i=1}^{l} k'_i)^2 + 3k' + r' = k'^2 + 3k' + r'.$$

To complete the proof, we simply show that $k' \leq k$ and $r' \leq r+k$. That $k'$ is a most $k$ is straightforward from Rule 3.1.3, which is the only case when the parameter $k$ is modified, in particular decreased. As for the fuzzy vertex cover, when we apply Rules 3.1.1 and 3.1.2 we only remove vertices from $G$, therefore no fuzzy edges are introduced and the fuzzy vertex cover can only get smaller or stay the same. Applying Rule 3.1.3, instead, might create new fuzzy edges incident only to the new vertex that is created by merging two vertices in $G$. This means that all new fuzzy edges can be covered by adding such new vertex to the fuzzy vertex cover. Besides, when a fuzzy edge is created, the parameter $k$ is always decreased by at least one (see Rule 3.1.3 case 3 $(c)$). Hence a vertex will be added to the fuzzy vertex cover of the reduced graph, no more than $k$ times. Concluding we have that $r' \leq r+k$ as claimed, so that if the graph $H$ is YES instance, then it has at most $k'^2 + 3k' + r' \leq k^2 + 4k + r$ vertices, concluding the proof. ∎

It is easy to construct examples where we have $(k+1) \cdot N + k \cdot (k+1-N) + (k+1-N) + r = k^2 + 2k + 1 + r$ vertices in a *yes* instance for any given $k$; see Figure 3 for an example. Hence Theorem 3.6 gives a quite tight bound on the size of the kernel, that is in any case $O(k^2 + r)$.

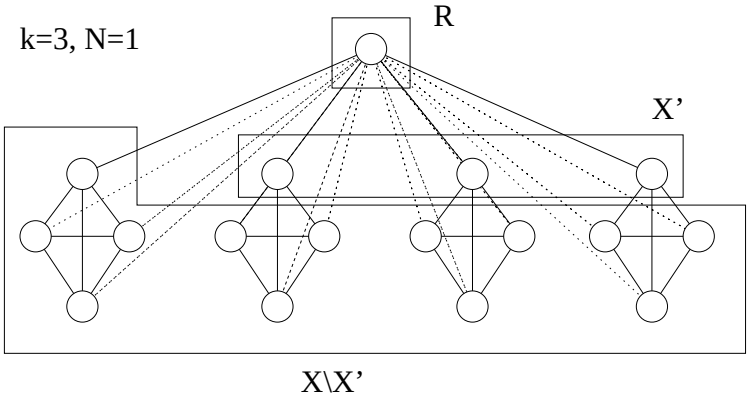


Figure 3: Example of a kernel with $k^2 + 2k + 1 + r$ vertices for $k = 3$. The non-edges are not drawn to keep the figure easier to follow.

We have thus proved that the $(k, r)$-WFCE problem has a kernel of size $O(k^2 + r)$, where it is then possible to find a solution just by trying all possible editings of its edges. Given this, and that all the reduction rules can clearly be applied only a polynomial number of times, and all run in polynomial time by Lemmas 3.3, 3.4 and 3.5, we can state the final result of this section.

**Theorem 3.7** *The* $(k, r)$-Weighted Fuzzy Cluster Editing *problem can be solved in time* $n^{O(1)} + O((k^2 + r)^{2k})$.

# 4 Parameterized edge clique partition with partial information

In this section, we study the problem of partitioning the edges of a fuzzy graph $G = (V, E, F)$ into edge-disjoint cliques. In this problem, no editing of the edges or non-edges of $G$ is involved, but we have to decide for each fuzzy edge whether or not it should become a real edge or a non-edge. Below is a formal definition of the problem.

Fuzzy Edge Clique Partitioning (FECP)
*Instance:* A fuzzy graph $G = (V, E, F)$ and an integer $k \geq 0$.
*Question:* Is there a realization $(R^+, R^-)$ of the fuzzy edges such that the edges of $G' = (V, E \cup R^+)$ can be partitioned into at most $k$ edge-disjoint cliques?

Naturally, being a more general version of the problem on non-fuzzy graphs, the Fuzzy Edge Clique Partitioning problem is NP-hard as well. Interestingly, we show that it remains NP-hard also when $k$ is a fixed constant and not a part of the input, for every $k \geq 3$. Recall that, in contrast, the Edge Clique Partitioning problem is FPT when parameterized by $k$. We show the FECP problem parameterized by both $k$ and $r$, where $r$ is again the size of a minimum fuzzy vertex cover, is FPT. We call this version of the problem $(k, r)$-FECP.

### 4.1 The $k$-Fuzzy Edge Clique Partitioning problem is NP-complete

Here we prove that deciding whether the edges of a fuzzy graph can be partitioned into at most $k$ edge-disjoint cliques, is NP-complete for every fixed $k \geq 3$, and polynomial otherwise. The problem we reduce from, is the classical $k$-Coloring problem. In this problem, the input is a graph $G = (V, E)$ and a fixed $k > 0$, and the question is whether the vertices of $G$ can be colored with at most $k$ colors, such that no two adjacent vertices have the same color. It is well known that this problem is NP-complete for every fixed $k \geq 3$.

**Theorem 4.1** *The $k$-Fuzzy Edge Clique Partitioning problem is NP-complete for fixed $k \geq 3$.*

**Proof.** Given a graph $G = (V, E)$ we build a fuzzy graph $G' = (V', E', F)$ as follows. For each vertex $v_i \in V$, create a new vertex $u_i$ and call $U$ the set of such vertices, so that $V' = V \cup U$. Now the only real edges in $E'$ are the edges $v_i u_i$, because we replace each $v_i v_j \in E$ with a non-edge, and for every other pair of vertices we add a fuzzy edge. See Figure 4 for an example.

Now we claim that $G = (V, E)$ can be colored with at most $k$ colors if and only if there exists a normalization of $G' = (V \cup U, E', F)$ whose edges can be partitioned into at most $k$ edge-disjoint cliques.

Assume there is a coloring of $G = (V, E)$ that uses only $k$ colors, so that no two adjacent vertices have the same color. This is equivalent to partitioning $V$ into $k$ sets $A_1, \cdots, A_k$, such that $A_j$ is an independent set for each $1 \leq j \leq k$. Let us partition also $U$ into $k$ sets $A'_1, \cdots, A'_k$ such that $u_i \in A'_j$ if and only if $v_i \in A_j$. Then, by construction and the fact that $G[A_j]$ contains only non-edges, we know that $G'[A_j \cup A'_j]$ contains only fuzzy edges and real edges for each $1 \leq j \leq k$. Hence, if we realize all fuzzy edges inside each $G'[A_j \cup A'_j]$ into real edges, and all the remaining fuzzy edges into non-edges, we get a non-fuzzy graph consisting of $k$ vertex-disjoint cliques, with no edges between them. Hence, we have a natural solution for the $k$-Edge Clique Partitioning problem on $G'$, since vertex-disjoint cliques are also edge-disjoint and their union contains all edges.

Assume now that there exists a normalization $H$ of $G'$, that admits a partition $K = \{K_1, \ldots, K_l\}$ of its edges, so that $H(K_i)$ is a clique for each $1 \leq i \leq l$ and $l \leq k$. By construction we know that $G'$ is not edgeless, so that $l \geq 1$. Let us now build a solution for the $k$-Coloring problem on $G$ using $K$. We associate a different color $c_i$ with each element $K_i \in K$, and if $v_j u_j \in K_i$ then we assign color $c_i$ to a vertex $v_j \in V$. We claim that this gives a legal $k$-coloring. First of all, there are at most $k$ cliques in $K$, so at most $k$ colors are used. Besides the color of a vertex in $G$ is well defined because $K$ is a partitioning. Assume, by contradiction, that there is at least one edge $v_i v_j \in E$ such that $v_i$ and $v_j$ have the same color. This implies that $v_i u_i$ and $v_j u_j$ belong to the same clique of $K$. However, by construction, there should be a non-edge between $v_i$ and $v_j$ in $G'$. Hence there cannot be any normalization of $G'$ where $v_i u_i$ and $v_j u_j$ belong to the same clique, and therefore to the same element of a valid edge clique partition. This proves our claim.

Clearly, $G'$ can be constructed from $G$ in polynomial time; hence the theorem follows. ∎

As a complementary result, we prove next that the $k$-FECP problem is solvable in polynomial time when $k$ is 0, 1, or 2. For the following result, we define an *isolated vertex of a fuzzy graph* to be a vertex that is not incident to any real edge. Similarly, we define a *universal vertex of a fuzzy graph* to be a vertex that is not incident to any non-edge.

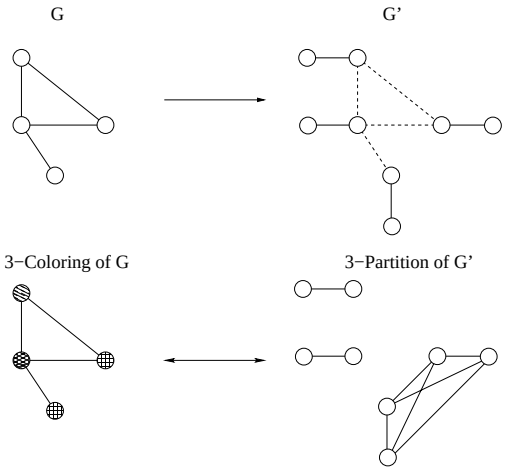


Figure 4: An example of the graph $G'$ obtained from a graph $G$, and the equivalence of a solution of the 3-Coloring problem on $G$ with a solution of the 3-Edge Clique Partitioning problem on $G'$. In $G'$ the fuzzy edges are not drawn to keep the figure clean, while in the corresponding clique partition the non edges are not drawn.

**Theorem 4.2** *The k-FECP problem can be solved in polynomial time for $k \leq 2$.*

**Proof.** The 0-FECP problem is solvable if and only if there is a normalization of the input fuzzy graph that is edgeless. In other words if the input graph has no real edges.

The 1-FECP problem is solvable if and only if there are no non-edges in the input graph after we removed all isolated vertices, i.e., the input graph can be realized into a clique plus some isolated vertices.

In each case the conditions can clearly be checked in polynomial time.

For the 2-FECP problem we can assume that there are no isolated vertices in the input graph, as they do not affect the solvability of the problem, and that no solution exists for $k \leq 1$. In this case, if there exists a solution, then it consists of exactly two cliques, that are either disjoint or intersect in exactly one vertex. In both cases it is easy to see that the complement of a solution is always bipartite, in particular a complete bipartite graph, or a complete bipartite graph plus an isolated vertex. Hence a necessary condition for the input graph to be a YES-instance, is that the graph induced by its non-edges is bipartite. This condition alone, however, is clearly not sufficient, and that is why we distinguish two cases.

If all connected components of the input graph can be realized into cliques, i.e., do not contain non-edges, we create a graph that has a vertex for each connected component and an edge between two vertices only if there is a non-edge between the corresponding connected components. If this graph is bipartite, then we clearly have a feasible solution, since each bipartition can be realized into a clique.

If the above graph is not bipartite, or if at most one connected component of the input graph contains non-edges, then we know that if a solution exists, it must consists of two cliques intersecting in one vertex. Hence, we simply try and remove one universal vertex, and check whether the graph without this vertex has a normalization consisting of two disjoint cliques, with the technique explained above. If the answer is NO for each universal vertex, then there is no solution, otherwise we can clearly add back the vertex making it universal to the two

cliques we found, yielding a valid solution.

This approach can clearly be implemented to run in polynomial time, since it essentially checks whether a graph is bipartite at most $n+1$ times. ∎

## 4.2 The $(k, r)$-Fuzzy Edge Clique Partitioning problem is FPT

To obtain a kernel for $(k, r)$-FECP, we first give some observations that apply to any valid solution of the problem on non-fuzzy graphs, i.e., the $k$-Edge Clique Partitioning problem.

For a non-fuzzy graph $G = (V, E)$, and a fixed $k \geq 0$, we call a *feasible solution* a partition $K = \{K_1, K_2, ..., K_l\}$ of $E$ such that $G(K_i)$ is a clique for each $i$, and $l \leq k$. For $K_i \in K$, we define $V(K_i)$ as the union of the endpoints of the edges in $K_i$, i.e. $V(G[K_i])$. We call *gateways* the vertices that are in the intersection of some cliques defined by elements of $K$, while the vertices contained only in one clique are called *normal*. Two normal vertices in the same clique are said to be *co-normal*. We define a set $V' \subseteq V$ to be a *type* if there is at least one vertex $v$ such that $N[v] = V'$. So we say that two vertices $u$ and $v$ are of the *same type* if $N[u] = N[v]$, and that they are of *different type* otherwise. Finally notice that the intersection of two cliques in any solution cannot consist of more than one vertex, or there would be one edge covered by two cliques.

**Theorem 4.3 ([7])** *Every edge clique partition of a complete graph on $n$ vertices, except the trivial one of a single clique, contains at least $n$ cliques.*

**Lemma 4.4** *If the answer to the $k$-Edge Clique Partitioning problem for a graph $G = (V, E)$ is* yes*, then the answer is* yes *also for each induced subgraph of $G$.*

**Proof.** Let $K = \{K_1, \ldots, K_l\}$ be a partition of $E$, such that $l \leq k$ and $G(K_i)$ is a clique for each $1 \leq i \leq l$. Then consider $V' \subset V$, and $G' = G[V \setminus V']$. Clearly, all vertices of $G(K_i)$ which are also in $G'$, can also be covered by one clique whose edges will not belong to any other clique. Therefore it is possible to map each clique defined by $K$ to some, possibly empty, clique in $G'$, yielding a feasible solution for $G'$. ∎

Lemma 4.4 implies that if there is even only one induced subgraph of $G$ for which the answer is no, then $G$ itself is a no instance. We will use this observation often.

**Observation 4.5** *In any solution of $k$-Edge Clique Partitioning, there cannot be more than $\binom{k}{2}$ gateway vertices.*

**Proof.** Every two cliques defined by a feasible solution can intersect in at most one vertex, or they would cover the same edge. Since there are at most $\binom{k}{2}$ possible intersections among $k$ cliques, the result follows. ∎

In order to show that this upper bound is tight, we provide a way to construct a graph $G$ with $k(k-1)/2$ gateways, for any $k$. Let $K = \{K_1, K_2, \ldots, K_k\}$ be a partition of the edges of a graph $G$ into $k$ cliques, and let $|V(K_i)| = k$ for each $1 \leq i \leq k$. Now let $v_i^j$ be the vertex $i$ of $G(K_j)$. Then for each $1 \leq i, j \leq k$, we set $v_i^j = v_j^i$ and complete the construction of $G$. In Figure 5 we give an example for $k = 6$. It is easy to see that every two cliques intersect in exactly one vertex, so that there is no edge covered by more than one clique. Furthermore, for every pair of cliques, we have a different intersection implying a tight bound. This also means that all gateways have different types.

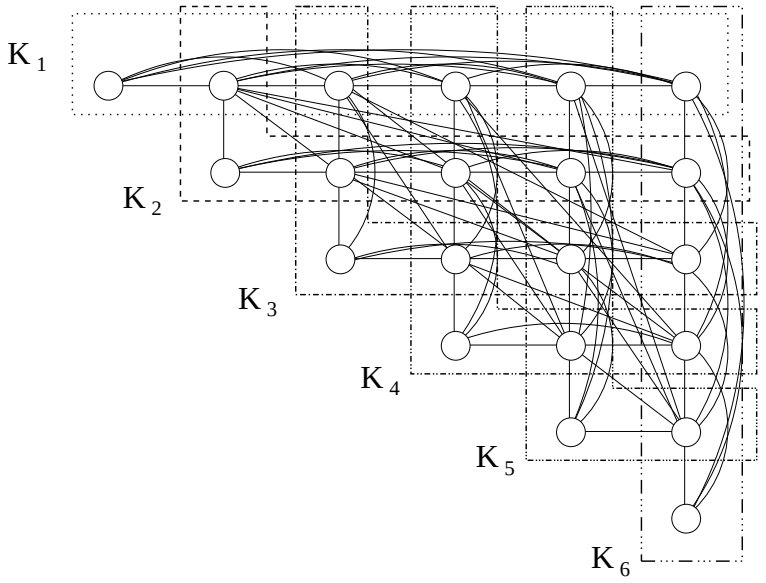


Figure 5: Example of a graph with $\binom{k}{2}$ gateway vertices. Every box represents a clique, and every two cliques intersect in exactly one distinct vertex.

**Observation 4.6** *Given a feasible solution, all the vertices of the same type either belong to the same clique and are co-normal, or they are all gateways.*

**Proof.** Take a normal vertex $v$ in the solution. It belongs to only one clique $K$ by definition, which is therefore also its type. Hence, all other vertices of the same type of $v$ must be only in $K$ as well and be co-normal with $v$.

If $v$ is not normal, then it must be a gateway, and by what we have just proved, no other vertex of the same type of $v$ can be normal either, or $v$ would be as well giving a contradiction. ∎

**Observation 4.7** *If there are more than $k + \binom{k}{2}$ vertices of pairwise different type, then the answer to $k$-*Edge Clique Partitioning *is* no.

**Proof.** By Observation 4.6, we know that to each type must correspond to either a clique in a solution, or a set of gateways. Since a solution cannot have more than $k$ cliques and by Observation 4.5 there can be at most $\binom{k}{2}$ gateway vertices, we can conclude that there cannot be more than $k + \binom{k}{2}$ different types of vertices in a yes-instance. ∎

Figure 5 shows that also the bound in Rule 4.7 is tight.

It is now enough to give a simple generalization of the previous observations to get a polynomial time kernelization for the $(k, r)$-FECP problem. ¿From now on we assume a fuzzy input graph $G = (V, E, F)$.

First we need to introduce a generalization of the type of a vertex for fuzzy graphs. The *fuzzy neighborhood* of a vertex $v$ is the set of the vertices $w$ such that $vw \in F$. We say that two vertices are of the same *absolute type* if their closed and fuzzy neighborhoods are equal.

Consider a fuzzy graph $G = (V, E, F)$, and let $S \subset V$ be a minimum fuzzy vertex cover of $G$, such that $|S| \leq r$. Then for each vertex in $X = V \setminus S$, there can be at most $3^r$ possible

ways to have adjacencies in $S$. So we can classify the vertices of $X$ into $3^r$ categories, so that the vertices in the same category have the same absolute type with respect to the vertices in $S$. Since $G[X]$ is a non-fuzzy graph, if there is no solution to $k$-Edge Clique Partitioning for $G[X]$, then there is no solution to $(k,r)$-FECP on $G$ no matter how we realize the fuzzy edges, due to Lemma 4.4.

**Rule 4.2.1** *If there are more than* $(k + \binom{k}{2}) \cdot 3^r$ *vertices with different absolute type in* $X$*, then the answer is* NO.

**Lemma 4.8** *Rule 4.2.1 is correct and can be executed in polynomial time.*

**Proof.** If there are more then $(k + \binom{k}{2}) \cdot 3^r$ absolute types of vertices, then $G[X]$ must have more than $(k + \binom{k}{2})$ vertices of different types. Hence by Observation 4.7, there is no solution for $G[X]$. By Lemma 4.4, this implies that there is no solution for $G$ as well, proving the first part of the statement.

The rule can be easily executed in polynomial time by listing the absolute closed neighborhoods of the vertices of $G$, and checking whether there are more than $(k+\binom{k}{2})\cdot 3^r$ different ones. Since $k$ and $r$ are constants, the result follows. ∎

**Rule 4.2.2** *If Rule 4.2.1 does not apply and there are more than* $\binom{k}{2} + 1$ *vertices of the same absolute type in* $X$*, then remove one.*

**Lemma 4.9** *Rule 4.2.2 is correct and can be executed in polynomial time.*

**Proof.** Let $u$ be the vertex we remove. Then we show that there is a solution for $G$ if and only if there is a solution for $G - u$. Assume that there is a normalization $H$ of $G - u$ that admits a feasible solution $K'$. Since in $G - u$ there are at least $\binom{k}{2} + 1$ vertices of the same absolute type as $u$, we know that at least one of them, let us say $v$, is a normal vertex for exactly one clique induced by some set of $K'$. By Observation 4.5 there cannot be more than $\binom{k}{2}$ gateways in any solution. This means that if we realize the fuzzy edges of $G[V \setminus \{u\}]$ as in $H$ and the fuzzy edges incident to $u$ as the fuzzy edges incident to $v$ in $G - u$, we get a normalization $H'$ of $G$ where $u$ and $v$ have the same type. Hence a feasible solution can be obtained from $K'$ by just adding all edges incident to $u$ to the same set of $K'$ containing all edges incident to $v$.

On the other hand, by Lemma 4.4, if there is a normalization $G'$ of $G$ with a feasible solution, then there is also a normalization of $G - u$ that has a feasible solution, namely $G' - u$. ∎

**Lemma 4.10** *If Rules 4.2.1 and 4.2.2 do not apply, then the graph has at most* $(\binom{k}{2} + 1) \cdot ((k + \binom{k}{2}) \cdot 3^r) + r$ *vertices.*

**Proof.** It follows directly by the fact that Rule 4.2.1 and Rule 4.2.2 do not apply. ∎

**Theorem 4.11** $(k,r)$-Fuzzy Edge Clique Partitioning *is FPT with a kernel of size* $O(k^4 \cdot 3^r)$.

**Proof.** The size of the kernel follows from Lemma 4.10, so we only need to show that the preprocessing can be performed in polynomial time. Rules 4.2.1 and 4.2.2 can be applied in polynomial time by Lemma 4.8 and 4.9. Besides they are applied at most a polynomial number of times. In fact, every time we apply Rule 4.2.2 we either remove one vertex or stop, and Rule 4.2.1 needs to be applied only once before every application of Rule 4.2.2. In total we can have at most $2n$ application of the rules, hence the theorem follows. ▪

# 5 Concluding remarks

In this paper we have studied the parameterized complexity of two important examples of *graph clustering problems* on inputs consisting of fuzzy graphs: graphs that represent incomplete information about relationships. We believe that the investigation of "problems on fuzzy graphs" is extremely well-motivated by applications, particularly in areas such as machine learning and bioinformatics, where complete information about the graphs modeling various computational objectives is often not available. In this general context, much more remains to be done.

We have described two FPT algorithms: one for the Weighted Fuzzy Cluster Editing problem, and another for the Fuzzy Edge Clique Partitioning problem. Both are parameterized by the compound parameter $(k, r)$ where $k$ is a *cost parameter* representing:

- the total cost of the editing in the case of Weighted Fuzzy Cluster Editing, and
- the number of cliques in the partition in the case of Fuzzy Edge Clique Partitioning.

The parameter $r$ is a *structural parameter* and in both problems it represents the minimum number of vertices required to cover the undecided edges of the fuzzy graph taken as input. This structural parameter could be well-motivated by applications where only a small number of "trouble-maker" vertices are the "cause" of the uncertain information about the input.

We have also shown that in the case of Fuzzy Edge Clique Partitioning, it is not possible to extend the above positive outcome to a parameterization only by $k$.

In the case of the Fuzzy Cluster Editing problem, the analogous question remains open, and this is in fact a prominent concrete open problem in parameterized complexity. Apart from the important machine learning applications noted in [2, 3], it has recently been shown that for the special case where all weights are 1, the Fuzzy Cluster Editing problem (parameterized only by $k$) is FPT-equivalent [14], to the Minimum Terminal Edge Separation problem left open by Marx in [24].

Another area of open problems concerning this work is that of *improving kernelization bounds*. Because FPT kernelization is of great practical significance due to the general connection to efficient pre-processing (see [15, 21, 26] for background and discussion of this point), it is an outstanding open problem as to whether Fuzzy Edge Clique Partitioning admits a $Poly(k, r)$ kernelization.


# Acknowledgments

The authors would like to thank Daniel Lokshtanov for drawing our attention to the bound given in Theorem 3.6.

# References


[1] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5), 2008.

[2] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *Proceedings of FOCS 2002 - 43rd Symposium on Foundations of Computer Science*, pages 238–247, 2002.

[3] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[4] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999.

[5] S. Böcker, S. Briesemeister, Q.B.A. Bui, and A. Truss. Going weighted: Parameterized algorithms for cluster editing. *Theor. Comput. Sci.*, In press, 2009.

[6] Hans L. Bodlaender, Michael R. Fellows, Pinar Heggernes, Federico Mancini, Charis Papadopoulos, and Frances A. Rosamond. Clustering with partial information. In *Proceedings of MFCS 2008 - 33rd International Symposium on Mathematical Foundations of Computer Science*, volume 5162, pages 144–155. Springer, 2008.

[7] N.J. De Bruijin and P. Erdos. On a combinatorial problem. *Ind. Math.*, 10:421–423, 1948.

[8] L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Inf. Process. Lett.*, 58(4):171–176, 1996.

[9] M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005.

[10] Z-Z. Chen, T. Jiang, and G. Lin. Computing phylogenetic roots with bounded degrees and errors. *SIAM J. Comput.*, 32(4):864–879, 2003.

[11] E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci.*, 361(2-3):172–187, 2006.

[12] E. D. Demaine and N. Immorlica. Correlation clustering with partial information. In *Proceedings of RANDOM-APPROX 2003 - 7th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 1–13, 2003.

[13] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[14] D. Emanuel and A. Fiat. Correlation clustering - minimizing disagreements on arbitrary weighted graphs. In *Proceedings of ESA 2003 - 11th Annual European Symposium on Algorithms*, pages 208–220, 2003.

[15] M. R. Fellows. The lost continent of polynomial time: Preprocessing and kernelization. In *Proceedings of IWPEC 2006 - Parameterized and Exact Computation, Second International Workshop*, pages 276–277, 2006.

[16] M. R. Fellows, M. A. Langston, F. A. Rosamond, and P. Shaw. Efficient parameterized preprocessing for cluster editing. In *Proceedings of FCT 2007 - Fundamentals of Computation Theory, 16th International Symposium*, pages 312–321, 2007.

[17] A. V. Goldberg. Recent developments in maximum flow algorithms (invited lecture). In *Proceedings of SWAT '98 - 6th Scandinavian Workshop on Algorithm Theory*, volume 1432, pages 1–10, 1998.

[18] M. C. Golumbic, H. Kaplan, and R. Shamir. Graph sandwich problems. *J. Algorithms*, 19(3):449–473, 1995.

[19] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. Graph-modeled data clustering: Exact algorithms for clique generation. *Theory Comput. Syst.*, 38(4):373–392, 2005.

[20] J. Guo. A more effective linear kernelization for cluster editing. In *Proceedings of ESCAPE 2007 - Combinatorics, Algorithms, Probabilistic and Experimental Methodologies, First International Symposium*, pages 36–47, 2007.

[21] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.

[22] M. Krivánek and J. Morávek. *NP* -hard problems in hierarchical-tree clustering. *Acta Inf.*, 23(3):311–323, 1986.

[23] S. H. Ma, W. D. Wallis, and J. L. Wu. The complexity of the clique partition number problem. *Congr. Numer.*, 67:59–66, 1988.

[24] D. Marx. Parameterized graph separation problems. In *Proceedings of IWPEC'04 - 1st International Workshop on Parameterized and Exact Computation*, pages 71–82. Springer, 2004.

[25] E. Mujuni and F. A. Rosamond. Parameterized complexity of the clique partition problem. In *Proceedings of ACiD 2007 - 2nd Workshop of Algorithms and Complexity in Durham*, 2007.

[26] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[27] J. Orlin. Contentment in graph theory: Covering graphs with cliques. *Indagationes Math.*, 39:406–424, 1977.

[28] F. Protti, M. D. da Silva, and J. L. Szwarcfiter. Applying modular decomposition to parameterized bicluster editing. In *Proceedings of IWPEC 2006 - Parameterized and Exact Computation, Second International Workshop*, pages 1–12, 2006.

[29] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182, 2004.