# Clustering to Given Connectivities

**Petr A. Golovach**
Department of Informatics, University of Bergen, Norway
petr.golovach@uib.no

**Dimitrios M. Thilikos**
AlGCo project-team, LIRMM, Université de Montpellier, CNRS, Montpellier, France
sedthilk@thilikos.info

## Abstract

We define a general variant of the graph clustering problem where the criterion of density for the clusters is (high) connectivity. In Clustering to Given Connectivities, we are given an $n$-vertex graph $G$, an integer $k$, and a sequence $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$ of positive integers and we ask whether it is possible to remove at most $k$ edges from $G$ such that the resulting connected components are *exactly* $t$ and their corresponding edge connectivities are lower-bounded by the numbers in $\Lambda$. We prove that this problem, parameterized by $k$, is fixed parameter tractable, i.e., can be solved by an $f(k) \cdot n^{O(1)}$-step algorithm, for some function $f$ that depends only on the parameter $k$. Our algorithm uses the recursive understanding technique that is especially adapted so to deal with the fact that we do not impose any restriction to the connectivity demands in $\Lambda$.

**2012 ACM Subject Classification** Mathematics of computing → Graph theory

**Keywords and phrases** graph clustering, edge connectivity, parameterized complexity

**Digital Object Identifier** 10.4230/LIPIcs.IPEC.2019.18

**Related Version** A full version of the paper is available at `https://arxiv.org/abs/1803.09483`.

**Funding** *Petr A. Golovach*: Supported by the Research Council of Norway via the projects CLASSIS and MULTIVAL. Supported by the Research Council of Norway and the French Ministry of Europe and Foreign Affairs, via the Franco-Norwegian project PHC AURORA 2019.
*Dimitrios M. Thilikos*: Supported by projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010). Supported by the Research Council of Norway and the French Ministry of Europe and Foreign Affairs, via the Franco-Norwegian project PHC AURORA 2019.

## 1 Introduction

Clustering deals with grouping the elements of a data set based on some similarity measure between them. As a general computational procedure, clustering is fundamental in several scientific fields including machine learning, information retrieval, bioinformatics, data compression, and pattern recognition (see [7, 68, 70]). In many such applications, data sets are organized and/or represented by graphs that naturally express relations between entities. A *graph clustering problem* asks for a partition of the vertices of a graph into vertex sets, called *clusters*, so that each cluster enjoys some desirable characteristics of "density" or "good interconnectivity", while having few edges between the clusters (see [12, 63] for related surveys).

**Parameterizations of graph clustering problems.** As a general problem on graphs, graph clustering has many variants. Most of them depend on the *density* criterion that is imposed on the clusters and, in most of the cases, they are NP-complete. However, in many real-world instances, one may expect that the number of edges between clusters is much smaller than the size of the graph. This initiated the research for parameterized algorithms for graph clustering problems. Here the aim is to investigate when the problem is FPT (Fixed Parameter


Tractable), when parameterized by the number $k$ of edges between clusters, i.e. it admits a $f(k) \cdot n^{O(1)}$ step algorithm, also called FPT-algorithm (see [21, 27, 30, 59] for textbooks on parameterized algorithms and the corresponding parameterized complexity class hierarchy). More general parameterizations may also involve $k$ edit operations to the desired cluster property.

In the most strict sense, one may demand that all vertices in a cluster are pair-wise connected, i.e., they form a clique. This corresponds to the Cluster Deletion problem and its more general version Cluster Editing where we ask for the minimum edge additions or deletions that can transform a graph to a collection of cliques. Cluster Editing was introduced by Ben-Dor, Shamir, and Yakhini in [6] in the context of computational biology and, independently, by Bansal, Blum, and Chawla [5] motivated by machine learning problems related to document clustering (see also [65]). Algorithmic research on these problems and their variants is extensive, see [1–3, 28, 35, 65]. Moreover their standard parameterizations are FPT and there is a long list of improvements on the running times of the corresponding FPT-algorithms [10, 11, 13, 14, 17, 18, 29, 39, 41, 62].

In most practical cases, in a good clustering, it is not necessary that clusters induce cliques. This gives rise to several difference measures of density or connectivity. In this direction, Heggernes et al., in [46], introduced the $(p, q)$-Cluster Graph Recognition problem where clusters are cliques that may miss at most $p$ edges (also called $\gamma$-quasi cliques) [60, 61]). This problem was generalized in [56], where, given a function $\mu$ and a parameter $p$, each cluster $C$ should satisfy $\mu(C) \leq p$, and was proved to be FTP for several instantiations of $\mu$. In [47], Hüffner et al. introduced the Highly Connected Deletion problem, where each cluster $C$ should induce a highly connected graph (i.e., have edge connectivity bigger than $|C|/2$ – see also [45,48]) and proved that this problem is FPT. Algorithmic improvements and variants of this problem where recently studied by Bliznets and Karpov in [9]. In [42], Guo et al. studied the problems $s$-Defective Clique Editing, Average-$s$-Plex Delection, and $\mu$-Clique Deletion where each cluster $S$ is demanded to be a clique missing $s$ edges, a graph of average degree at least $|C| - s$, or a graph with average density $s$, respectively (the two first variants are FPT, while this is not expected for the last one). In [64] clusters are of diameter at most $s$ ($s$-clubs), in [4, 43, 58, 67] every vertex of a cluster should have an edge to all but at most $s - 1$ other vertices of it ($s$-plexes). In [32], Fomin et al. considered the case where the number of clusters to be obtained is *exactly* $p$ and proved that this version is also in FPT. Other Parameterizations of Cluster Editing where introduced and shown to be FPT in [8, 15, 25, 53, 69]).

**Our results.** Here we adopt connectivity as a general density criterion for the clusters (following the line of [9, 48]). We study a general variant of graph clustering where we prespecify both the number of clusters (as done in [32]) but also the connectivities of the graphs induced by them. Actually we consider the edge weighted version of the problem where the weighted edge connectivity $\lambda^{\mathsf{w}}(G)$ of an edge weighted graph $G$ is defined as the minimum weight of an edge cut (see Section 2 for formal definitions).

Clustering to Given Weighted Connectivities (CGWC)

*Input:* A weighted graph $G$ with an edge weight function $w\colon E(G) \to \mathbb{N}$, a $t$-tuple $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$, where $\lambda_i \in \mathbb{N} \cup \{+\infty\}$ for $i \in \{1, \ldots, t\}$ and $\lambda_1 \leq \ldots \leq \lambda_t$, and a nonnegative integer $k$.

*Task:* Decide whether there is a set $F \subseteq E(G)$ with $\mathsf{w}(F) \leq k$ such that $G - F$ has $t$ connected components $G_1, \ldots, G_t$ where each $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ for $i \in \{1, \ldots, t\}$.

(It is convenient to allow $\lambda_i = +\infty$, because we assume that the (weighted) connectivity of the single-vertex graph is $+\infty$. )

The above problem can be seen as a generalization of the well-known $t$-Cut problem, asking for a partition of a graph into exactly $t$ nonempty components such that the total number of edges between the components is at most $k$. Indeed, this problem is CGWC for unit weights and $\lambda_1 = \ldots = \lambda_t = 1$. As it was observed by Goldschmidt and Hochbaum in [36], $t$-Cut is NP-hard if $t$ is a part of the input. This immediately implies the NP-hardness of CGWC. Therefore, we are interested in the parameterized complexity of the problem. The main results of Goldschmidt and Hochbaum [36] is that $t$-Cut can be solved in time $\mathcal{O}(n^{t^2})$, that is, the problem is polynomial for any *fixed* $t$. In other words, $t$-Cut belongs in the parameterized complexity class XP when parameterized by $t$. This results was shown to be tight in the sense that we cannot expect an FPT algorithm for this problem unless the basic conjectures of the Parameterized Complexity theory fail by Downey et al. who proved in [26] that the problem is W[1]-hard when parameterized by $t$. The situation changes if we parameterize the problem by $k$. By the celebrated result of Kawarabayashi and Thorup [49], $t$-Cut is FPT when parameterized by $k$.

In this paper, we prove that CGWC is FPT when parameterized by $k$. For our proofs we follow the *recursive understanding* technique introduced by Chitnis et al. [19] (see also [40]) combined with the *random separation* technique introduced by Cai, Chan and Chan in [16]. Already in [19], Chitnis et al. demonstrated that this technique is a powerful tool for the design of FPT-algorithms for various cut problems. This technique was further developed by Cygan et al. in [23] for proving that the Minimum Bisection problem is FPT (see also [33, 37, 52, 57] for other recent applications of this technique on the design of FPT-algorithms). Nevertheless, we stress that for CGWC the application for the recursive understanding technique becomes quite nonstandard and demands additional work due to the fact that neither $t$ nor the connectivity constraints $\lambda_1, \ldots, \lambda_t$ are restricted by any constant or any function of the parameter $k$ (we stress that the general meta-algorithmic framework of [57] is not directly applicable to our problem and in the conclusion section (Section 5) we provide some discussion on how to tackle this). Towards dealing with the diverse connectivities, we deal with special annotated/weighted versions of the problem and introduce adequate connectivity mimicking encodings in order to make recursive understanding possible.

**Paper organization.** Due to space constraints, we only give high level descriptions of our results. In Section 2, we give definitions that are used throughout the paper. In Section 3, we sketch our algorithm for the basic case where the input graph is connected. For this, we introduce all concepts and results that support the applicability of the recursive understanding technique. We stress that, at this point, the connectivity assumption is important as this makes it easier to control the diverse connectivities of the clusters. In Section 4, we briefly explain how we deal with the general non-connected case. The algorithm in the general case is based on a series of observations on the way connectivities are distributed in the connected components of $G$. In Section 5, we briefly discuss alternative approaches for CGWC and further directions of research. The details and complete proofs could be found in the full version of the paper [38].

## 2 Preliminaries

We consider finite undirected simple graphs. We use $n$ to denote the number of vertices and $m$ the number of edges of the considered graphs unless it creates confusion. For disjoint subsets $A, B \subseteq V(G)$, $E(A, B)$ denotes the set of edges with one end-vertex in $A$ and the second in $B$. A set of edges $S \subseteq E(G)$ of a connected graph $G$ is an *(edge) separator* if $G - S$

is disconnected. For two disjoint subsets $A, B \subseteq V(G)$, $S \subseteq E(G)$ is an $(A,B)$-separator if $G - S$ has no $(u,v)$-path with $u \in A$ and $v \in B$. Recall (see, e.g., [24]) that if $S$ is an inclusion minimal $(A,B)$-separator, then $S = E(A', B')$ for some partition $(A', B')$ of $V(G)$ with $A \subseteq A'$ and $B \subseteq B'$.

Let $k$ be a positive integer. A graph $G$ is *(edge) $k$-connected* if for every $S \subseteq E(G)$ with $|S| \leq k-1$, $G - S$ is connected, that is, $G$ has no separator of size at most $k-1$. Since we consider only edge connectivity, whenever we say that a graph $G$ is $k$-connected, we mean that $G$ is edge $k$-connected. Similarly, whenever we mention a separator, we mean an edge separator.

For technical reasons, it is convenient for us to work with edge weighted graphs. Let $G$ be a graph and let $w\colon E(G) \to \mathbb{N}$ be an (edge) weight function. Whenever we say that $G$ is a weighted graph, it is assumed that an edge weight function is given and we use $w$ to denote weights throughout the paper. For a set of edges $S$, $\mathsf{w}(S) = \sum_{e \in S} \mathsf{w}(e)$.

For disjoint subsets $A, B \subseteq V(G)$, $w_G(A,B) = \mathsf{w}(E(A,B))$. We say that $G$ is weight $k$-connected if for every $S \subseteq E(G)$ with $\mathsf{w}(S) \leq k-1$, $G - S$ is connected. We denote by $\lambda^{\mathsf{w}}(G)$ the *weighted connectivity* of $G$, that is, the maximum value of $k$ such that $G$ is weight $k$-connected; we assume that every graph is weight 0-connected and for the single-vertex graph $G$, $\lambda^{\mathsf{w}}(G) = +\infty$. For disjoint subsets $A, B \subseteq V(G)$, $\lambda^{\mathsf{w}}_G(A,B) = \min\{\mathsf{w}(S) \mid S \text{ is an } (A,B)\text{-separator}\}$. We say that an $(A,B)$-separator $S$ is *minimum* if $\mathsf{w}(S) = \lambda^{\mathsf{w}}_G(A,B)$. For two vertices $u, v \in V(G)$, $\lambda^{\mathsf{w}}(u,v) = \lambda^{\mathsf{w}}(\{u\},\{v\})$ and we assume that $\lambda^{\mathsf{w}}(u,u) = +\infty$. Similarly, for a set $A$ and a vertex $v$, we write $\lambda^{\mathsf{w}}_G(A,v)$ instead of $\lambda^{\mathsf{w}}_G(A,\{v\})$. Clearly, $\lambda^{\mathsf{w}}(G) = \min\{\lambda^{\mathsf{w}}_G(u,v) \mid u,v \in V(G)\}$. We can omit the subscript if it does not create confusion.

Let $U \subseteq V(G)$. We say that the weighted graph $G'$ is obtained from $G$ by the *weighted contraction* of $U$ if it is constructed as follows: we delete the vertices of $U$ and replace the set by a single vertex $u$ that is made adjacent to every $v \in V(G) \setminus U$ adjacent to a vertex of $U$ and the weight of $uv$ is defined as $\sum_{xv \in E(G),\ x \in U} \mathsf{w}(xv)$. Note that we do not require $G[U]$ be connected. For an edge $uv$, the weighted contraction of $uv$ is the weighted contraction of the set $\{u,v\}$.

## 3 Clustering to Given Weighted Connectivities for connected graphs

In this section we show that CGWC is FPT when parameterized by $k$ if the input graph is connected. We prove the following theorem that is used as the main building block for the general case.

▶ **Theorem 1.** *There exist some computable function $f : \mathbb{N} \to \mathbb{N}$, such that* CGWC *can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$ if the input graph is connected.*

The remaining part of the section contains the sketch of the proof of this theorem. In Subsection 3.1 we give some additional definitions and state auxiliary results. Then in Subsection 3.2 we sketch the proof itself.

### 3.1 Auxiliary results

To solve CGWC for connected graphs, we use the recursive understanding technique introduced by Chitnis et al. in [19]. Therefore, we need notions that are specific to this technique and some results established by Chitnis et al. [19]. Note that we adapt the definitions and the statements of the results for the case of edge weighted graphs.

**Weighted good edge separations.** Let $G$ be a connected weighted graph with an edge weight function $w\colon E(G) \to \mathbb{N}$. Let also $p$ and $q$ be positive integers. A partition $(A, B)$ of $V(G)$ is called a $(q,p)$-*good edge separation* if

- $|A| > q$ and $|B| > q$,
- $\mathsf{w}(A, B) \leq p$,
- $G[A]$ and $G[B]$ are connected.

It is said that $G$ is $(q,p)$-*unbreakable* if for any partition $(A, B)$ of $V(G)$ such that $\mathsf{w}(A, B) \leq p$, it holds that $|A| \leq q$ or $|B| \leq q$.

We use the following variant of Lemma 7 of [19] that is more convenient for our purposes. For the unweighted case, this variant was stated in [31].

▶ **Lemma 2.** *There exists a deterministic algorithm that, given a weighted connected graph $G$ along with positive integers $p$ and $q$, in time $2^{\mathcal{O}(\min\{p,q\}\log(p+q))} \cdot n^3 \log n$ either finds a $(q,p)$-good edge separation or correctly concludes that $G$ is $(pq,p)$-unbreakable.*

**Mimicking connectivities by cut reductions.** Let $r$ be a nonnegative integer. A pair $(G, \mathbf{x})$, where $G$ is a graph and $\mathbf{x} = \langle x_1, \ldots, x_r\rangle$ is an $r$-tuple of distinct vertices of $G$ is called an *$r$-boundaried graph* or simply a *boundaried graph*. Respectively, $\mathbf{x} = \langle x_1, \ldots, x_r\rangle$ is a *boundary*. Note that a boundary is an ordered set. Hence, two $r$-boundaried graphs that differ only by the order of the vertices in theirs boundaries are distinct. Still, we can treat $\mathbf{x}$ as a set when the ordering is irrelevant. Observe also that a boundary could be empty. Slightly abusing notation, we may say that $G$ is a ($r$-) boundaried graph assuming that a boundary is given. We say that $(G, \mathbf{x})$ is a *properly* boundaried graph if the vertices of $\mathbf{x}$ are pairwise nonadjacent and each component of $G$ contains at least one vertex of $\mathbf{x}$.

Two $r$-boundaried weighted graphs $(G_1, \mathbf{x}^{(1)})$ and $(G_2, \mathbf{x}^{(2)})$, where $\mathbf{x}^{(h)} = \langle x_1^{(h)}, \ldots, x_r^{(h)}\rangle$ for $h = 1, 2$, are *isomorphic* if there is an isomorphism of $G_1$ to $G_2$ that maps each $x_i^{(1)}$ to $x_i^{(2)}$ for $i \in \{1, \ldots, r\}$ and each edge is mapped to an edge of the same weight.

Let $(G_1, \mathbf{x}^{(1)})$ and $(G_2, \mathbf{x}^{(2)})$ be $r$-boundaried graphs with $\mathbf{x}^{(h)} = \langle x_1^{(h)}, \ldots, x_r^{(h)}\rangle$ for $h = 1, 2$, and assume that $(G_2, \mathbf{x}^{(2)})$ is a properly boundaried graph. We define the *boundary sum* $(G_1, \mathbf{x}^{(1)}) \oplus_b (G_2, \mathbf{x}^{(2)})$ (or simply $G_1 \oplus_b G_2$) as the (non-boundaried) graph obtained by taking disjoint copies of $G_1$ and $G_2$ and identifying $x_i^{(1)}$ and $x_i^{(2)}$ for each $i \in \{1, \ldots, r\}$. Note that the definition is not symmetric as we require that $(G_2, \mathbf{x}^{(2)})$ is a properly boundaried graph and we have no such a restriction for $x_1^{(1)}, \ldots, x_r^{(1)}$.

Let $\mathbf{X} = (X_1, \ldots, X_p)$ and $\mathbf{Y} = (Y_1, \ldots, Y_q)$ be two partitions of a set $Z$. We define the *product* $\mathbf{X} \times \mathbf{Y}$ of $\mathbf{X}$ and $\mathbf{Y}$ as the partition of $Z$ obtained from $\{X_i \cap Y_j \mid 1 \leq i \leq p,\ 1 \leq j \leq q\}$ by the deletion of empty sets. For partitions $\mathbf{X}^1, \ldots, \mathbf{X}^r$ of $Z$, we denote their *consecutive product* as $\prod_{i=1}^{r} \mathbf{X}^i$.

Let $(H, \mathbf{x})$ be a connected properly $r$-boundaried weighted graph. Let $p$ be a positive integer or $+\infty$. Slightly abusing notation we consider here $\mathbf{x}$ as a set. We construct the partition $\mathbf{Z}$ of $V(H)$ as follows. For $X \subseteq \mathbf{x}$, denote $\overline{X} = \mathbf{x} \setminus X$.

- For all distinct pairs $\{X, \overline{X}\}$ for nonempty $X \subset \mathbf{x}$, find a minimum weight $(X, \overline{X})$-separator $S_X = E(Y_X^1, Y_X^2)$ where $(Y_X^1, Y_X^2)$ is a partition of $V(H)$, $X \subseteq Y_X^1$ and $\overline{X} \subseteq Y_X^2$.
- For every $v \in V(H) \setminus \mathbf{x}$, find a minimum weight $(\mathbf{x}, \{v\})$-separator $S(v)$. Find $v^* \in V(H) \setminus \mathbf{x}$ such that $w(S(v^*)) = \min\{w(S(v)) \mid v \in V(H) \setminus \mathbf{x}\}$ and let $S(v^*) = E(Y_{\mathbf{x}}^1, Y_{\mathbf{x}}^2)$ where $(Y_{\mathbf{x}}^1, Y_{\mathbf{x}}^2)$ is a partition of $V(H)$ and $\mathbf{x} \subseteq Y_{\mathbf{x}}^1$.

- Construct the following partition of $V(H)$:

$$\mathbf{Z} = (Z_1, \ldots, Z_h) = \Big( \prod_{\substack{\text{distinct } \{X, \overline{X}\} \\ \varnothing \neq X \subset \mathbf{x}}} (Y_X^1, Y_X^2) \Big) \times (Y_{\mathbf{x}}^1, Y_{\mathbf{x}}^2) \times (\{x_1\}, \ldots, \{x_r\}, V(H) \setminus \mathbf{x}). \quad (1)$$

We construct $H'$ by performing the weighted contraction of the sets of $\mathbf{Z}$. Then for each edge $uv$ of $H'$ with $\mathsf{w}(uv) > p$, we set $\mathsf{w}(uv) = p$, that is, we truncate the weights by $p$. Notice that because the partition $(\{x_1\}, \ldots, \{x_r\}, V(H) \setminus \mathbf{x})$ is participating the product defining $\mathbf{Z}$, we have that $\{x\} \in \mathbf{Z}$ for each $x \in \mathbf{x}$, that is, the elements of the boundary are not contracted, and this is the only purpose of this partition in (1). We say that $H'$ is obtained from $(H, \mathbf{x})$ by the *cut reduction with respect to* $p$. Note that $H'$ is not unique as the construction depends on the choice of separators. It could be observed that we construct a *mimicking network* representing cuts of $H$ [44, 50] (see also [51]).

We extend this definition for disconnected graphs. Let $(H, \mathbf{x})$ be a properly $r$-boundaried weighted graph and let $p$ be a positive integer or $+\infty$. Denote by $H_1, \ldots, H_s$ the components of $H$ and let $\mathbf{x}^i = \mathbf{x} \cap V(H_i)$ for $i \in \{1, \ldots, s\}$. Consider the boundaried graphs $(H_i', \mathbf{x}^i)$ obtained from $(H_i, \mathbf{x})$ by cut reduction with respect to $p$ for $i \in \{1, \ldots, s\}$. We say that $(H', \mathbf{x})$, that is obtained by taking the union of $(H_i', \mathbf{x}^i)$ for $i \in \{1, \ldots, s\}$, is obtained by the *cut reduction with respect to* $p$.

The crucial property of $H'$ is that it keeps the separators of $H$ that are essential for the connectivity.

▶ **Lemma 3.** *Let $(H, \mathbf{x})$ be a properly $r$-boundaried weighted graph, and let $p \in \mathbb{N} \cup \{+\infty\}$ and $t \in \mathbb{N}$. Let also $(F, \mathbf{y})$ be an $r$-boundaried weighted graph, and let $G = (F, \mathbf{y}) \oplus_b (H, \mathbf{x})$. Then for an $r$-boundaried weighted graph $H'$ obtained from $H$ by the cut reduction with respect to $p$ and $t$ positive integers $\lambda_1, \ldots, \lambda_t \leq p$ it holds that $G$ has exactly $t$ components and they have the connectivities $\lambda_1, \ldots, \lambda_t$ respectively if and only if the same holds for $G' = (F, \mathbf{y}) \oplus_b (H', \mathbf{x})$, that is, $G'$ has $t$ components and they have the connectivities $\lambda_1, \ldots, \lambda_t$ respectively.*

It is also important to observe that it holds that $|V(H')| \leq 2^{2^{r-1}} + r$, that is, the size of an $r$-boundaried weighted graph obtained by cut reduction is bounded by a function of $r$.

For positive integers $r$ and $s$, we define $\mathcal{H}_{r,s}$ as the family of all pairwise nonisomorphic properly $r$-boundaried weighted graphs $(G, \mathbf{x})$ with at most $2^{2^{r-1}} + r$ vertices where the weights of edges are in $\{1, \ldots, s\}$ and for every component $C$ of $G$ with $V(C) \setminus \mathbf{x} \neq \varnothing$, there is a vertex $v \in V(C) \setminus \mathbf{x}$ such that $\lambda^{\mathsf{w}}(\mathbf{x}, v) \leq s$. We also formally define $\mathcal{H}_{0,s}$ as the set containing the empty graph. Note that $|\mathcal{H}_{r,s}| \leq (s+1)^{\binom{2^{2^{r-1}}}{2} + r}$ and $\mathcal{H}_{r,s}$ can be constructed in time $2^{2^{2^{\mathcal{O}(r)}}} \log s$.

**Variants of CGWC.** To apply the recursive understanding technique, we also have to solve a special variant of CGWC tailored for recursion. To define it, we first introduce the following variant of the problem. The difference is that a solution should be chosen from a given subset of edges.

Annotated CGWC

*Input:* A weighted graph $G$ with an edge weight function $w\colon E(G) \to \mathbb{N}$, $L \subseteq E(G)$, a $t$-tuple $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$, where $\lambda_i \in \mathbb{N} \cup \{+\infty\}$ for $i \in \{1, \ldots, t\}$ and $\lambda_1 \leq \ldots \leq \lambda_t$, and a nonnegative integer $k$.

*Task:* Decide whether there is a set $F \subseteq L$ with $\mathsf{w}(F) \leq k$ such that $G - F$ has $t$ connected components $G_1, \ldots, G_t$ where each $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ for $i \in \{1, \ldots, t\}$.

Clearly, if $L = E(G)$, then Annotated CGWC is CGWC. Let $(G, w, L, \Lambda, k)$ be an instance of Annotated CGWC. We say that $F \subseteq L$ with $\mathsf{w}(F) \leq k$ such that $G - F$ has $t$ connected components $G_1, \ldots, G_t$ where each $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ for $i \in \{1, \ldots, t\}$ is a *solution* for the instance.

Now we define Border A-CGWC.

Border A-CGWC

*Input:* A weighted $r$-boundaried connected graph $(G, \mathbf{x})$ with an edge weight function $w\colon E(G) \to \mathbb{N}$, $L \subseteq E(G)$, a $t$-tuple $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$, where $\lambda_i \in \mathbb{N} \cup \{+\infty\}$ for $i \in \{1, \ldots, t\}$ and $\lambda_1 \leq \ldots \leq \lambda_t$, and a nonnegative integer $k$ such that $r \leq 4k$ and $k \geq t - 1$.

*Task:* For each weighted properly $r$-boundaried graph $(H, \mathbf{y}) \in \mathcal{H}_{r,2k}$ and each $\hat{\Lambda} = \langle \hat{\lambda}_1, \ldots, \hat{\lambda}_s \rangle \subseteq \Lambda$, find the minimum $0 \leq \hat{k} \leq k$ such that $((G, \mathbf{x}) \oplus_b (H, \mathbf{y}), w, L, \hat{\Lambda}, \hat{k})$ is a yes-instance of Annotated CGWC and output a solution $F$ for this instance or output $\varnothing$ if $\hat{k}$ does not exist.

Slightly abusing notation, we use $w$ to denote the weights of edges of $G$ and $H$. Notice that Border A-CGWC is neither decision nor optimization problem, and its solution is a list of subsets of $L$. Observe also that a solution of Border A-CGWC is not necessarily unique. Still, for any two solutions, that is, lists $\mathcal{L}_1$ and $\mathcal{L}_2$ of subsets of $L$, the following holds: for each weighted properly $r$-boundaried graph $(H, \mathbf{y}) \in \mathcal{H}_{r,2k}$ and each $\hat{\Lambda} = \langle \hat{\lambda}_1, \ldots, \hat{\lambda}_s \rangle \subseteq \Lambda$, the lists $\mathcal{L}_1$ and $\mathcal{L}_2$ contain the sets of the same weight. To solve Annotated CGWC, it is sufficient to solve Border A-CGWC for $r = 0$. If the output contains nonempty set for $\hat{\Lambda} = \Lambda$, we have a yes-instance of Annotated CGWC. If the output contains empty set for this $\hat{\Lambda}$, we should verify additionally whether $\varnothing$ is a solution, that is, whether $\Lambda = \{\lambda_1\}$ and $\lambda^{\mathsf{w}}(G) \geq \lambda_1$. To apply the recursive understanding technique, we first solve Border A-CGWC for $(q, 2k)$-unbreakable graphs for some appropriate value of $q$ and then use this result for the general case of Border A-CGWC.

**Restricted BFS subgraphs.** Let $G$ be a graph, $u \in V(G)$, and let $r$ be a positive integer. We construct the subgraph $B_r(u)$ using a modified breadth-first search algorithm. Recall that in the standard breadth-first search algorithm (see, e.g., [20]) starting from $u$, we first label $u$ by $\ell(u) = 0$ and put $u$ into a queue $Q$. Then we iterate as follows: if $Q$ is nonempty, then take the first vertex $x$ in the queue and for every nonlabeled neighbor $y$, assign $\ell(y) = \ell(x) + 1$ and put $y$ into $Q$. We start in the same way by assigning $u$ the label $\ell(u) = 0$ and putting $u$ into $Q$. Then while $Q$ is nonempty and the first element $x$ has the label $\ell(x) \leq r - 1$, we consider arbitrary chosen $\min\{r, d_G(x)\}$ vertices $y \in N_G(x)$, assign to unlabeled vertices $y$ the label $\ell(y) = \ell(x) + 1$ and put them into $Q$. The graph $B_r(u)$ is the subgraph of $G$ induced by the labeled vertices $v$ with $\ell(v) \leq r$. We say that $B_r(x)$ is an *$r$-restricted BFS subgraph* of $G$. Note that such a subgraph is not unique.

## 3.2 Sketch of the proof of Theorem 1

First, we construct an algorithm for Border A-CGWC for connected $(q, 2k)$-unbreakable graphs. The crucial step is to solve Annotated CGWC.

▶ **Lemma 4.** Annotated CGWC *can be solved and a solution can be found (if exists) in time* $2^{\mathcal{O}(q(q+k)\log(q+k))} \cdot n^{\mathcal{O}(1)}$ *for connected* $(q, 2k)$*-unbreakable graphs.*

**Sketch of the proof.** Let $(G, w, L, \Lambda, k)$ be an instance of Annotated CGWC where $G$ is a connected $(q, 2k)$-unbreakable graph. Let also $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$, $\lambda_1 \leq \ldots \leq \lambda_t$. Clearly, the problem is easy if $t = 1$ and the problem is trivial if $t > k + 1$, because a connected graph $G$ can be separated into at most $k + 1$ components by at most $k$ edge deletions. Let $2 \leq t \leq k + 1$. If $|V(G)| \leq 3q$, we solve Annotated CGWC using brute force. From now we assume that $|V(G)| > 3q$.

Suppose that $(G, w, L, \Lambda, k)$ is a yes-instance of Annotated CGWC and let $F \subseteq L$ be a solution. Let $G_1, \ldots, G_t$ be the components of $G - F$ and $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ for $i \in \{1, \ldots, t\}$. Using that $G$ is a $(q, 2k)$-unbreakable graph, we show that there is a component $G_i$ with at least $q + 1$ vertices and the total number of vertices in the other components is at most $q$. We say that $G_i$ is a *big* components and call the other components *small*. For each $i \in \{1, \ldots, t\}$, we verify whether there is a solution $F$ where $\lambda_i$ is the connectivity constraint for the big component of $G - F$.

Assume that $\lambda_i > k$. We show that in this case $V(G_i)$ is an inclusion maximal set of vertices $X$ of $G$ with the property that for every two vertices $u, v \in X$, $\lambda^{\mathsf{w}}_G(u, v) \geq k + 1$. We use this to find the big component and then find other clusters by brute force.

From now we assume that $\lambda_i \leq k$. To deal with this case we apply the *random separation* technique introduces by Cai, Chan and Chan in [16]. To avoid dealing with randomized algorithms, we use the Lemma 1 of [19]. Assume again that $(G, w, L, \Lambda, k)$ is a yes-instance of Annotated CGWC, $F \subseteq L$ is a solution, and $G_1, \ldots, G_t$ are the components of $G - F$ where $G_i$ is the big component. Let $A = \bigcup_{j \in \{1,\ldots,t\} \setminus \{i\}} V(G_j)$. Recall that $|A| \leq q$. Let also $X \subseteq V(G_i)$ be the set of vertices of $G_i$ that have neighbors in $A$. Note that $|X| \leq k$. For each $u \in X$, we consider a $(q + \lambda_i)$-restricted BFS subgraph $B(u) = B_{q+\lambda_i}(u)$ of $G_i$. Let $B = \bigcup_{u \in X} V(B(u))$. We have that $|V(B(u))| = 2^{\mathcal{O}((q+k)\log(q+k))}$ since $\lambda_i \leq k$. Hence, $|B| = 2^{\mathcal{O}((q+k)\log(q+k))}$. Note also that $|B| \geq q + 1$. We say that a set $S \subseteq V(G)$ is $(A, B)$-*good* or, simply, *good* if $B \subseteq S$ and $A \cap S = \emptyset$. By Lemma 1 of [19], we can construct in time $2^{\mathcal{O}(q(q+k)\log(q+k))} \cdot n \log n$ a family $\mathcal{S}$ of at most $2^{\mathcal{O}(q(q+k)\log(q+k))} \cdot \log n$ subsets of $V(G)$ such that if $(G, w, L, \Lambda, k)$ is a yes-instance and $A$ and $B$ exist for some solution, then $\mathcal{S}$ contains an $(A, B)$-good set.

We construct such a family $\mathcal{S}$, and for each $S \in \mathcal{S}$, we look for $F \subseteq L$ such that the following holds:

- **(i)** $\mathsf{w}(F) \leq k$,
- **(ii)** $G - F$ has $t$ components $G_1, \ldots, G_t$ such that each $G_j$ is weight $\lambda_j$-connected and $|V(G_i)| > q$, and
- **(iii)** $S \subseteq V(G_i)$.

We describe the algorithm that produces the YES answer if $S$ is good and, moreover, if the algorithm outputs YES, then $(G, w, L, \Lambda, k)$ is a yes-instance of Annotated CGWC. Note that the algorithm can output the false NO answer if $S$ is not a good set. Nevertheless, because for an yes-instance of Annotated CGWC, we always have a good set $S \in \mathcal{S}$, we have that $(G, w, L, \Lambda, k)$ is a yes-instance if and only if the algorithm outputs YES for at least one $S \in \mathcal{S}$.

The algorithm uses the following property of $(A, B)$-good sets.

▷ Claim (A). *If $S$ is an $(A, B)$-good set, then for each component $H$ of $G - S$, either $V(H) \subseteq V(G_i)$ or $V(H) \cap V(G_i) = \emptyset$.*

We apply a number of reduction rules that either increase the set $S$ or conclude that $S$ is not good and stop. Each rule increasing $S$ is applied exhaustively. For each rule, we show

that it is *safe* in the sense that if we increase $S$, then if the original $S$ was good, then the obtained set is good as well, and if we conclude that the original $S$ is not good, then this is a correct conclusion and, therefore, we can return NO and stop. We underline that whenever we return NO in the rules, this means that we discard the current choice of $S$.

Due to the size of $B$, we get the following rule.

▶ **Reduction Rule 3.1.** If $|S| \leq q$, then return NO and stop.

Denote by $H_1, \ldots, H_s$ the components of $G - S$. Applying Claim A we obtain the next rules.

▶ **Reduction Rule 3.2.** For every $j \in \{1, \ldots, s\}$, if $E(V(H_j), S) \setminus L \neq \emptyset$ or $\mathsf{w}(V(H_j), S) \geq k+1$ or $|V(H_j)| > q$, then set $S = S \cup V(H_j)$.

▶ **Reduction Rule 3.3.** If there is $u \in S$ adjacent to a vertex of $H_j$ for some $j \in \{1, \ldots, s\}$ and there is a connected set $Z \subseteq S$ such that a) $u \in Z$, b) $|Z| \leq q$, c) $\mathsf{w}(Z, S \setminus Z) \leq \lambda_i - 1$, then set $S = S \cup V(H_j)$.

To apply the last rule, we use the result of Fomin and Villanger [34] that allows to list all the sets $Z$ satisfying a)–c) in time $2^{\mathcal{O}(k \log(q+k))} \cdot n$ because $\lambda_i \leq k$. We apply Reduction Rule 3.3 recomputing the components of $G - S$ after each modification of $S$. Then we again use the result of Fomin and Villanger [34] to apply the next rule.

▶ **Reduction Rule 3.4.** If there is a connected set $Z \subseteq S$ such that $|Z| \leq q$ and $\mathsf{w}(Z, V(G) \setminus Z) \leq \lambda_i - 1$, then set return NO and stop.

Assume that we do not stop while executing Reduction Rule 3.4. We use the flowing claim to identify the components of $G - S$ whose vertices, definitely, are not in the big cluster.

▷ Claim (B). *If $S$ is an $(A, B)$-good set, then if for a component $H$ of $G - S$, there is $v \in V(H)$ such that $\lambda^{\mathsf{w}}(v, S) < \lambda_i$, then $V(H) \cap V(G_i) = \emptyset$.*

Let $H_1, \ldots, H_s$ be the components of $G - S$. We set

$$I = \{j \in \{1, \ldots, s\} \mid \text{there is } v \in V(H_j) \text{ such that } \mathsf{w}(v, S) < \lambda_i\}.$$

▶ **Reduction Rule 3.5.** If $|\bigcup_{j \in I} V(H_j)| > q$ or $\mathsf{w}(\bigcup_{j \in I} V(H_j), S) > k$, then return NO and stop.

▷ Claim (C). *For any $J \subseteq \{1, \ldots, s\}$ such that $I \subseteq J$ and $\mathsf{w}(\bigcup_{j \in J} V(H_j), S) \leq k$, the graph $G' = G - \bigcup_{j \in J} V(H_j)$ is weight $\lambda_i$-connected.*

By applying Reduction Rules 3.1–3.5, we either increase $S$ or stop. Now we have to find an $F \subseteq L$ such that (i)–(iii) are fulfilled and, by applying Claims A and B, we impose two additional constraints:

- **(iv)** for every $j \in \{1, \ldots, s\} \setminus I$, either $V(H_j) \subseteq V(G_i)$ or $V(H_j) \cap V(G_i) = \emptyset$,
- **(v)** for every $j \in I$, $V(H_j) \cap V(G_i) = \emptyset$.

Note that by Claim C, we automatically obtain that $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ if (i), (iii)-(v) are fulfilled. Also because of Reduction Rule 3.1, we have that $|V(G_i)| > q$ if (iii) holds. Hence, we can replace (ii) by the relaxed condition:

- **(ii)** $G - F$ has $t$ components $G_1, \ldots, G_t$ such that $G_j$ is weight $\lambda_j$-connected for $j \in \{1, \ldots, t\}$, $j \neq i$.

We find $F$, if such a set exists, by a dynamic programming algorithm that sorts the vertices of $G - S$ staring with the components with indices in $I$. ◀

Using Lemma 4, we construct the algorithm for Border A-CGWC for connected $(q, 2k)$-unbreakable graphs.

▶ **Lemma 5.** Border A-CGWC *can be solved in time* $2^{q^3 2^{2^{\mathcal{O}(k)}}} \cdot n^{\mathcal{O}(1)}$ *for connected* $(q, 2k)$*-unbreakable graphs.*

Now we construct an algorithm for Border A-CGWC and this result implies Theorem 1.

▶ **Lemma 6.** Border A-CGWC *can be solved in time* $f(k) \cdot n^{\mathcal{O}(1)}$.

**Sketch of the proof.** We construct a recursive algorithm for Border A-CGWC. Let $(G, \mathbf{x}, w, L, \Lambda, k)$ be an instance of Border A-CGWC. Recall that $(G, \mathbf{x})$ is an $r$-boundaried connected graph and $r \leq k$. Recall also that $\Lambda$ contains at most $k+1$ elements.

We define the constants $p$ and $q$ that are used throughout the proof as follows:

$$p = 2k2^{k+1}(2k+1)^{\left(2^{2^{4k-1}}+4k\right)} + 4k \text{ and } q = 2^{2^{p-1}} + p. \tag{2}$$

We are going to use $q$ as a part of the unbreakability threshold.

We apply Lemma 2 and in time $2^{\mathcal{O}(k\log(q+k))} \cdot n^3 \log n$ either find a $(q, 2k)$-good edge separation $(A, B)$ of $G$ or conclude that $G$ is $(2kq, 2k)$-unbreakable.

If $G$ is $(2kq, 2k)$-unbreakable, we apply Lemma 5 and solve the problem in time $2^{q^3 2^{2^{\mathcal{O}(k)}}} \cdot n^{\mathcal{O}(1)}$. Assume from now that we are given a $(q, 2k)$-good edge separation $(A, B)$ of $G$.

Since $|\mathbf{x}| \leq 4k$ and the vertices of $\mathbf{x}$ are separated between $A$ and $B$, either $A$ or $B$ contains at most $2k$ vertices of $\mathbf{x}$. Assume without loss of generality that $|A \cap \mathbf{x}| \leq 2k$. Let $T$ be the set of end-vertices of the edges of $E(A, B)$ in $A$. Clearly, $|T| \leq 2k$. We form a new $\hat{r}$-tuple $\hat{\mathbf{x}} = \langle \hat{x}_1, \ldots, \hat{x}_{\hat{r}} \rangle$ of vertices $A$ from the vertices of $(A \cap \mathbf{x}) \cup T$; note that $\hat{r} \leq 4k$. We consider $\hat{G} = G[A]$ as the $\hat{\mathbf{x}}$-boundaried graph. We set $\hat{L} = L \cap E(\hat{G})$. This way, we obtain the instance $(\hat{G}, \hat{\mathbf{x}}, w, \hat{L}, \Lambda, k)$ of Border A-CGWC.

Now we solve Border A-CGWC for $(\hat{G}, \hat{\mathbf{x}}, w, \hat{L}, \Lambda, k)$.

If $|V(\hat{G})| \leq 2^q$, we can simply use brute force. If $|V(\hat{G})| > 2^q$, we recursively solve Border A-CGWC for $(\hat{G}, \hat{\mathbf{x}}, w, \hat{L}, \Lambda, k)$ by calling our algorithm for the instance that has lesser size, because $|V(\hat{G})| \leq |V(G)| - q$.

By solving Border A-CGWC for $(\hat{G}, \hat{\mathbf{x}}, \hat{L}, \Lambda, k)$, we obtain a list $\mathcal{L}$ of sets where each element is either $\emptyset$ or $F \subseteq \hat{L}$ that is a solution for the corresponding instance of Annotated CGWC for some $(H, \mathbf{y}) \in \mathcal{H}_{\hat{r},2k}$, $\hat{\Lambda} \subseteq \Lambda$ and $\hat{k} \leq k$. Denote by $M$ the union of all the sets in $\mathcal{L}$. Clearly, $M \subseteq \hat{L}$.

We define $L^* = (L \setminus \hat{L}) \cup M$. Since $M \subseteq \hat{L}$, $L^* \subseteq L$. We show that the instances $(G, \mathbf{x}, w, L, \Lambda, k)$ and $(G, \mathbf{x}, w, L^*, \Lambda, k)$ are essentially equivalent by proving the following claim by making use of Lemma 3.

▷ Claim (A). *For every weighted properly $r$-boundaried graph $(H, \mathbf{y}) \in \mathcal{H}_{r,2k}$, every $\hat{\Lambda} = \langle \hat{\lambda}_1, \ldots, \hat{\lambda}_s \rangle \subseteq \Lambda$ and every nonnegative integer $\hat{k} \leq k$, $((G, \mathbf{x}) \oplus_b (H, \mathbf{y}), w, L, \hat{\Lambda}, \hat{k})$ is a yes-instance of* Annotated CGWC *if and only if $((G, \mathbf{x}) \oplus_b (H, \mathbf{y}), w, L^*, \hat{\Lambda}, \hat{k})$ is a yes-instance of* Annotated CGWC.

By Claim A we obtain that every solution of $(G, \mathbf{x}, w, L^*, \Lambda, k)$ is a solution of $(G, \mathbf{x}, w, L, \Lambda, k)$, and there is a solution of $(G, \mathbf{x}, w, L, \Lambda, k)$ that is a solution of $(G, \mathbf{x}, w, L^*, \Lambda, k)$. Therefore, it is sufficient for us to solve $(G, \mathbf{x}, w, L^*, \Lambda, k)$.

Let $Z \subseteq A$ be the set of end-vertices of the edges of $M$ and the vertices of $\hat{\mathbf{x}}$. Because $\hat{r} \leq 4k$, $\mathcal{H}_{\hat{r},2k} \leq (2k+1)^{\left(2^{2^{4k-1}}+4k\right)}$. Since $t \leq k+1$, there are at most $2^{k+1}$ subtuples $\hat{\Lambda}$ of $\Lambda$.

For each $(H,y) \in \mathcal{H}_{\hat{r},2k}$ and $\hat{\Lambda} \subseteq \Lambda$, the solution $\mathcal{L}$ of Border A-CGWC for $(\hat{G},\hat{\mathbf{x}},\hat{L},\Lambda,k)$ contains a set $F$ of size at most $k$. This implies that

$$|M| \leq k2^{k+1}(2k+1)^{\left(2^{2^{4k-1}}+4k\right)}.$$

Because $|\hat{\mathbf{x}}| \leq 4k$, we obtain that

$$|Z| \leq 2|M|+4k \leq 2k2^{k+1}(2k+1)^{\left(2^{2^{4k-1}}+4k\right)}+4k = p \tag{3}$$

for $p$ defined in (2).

Let $U = A \setminus Z$. We define $Q = G - U$. Let also $R$ be the graph with the vertex set $A$ and the edge set $E(G[A]) \setminus E(G[Z])$. We order the vertices of $Z$ arbitrarily and consider $Z$ to be $|Z|$-tuple of the vertices of $Q$ and $R$. Observe that $(R,Z)$ is a properly $|Z|$-boundaried graph as $G[A]$ is connected. Since $V(F) \cap V(R) = Z$, we have that $G = (Q,Z) \oplus_b (R,Z)$. Let $(R^*,Z)$ be the boundaried graph obtained from $(R,Z)$ by the cut reduction with respect to $+\infty$. We define $G^* = (Q,Z) \oplus_b (R^*,Z)$. Note that $L^* \subseteq E(Q) \subseteq E(G^*)$. We show that we can replace $G$ by $G^*$ in the considered instance $(G,\mathbf{x},w,L^*,\Lambda,k)$ of Border A-CGWC by making use of Lemma 3.

▷ Claim (B). *For every weighted properly $r$-boundaried graph $(H,\mathbf{y}) \in \mathcal{H}_{r,2k}$, every $\hat{\Lambda} = \langle \hat{\lambda}_1,\ldots,\hat{\lambda}_s\rangle \subseteq \Lambda$ and every nonnegative integer $\hat{k} \leq k$, a set $F \subseteq L^*$ is a solution for the instance $((G,\mathbf{x}) \oplus_b (H,\mathbf{y}),w,L^*,\hat{\Lambda},\hat{k})$ if and only if $F$ is a solution for $((G^*,\mathbf{x}) \oplus_b (H,\mathbf{y}),w,L^*,\hat{\Lambda},\hat{k})$.*

By Claim B, to solve Border A-CGWC for $(G,\mathbf{x},w,L^*,\Lambda,k)$, it is sufficient to solve the problem for $(G^*,\mathbf{x},w,L^*,\Lambda,k)$. Observe that $|V(G^*)| = |B| + |V(R^*)|$. Because $(R^*,Z)$ is obtained by the cut reduction, we can show that $|V(R^*)| \leq 2^{2^{|Z|-1}} + |Z|$. Using (3), we have that

$$|V(R^*)| \leq 2^{2^{p-1}} + p = q$$

for $q$ defined in (2). Recall that $|A| > q$ since $(A,B)$ is a $(q,2k)$-good edge separation of $G$. Therefore,

$$|V(G^*)| = |B| + |V(R^*)| \leq |B| + q < |A| + |B| = |V(G)|.$$

We use this and solve Border A-CGWC for $(G^*,\mathbf{x},w,L^*,\Lambda,k)$ recursively by applying our recursive algorithm for the instance with the input graph of smaller size. ◀

## 4 The algorithm for Clustering to Given Weighted Connectivities

In this section we extend the result obtained in Section 3 and show that CGWC is FPT when parameterized by $k$ even if the input graph is disconnected. Let $\alpha = \langle \alpha_1,\ldots,\alpha_t\rangle$ where $\alpha_i \in \mathbb{N} \cup \{+\infty\}$ for $i \in \{1,\ldots,t\}$ and $\alpha_1 \leq \ldots \leq \alpha_t$. We call the *variate* of $\alpha$ the set of distinct elements of $\alpha$ and denote it by $\mathbf{var}(\alpha)$. Let also $\beta = \langle \beta_1,\ldots,\beta_t\rangle$ where $\beta_i \in \mathbb{N} \cup \{+\infty\}$ for $i \in \{1,\ldots,s\}$ and $\beta_1 \leq \ldots \leq \beta_t$. We write $\alpha \preceq \beta$ if $\alpha_i \leq \beta_i$ for $i \in \{1,\ldots,t\}$.

▶ **Theorem 7.** CGWC *can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$.*

**Sketch of the proof.** Let $(G, w, \Lambda, k)$ be an instance CGWC, $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$.

We find the components of $G$ and compute their weighted connectivities using the algorithm of Stoer and Wagner [66] for finding minimum cuts. Assume that $G_1, \ldots, G_s$ are the components of $G$ and $\lambda^{\mathsf{w}}(G_1) \leq \ldots \leq \lambda^{\mathsf{w}}(G_s)$. If $s > t$, then $(G, w, \Lambda, k)$ is a trivial no-instance. If $s < t - k$, then $(G, w, \Lambda, k)$ is no-instance, because by deleting at most $k$ edges it is possible to obtain at most $k$ additional components. In all these cases we return the corresponding answer and stop. From now we assume that $t - k \leq s < t$. We exhaustively apply the following reduction rule based on the observation that a component of high connectivity cannot be split.

▶ **Reduction Rule 4.1.** If there is $i \in \{1, \ldots, s\}$ such that $\lambda^{\mathsf{w}}(G_i) > k$, then find the maximum $j \in \{1, \ldots, t\}$ such that $\lambda^{\mathsf{w}}(G_i) \geq \lambda_j$ and set $G = G - V(G_i)$ and $\Lambda = \Lambda \setminus \{\lambda_j\}$.

To simplify notations, assume that $G$ with its components $G_1, \ldots, G_s$ and $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$ is obtained from the original input by the exhaustive application of Reduction Rule 4.1. Note that we still have that $t - k \leq s < t$. It may happen that $s = 0$, that is, $G$ became empty after the application of the rule. In this case $(G, w, \Lambda, k)$ is a trivial no-instance, and we return NO and stop. Assume that $s \geq 1$. Observe that we obtain that $\lambda^{\mathsf{w}}(G_i) \leq k$ for $i \in \{1, \ldots, s\}$, because Reduction Rule 4.1 cannot be applied any more. If $|\mathbf{var}(\Lambda)| > 3k$, then it could be shown that we have a no-instance of the problem. Respectively, we return NO and stop. Assume that $|\mathbf{var}(\Lambda)| \leq 3k$, that is, the variety of $\Lambda$ is bounded. We use this to construct the FPT Turing reduction of the problem to the special case when $\lambda^{\mathsf{w}}(C) = \lambda \leq k$ for every component $C$ of the input graph $G$. For this special case, we solve CGWC by constructing the FPT Turing reduction of CGWC based on Theorem 1 to the Minimum Cost Matching problem that then can be solved in polynomial time by, e.g, the Hungarian method [54, 55]. ◀

## 5 Conclusion

We proved that Clustering to Given Connectivities is FPT when parameterized by $k$. We obtained this result by making use of the recursive understanding technique [19]. The drawback of this approach is that the dependence of the running time on the parameter is huge and it seems unlikely that using the same approach one could avoid towers of the exponents similar to the function in Theorem 7. In particular, we do not see how to avoid using mimicking networks (see [44, 50] for the definitions and lower and upper bounds for the size of such networks) whose sizes are double-exponential in the number of terminals.

It can be observed that if we wish to prove just the *existence* of a (non-constructive) FPT-algorithm for CGWC, we can use a slightly different approach based on the meta-algorithmic result of Lokshtanov et al. [57] which applies to problems that can be expressed in Counting Monadic Second Order Logic (CMSOL).

▶ **Proposition 8** ( [57]). *Let $\psi$ be a CMSOL sentence. For all $c \in \mathbb{N}$, there exists $s \in \mathbb{N}$ such that if there exists an algorithm that decides whether $G \models \psi$ on $(s, c)$-unbreakable graphs in time $f(|\psi|) \cdot n^{O(1)}$ then there exists an algorithm that decides whether $G \models \psi$ on general graphs in time $f(|\psi|) \cdot n^{O(1)}$.*

We can use Proposition 8 to obtain a weaker version of Theorem 1 where the function $f$ is not any more computable. Notice that we cannot express in CMSOL the connectivity lower bounds imposed by the $t$-tuple $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$ directly, because the values of $\lambda_i$ are not bounded by any function of the parameter $k$. Hence, we have to go around of this issue. Let

$I = (G, \mathsf{w}, \Lambda, k)$ be an instance of CGWC. We consider a partition $\mathcal{S} = \{S_1, \ldots, S_q\}$ of $V(G)$ such that two vertices $x, y$ belong in the same set if and only $\lambda^{\mathsf{w}}(x, y) \geq k + 1$. Clearly, $\mathcal{S}$ can be computed in polynomial time. A key observation is that if $\lambda^{\mathsf{w}}(G[S_i]) \geq k + 1$, then either $G[S_i]$ is one of the clusters of a solution or $G[S_i]$ is a part of a cluster of a solution whose weighted edge connectivity is at most $k$. For each $i \in \{1, \ldots, q\}$ where $\lambda^{\mathsf{w}}(G[S_i]) \geq k + 1$, we contract all vertices in $S_i$ to a single vertex and, in the contracted graph $G'$, we assign to this new vertex a weight equal to $\lambda^{\mathsf{w}}(G[S_i])$. We also assign the weight $+\infty$ to the rest of the vertices of $G'$. Recall now that when $G$ is connected, $\Lambda = \langle \lambda_1, \ldots, \lambda_t \rangle$ has at most $k + 1$ different values for a yes-instance and for each $i \in \{1, \ldots, t\}$ we set up a set $R_i$ that contains all vertices of $G'$ that have weight at least $\lambda_i$. We now consider the structure $\alpha = (G', R_1, \ldots, R_t)$ and a generalization of the connected version of CGWC where the input has a structure $\alpha$ instead of a connected graph and where, in the question of the new problem, we additionally demand that $V(G_i) \subseteq R_i, i \in \{1, \ldots, t\}$ and also we ask $\lambda^{\mathsf{w}}(G_i) \geq \lambda_i$ only when $\lambda_i \leq k$, while we demand that $|V(G_i)| = 1$ when $\lambda_i \geq k + 1$. We call this new problem Generalized-CGWC and we observe that $I = (G, \mathsf{w}, \Lambda, k)$ is a yes-instance of CGWC if and only if $I' = ((G', R_1, \ldots, R_t), \mathsf{w}, \Lambda, k)$ is a yes-instance of Generalized-CGWC . It is now possible to verify that Generalized-CGWC can be expressed using CMSOL for every fixed value of $k$ and given $\Lambda$ as there are no unbounded connectivities to encode. To apply Proposition 8, we have to solve Generalized-CGWC on unbreakable graphs and this can be done similarly to the proof of Lemma 4. Therefore, we can derive the existence of an FPT-algorithm for CGWC on connected graphs and further extend this to general graphs using the reduction of Section 4.

We would like to underline that due to the plug-in of Proposition 8, the alternative approach provided by the above discussion does not provide *any computable function* bounding the parametric dependence of the running time. Under the light of such an alternative, the algorithm described in Section 3 appears to be "less huge" that it might appear by first sight. This is the main reason why we chose to use a more direct approach in our results. In fact Lemma 6 may be seen as a "constructive detour" to Proposition 8.

The natural question would be to ask whether we can get a better running time using different techniques. This question is interesting even for some special cases of CGC when the connectivity constraints are bounded by a constant or are the same for all components. From the other side, it is natural to ask about lower bounds on the running time. For an FPT parameterized problem, it is natural to ask whether it admits a polynomial kernel. We observe that it is unlikely that CGWC has a polynomial kernel even if there are no weights and the maximum connectivity constraint is one, because it was shown by Cygan et al. in [22] that already $t$-Cut parameterized by the solution size $k$ has no polynomial kernel unless NP $\subseteq$ coNP /poly. Another direction of research is to consider vertex connectivities instead of edge connectivities.

## References


1 Nir Ailon, Moses Charikar, and Alantha Newman. Proofs of Conjectures in "Aggregating Inconsistent Information: Ranking and Clustering". Technical Report TR-719-05, Department of Computer Science, Princeton University, USA, 2005.

2 Noga Alon, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. Quadratic forms on graphs. *Inventiones mathematicae*, 163(3):499–522, March 2006.

3 Sanjeev Arora, Eli Berger, Elad Hazan, Guy Kindler, and Muli Safra. On Non-Approximability for Quadratic Programs. In *46th Annual IEEE Symposium on Foundations of Computer*

*Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 206–215, 2005.

4 Balabhaskar Balasundaram, Sergiy Butenko, and Illya V. Hicks. Clique Relaxations in Social Network Analysis: The Maximum $k$-Plex Problem. *Operations Research*, 59(1):133–142, 2011. doi:10.1287/opre.1100.0851.

5 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation Clustering. *Machine Learning*, 56(1):89–113, July 2004.

6 Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering Gene Expression Patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.

7 Pavel Berkhin. A Survey of Clustering Data Mining Techniques. In *Grouping Multidimensional Data - Recent Advances in Clustering*, pages 25–71. Springer, 2006.

8 Nadja Betzler, Jiong Guo, Christian Komusiewicz, and Rolf Niedermeier. Average parameterization and partial kernelization for computing medians. *Journal of Computer and System Sciences*, 77(4):774–789, 2011.

9 Ivan Bliznets and Nikolay Karpov. Parameterized Algorithms for Partitioning Graphs into Highly Connected Clusters. In *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, pages 6:1–6:14, 2017.

10 S. Böcker, S. Briesemeister, Q.B.A. Bui, and A. Truss. Going weighted: Parameterized algorithms for cluster editing. *Theoretical Computer Science*, 410(52):5467–5480, 2009.

11 Sebastian Böcker. A golden ratio parameterized algorithm for Cluster Editing. *Journal of Discrete Algorithms*, 16:79–89, 2012. Selected papers from the 22nd International Workshop on Combinatorial Algorithms (IWOCA 2011). doi:10.1016/j.jda.2012.04.005.

12 Sebastian Böcker and Jan Baumbach. Cluster Editing. In Paola Bonizzoni, Vasco Brattka, and Benedikt Löwe, editors, *The Nature of Computation. Logic, Algorithms, Applications*, pages 33–44, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

13 Sebastian Böcker, Sebastian Briesemeister, Quang Bao Anh Bui, and Anke Truß. A Fixed-Parameter Approach for Weighted Cluster Editing. In *Proceedings of the 6th Asia-Pacific Bioinformatics Conference, APBC 2008, 14-17 January 2008, Kyoto, Japan*, pages 211–220, 2008.

14 Sebastian Böcker and Peter Damaschke. Even faster parameterized cluster deletion and cluster editing. *Information Processing Letters*, 111(14):717–721, 2011.

15 Hans L. Bodlaender, Michael R. Fellows, Pinar Heggernes, Federico Mancini, Charis Papadopoulos, and Frances Rosamond. Clustering with partial information. *Theoretical Computer Science*, 411(7):1202–1211, 2010.

16 Leizhen Cai, Siu Man Chan, and Siu On Chan. Random Separation: A New Method for Solving Fixed-Cardinality Optimization Problems. In *IWPEC*, volume 4169 of *Lecture Notes in Computer Science*, pages 239–250. Springer, 2006. doi:10.1007/11847250_22.

17 Yixin Cao and Jianer Chen. Cluster Editing: Kernelization Based on Edge Cuts. *Algorithmica*, 64(1):152–169, September 2012.

18 Jianer Chen and Jie Meng. A $2k$ kernel for the cluster editing problem. *Journal of Computer and System Sciences*, 78(1):211–220, 2012. JCSS Knowledge Representation and Reasoning. doi:10.1016/j.jcss.2011.04.001.

19 Rajesh Chitnis, Marek Cygan, Mohammad Taghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT Algorithms for Cut Problems Using Randomized Contractions. *SIAM J. Comput.*, 45(4):1171–1229, 2016. doi:10.1137/15M1032077.

20 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009. URL: http://mitpress.mit.edu/books/introduction-algorithms.

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

**22** Marek Cygan, Stefan Kratsch, Marcin Pilipczuk, Michal Pilipczuk, and Magnus Wahlström. Clique Cover and Graph Separation: New Incompressibility Results. *TOCT*, 6(2):6:1–6:19, 2014.

**23** Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Minimum bisection is fixed parameter tractable. In *STOC 2014*, pages 323–332. ACM, 2014. doi:10.1145/2591796.2591852.

**24** Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

**25** Martin Dörnfelder, Jiong Guo, Christian Komusiewicz, and Mathias Weller. On the parameterized complexity of consensus clustering. *Theor. Comput. Sci.*, 542:71–82, 2014.

**26** Rodney G. Downey, Vladimir Estivill-Castro, Michael R. Fellows, Elena Prieto, and Frances A. Rosamond. Cutting Up is Hard to Do: the Parameterized Complexity of k-Cut and Related Problems. *Electr. Notes Theor. Comput. Sci.*, 78:209–222, 2003. doi:10.1016/S1571-0661(04)81014-4.

**27** Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

**28** Jubin Edachery, Arunabha Sen, and Franz J. Brandenburg. Graph Clustering Using Distance-k Cliques. In Jan Kratochvíyl, editor, *Graph Drawing*, pages 98–106, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

**29** Michael Fellows, Michael Langston, Frances Rosamond, and Peter Shaw. Efficient Parameterized Preprocessing for Cluster Editing. In Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors, *Fundamentals of Computation Theory*, 2007.

**30** Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

**31** Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Spanning Circuits in Regular Matroids. *CoRR*, abs/1607.05516, 2016. arXiv:1607.05516.

**32** Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michal Pilipczuk, and Yngve Villanger. Tight bounds for parameterized complexity of Cluster Editing with a small number of clusters. *J. Comput. Syst. Sci.*, 80(7):1430–1447, 2014.

**33** Fedor V. Fomin, Daniel Lokshtanov, Michal Pilipczuk, Saket Saurabh, and Marcin Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1419–1432, 2017.

**34** Fedor V. Fomin and Yngve Villanger. Treewidth computation and extremal combinatorics. *Combinatorica*, 32(3):289–308, 2012. doi:10.1007/s00493-012-2536-z.

**35** Ioannis Giotis and Venkatesan Guruswami. Correlation Clustering with a Fixed Number of Clusters. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '06, pages 1167–1176, Philadelphia, PA, USA, 2006. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1109557.1109686.

**36** Olivier Goldschmidt and Dorit S. Hochbaum. A Polynomial Algorithm for the k-cut Problem for Fixed k. *Math. Oper. Res.*, 19(1):24–37, 1994. doi:10.1287/moor.19.1.24.

**37** Petr A. Golovach, Pinar Heggernes, Paloma T. Lima, and Pedro Montealegre. Finding Connected Secluded Subgraphs. *CoRR*, abs/1710.10979, 2017. To appear in IPEC 2017. arXiv:1710.10979.

**38** Petr A. Golovach and Dimitrios M. Thilikos. Clustering to Given Connectivities. *CoRR*, abs/1803.09483, 2018.

**39** Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Graph-Modeled Data Clustering: Exact Algorithms for Clique Generation. *Theory of Computing Systems*, 38(4):373–392, July 2005.

**40** Martin Grohe, Ken-ichi Kawarabayashi, Dániel Marx, and Paul Wollan. Finding topological subgraphs is fixed-parameter tractable. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, (STOC 2011)*, pages 479–488, 2011. doi:10.1145/1993636.1993700.

41 Jiong Guo. A more effective linear kernelization for cluster editing. *Theoretical Computer Science*, 410(8):718–726, 2009. doi:10.1016/j.tcs.2008.10.021.

42 Jiong Guo, Iyad A. Kanj, Christian Komusiewicz, and Johannes Uhlmann. Editing Graphs into Disjoint Unions of Dense Clusters. *Algorithmica*, 61(4):949–970, 2011.

43 Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. A More Relaxed Model for Graph-Based Data Clustering: s-Plex Cluster Editing. *SIAM J. Discrete Math.*, 24(4):1662–1683, 2010. doi:10.1137/090767285.

44 Torben Hagerup, Jyrki Katajainen, Naomi Nishimura, and Prabhakar Ragde. Characterizing Multiterminal Flow Networks and Computing Flows in Networks of Small Treewidth. *J. Comput. Syst. Sci.*, 57(3):366–375, 1998. doi:10.1006/jcss.1998.1592.

45 Erez Hartuv and Ron Shamir. A clustering algorithm based on graph connectivity. *Inf. Process. Lett.*, 76(4-6):175–181, 2000.

46 Pinar Heggernes, Daniel Lokshtanov, Jesper Nederlof, Christophe Paul, and Jan Arne Telle. Generalized Graph Clustering: Recognizing ($p$, $q$)-Cluster Graphs. In *Graph Theoretic Concepts in Computer Science - 36th International Workshop, WG 2010, Zarós, Crete, Greece, June 28-30, 2010 Revised Papers*, pages 171–183, 2010.

47 Falk Hüffner, Christian Komusiewicz, Adrian Liebtrau, and Rolf Niedermeier. Partitioning Biological Networks into Highly Connected Clusters with Maximum Edge Coverage. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 11(3):455–467, 2014.

48 Falk Hüffner, Christian Komusiewicz, and Manuel Sorge. Finding Highly Connected Subgraphs. In *SOFSEM 2015: Theory and Practice of Computer Science - 41st International Conference on Current Trends in Theory and Practice of Computer Science, Pec pod Sněžkou, Czech Republic, January 24-29, 2015. Proceedings*, pages 254–265, 2015.

49 Ken-ichi Kawarabayashi and Mikkel Thorup. The Minimum k-way Cut of Bounded Size is Fixed-Parameter Tractable. In *FOCS 2011*, pages 160–169. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.53.

50 Arindam Khan and Prasad Raghavendra. On mimicking networks representing minimum terminal cuts. *Inf. Process. Lett.*, 114(7):365–371, 2014. doi:10.1016/j.ipl.2014.02.011.

51 Arindam Khan and Prasad Raghavendra. On mimicking networks representing minimum terminal cuts. *Information Processing Letters*, 114(7):365–371, 2014.

52 Eun Jung Kim, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. Parameterized algorithms for min-max multiway cut and list digraph homomorphism. *J. Comput. Syst. Sci.*, 86:191–206, 2017.

53 Christian Komusiewicz and Johannes Uhlmann. Alternative Parameterizations for Cluster Editing. In *SOFSEM 2011: Theory and Practice of Computer Science - 37th Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 22-28, 2011. Proceedings*, pages 344–355, 2011.

54 H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Res. Logist. Quart.*, 2:83–97, 1955. doi:10.1002/nav.3800020109.

55 H. W. Kuhn. Variants of the Hungarian method for assignment problems. *Naval Res. Logist. Quart.*, 3:253–258 (1957), 1956. doi:10.1002/nav.3800030404.

56 Daniel Lokshtanov and Dániel Marx. Clustering with Local Restrictions. *Inf. Comput.*, 222:278–292, January 2013.

57 Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Reducing CMSO Model Checking to Highly Connected Graphs. In *ICALP 2018*, volume 107 of *LIPIcs*, pages 135:1–135:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

58 Hannes Moser, Rolf Niedermeier, and Manuel Sorge. Exact combinatorial algorithms and experiments for finding maximum k-plexes. *J. Comb. Optim.*, 24(3):347–373, 2012. doi:10.1007/s10878-011-9391-5.

59 Rolf Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2006.

**60** Jeffrey Pattillo, Alexander Veremyev, Sergiy Butenko, and Vladimir Boginski. On the maximum quasi-clique problem. *Discrete Applied Mathematics*, 161(1):244–257, 2013. doi:10.1016/j.dam.2012.07.019.

**61** Jeffrey Pattillo, Nataly Youssef, and Sergiy Butenko. On clique relaxation models in network analysis. *European Journal of Operational Research*, 226(1):9–18, 2013. doi:10.1016/j.ejor.2012.10.021.

**62** Fábio Protti, Maise Dantas da Silva, and Jayme Luiz Szwarcfiter. Applying Modular Decomposition to Parameterized Cluster Editing Problems. *Theory of Computing Systems*, 44(1):91–104, January 2009.

**63** Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007. doi:10.1016/j.cosrev.2007.05.001.

**64** Shahram Shahinpour and Sergiy Butenko. Distance-Based Clique Relaxations in Networks: s-Clique and s-Club. In Boris I. Goldengorin, Valery A. Kalyagin, and Panos M. Pardalos, editors, *Models, Algorithms, and Technologies for Network Analysis*, pages 149–174, New York, NY, 2013. Springer New York.

**65** Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Appl. Math.*, 144(1–2):173–182, 2004. doi:10.1016/j.dam.2004.01.007.

**66** Mechthild Stoer and Frank Wagner. A simple min-cut algorithm. *J. ACM*, 44(4):585–591, 1997. doi:10.1145/263867.263872.

**67** René van Bevern, Hannes Moser, and Rolf Niedermeier. Approximation and Tidying - A Problem Kernel for s-Plex Cluster Vertex Deletion. *Algorithmica*, 62(3-4):930–950, 2012.

**68** Ka-Chun Wong. A Short Survey on Data Clustering Algorithms. *CoRR*, abs/1511.09123, 2015. arXiv:1511.09123.

**69** Bang Ye Wu and Jia-Fen Chen. Balancing a Complete Signed Graph by Editing Edges and Deleting Nodes. In Ruay-Shiung Chang, Lakhmi C. Jain, and Sheng-Lung Peng, editors, *Advances in Intelligent Systems and Applications - Volume 1*, pages 79–88, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

**70** Dongkuan Xu and Yingjie Tian. A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, 2(2):165–193, June 2015.