The Parameterized Complexity of Network Microaggregation

Václav Blažej¹, Robert Ganian², Dušan Knop¹, Jan Pokorný¹, Šimon Schierreich¹, Kirill Simonov³

¹Faculty of Information Technology, Czech Technical University in Prague, Prague, Czechia
²Algorithms and Complexity Group, Technische Universität Wien, Vienna, Austria
³Hasso Plattner Institute, University of Potsdam, Postdam, Germany
blavez1@fit.cvut.cz, rganian@gmail.com, {dusan.knop,pokorj54,schiesim}@fit.cvut.cz, kirill.simonov@hpi.de

Abstract

Microaggregation is a classical statistical disclosure control technique which requires the input data to be partitioned into clusters while adhering to specified size constraints. We provide novel exact algorithms and lower bounds for the task of microaggregating a given network while considering both unrestricted and connected clusters, and analyze these from the perspective of the parameterized complexity paradigm. Altogether, our results assemble a complete complexity-theoretic picture for the network microaggregation problem with respect to the most natural parameterizations of the problem, including input-specific parameters capturing the size and homogeneity of the clusters as well as the treewidth and vertex cover number of the network.

Introduction

In view of the rising importance of user anonymity and privacy, it is of paramount importance to ensure that release of data about relations in social or other networks does not lead to the disclosure of individual information, while also preserving the informational content. Such tasks lie at the heart of statistical disclosure control, and are typically tackled by creating synthetic data from the available real-world network (Das et al. 2022; Belhajjame et al. 2020; Kim, Venkatesh, and Panda 2022).

One of the most classical approaches used to create such data is microaggregation (Domingo-Ferrer 2009; Yan et al. 2022), which typically aggregates available data into small homogeneous clusters and releases the centers of these clusters instead of the original data points. While it is always desirable to have a lower bound for the size of these clusters in order to achieve anonymity, depending on the context in which microaggregation is used it may either be useful to allow for clusters of variable size (so-called data-oriented microaggregation) or to require all the clusters to have roughly the same size (referred to as fixed-size microaggregation) (Domingo-Ferrer and Mateo-Sanz 2002; Solé, Muntés-Mulero, and Nin 2012). The advantage of the former is that one can achieve more homogeneous clusters, while the latter ensures that each synthetic data point represents roughly the same amount of original data points. Indeed, without an upper bound on the cluster size, it may easily happen that two data points in the microaggregated data represent sets of highly disproportionate sizes.

While microaggregation has obvious connections to clustering, the addition of size control restrictions is not compatible with most clustering approaches which typically aim at maximizing the sizes of coherent clusters. By now, there is an extensive body of research studying typical network clustering models from an empirical (Rattigan, Maier, and Jensen 2007; Mukherjee, Sarkar, and Lin 2017) as well as complexity-theoretical perspective (Orecchia and Zhu 2014; Micha and Shah 2020). However, research on microaggregation of networks has so far focused on empirical aspects (Sun et al. 2012; Itikhar, Wang, and Lin 2020) or (to a much smaller extent) on approximation specifically in the Euclidean space (Domingo-Ferrer, Sébé, and Solanas 2008; Domingo-Ferrer and Sebé 2006), while very little was known about the precise boundaries of tractability for optimal network microaggregation. In fact, we are aware of only a single paper that touched on this topic to date (Thaeter and Reischuk 2021), albeit the related topic of lower bounded clustering has been explored in two other works (Abu-Khzam et al. 2018; Casel 2019).

While formal definitions are provided in the Preliminaries later, for the upcoming discussions it will be useful to already concretize the network microaggregation problem (hereinafter NMA). On the input, we receive a network (modeled as an edge-weighted graph \(G\)), a lower bound \(\ell\), a distance bound \(d\), and an upper bound \(u\). The task is then to partition the vertices of \(G\) into clusters such that each cluster (1) has size between \(\ell\) and \(u\), and (2) admits a center (vertex) whose distance to every vertex in the cluster is at most \(d\). We remark that, in view of the aim of anonymizing \(G\), the center need not be part of the cluster and that a center can be reused for multiple clusters. Moreover, in connection to previous research on network clustering (Deligkas et al. 2022; Macgregor and Sun 2021; van Bevern et al. 2015), in some settings it is sensible to add a third requirement which avoids the creation of completely disconnected clusters: (3) each cluster must be connected in \(G\). We denote this variant connected network mi-
CROAGGREGATION (hereinafter CNMA).

The definition in the previous paragraph is close to that considered in previous works (Abu-Khzam et al. 2018), but it also generalizes these via the inclusion of an upper bound \(u\) on the size of the cluster. Note that in the standard case of NETWORK MICROAGGREGATION a lower bound \(\ell\) immediately implies an upper bound of \(2\ell - 1\) since a cluster of size at least \(2\ell\) can be arbitrarily split in two while not increasing the distance to the center. However, in the case of CONNECTED NETWORK MICROAGGREGATION this operation is not necessarily possible, as the cluster might lose its connectivity. Therefore having only a lower bound might yield arbitrarily large clusters in this case, leading to an extremely non-informative representation of the data (see Figure 1 for an example). Moreover, even in the case of NETWORK MICROAGGREGATION, specifying an upper bound allows for a finer control of the cluster sizes, as in the fixed-size microaggregation setting.

Contribution. We initiate the comprehensive study of the complexity of both NMA and CNMA. While both problems are easily seen to be \(\text{NP}\)-complete in their full generality, in many scenarios of interest the instances we deal with have specific structural properties and the natural question that arises is whether one can exploit such properties to obtain exact algorithms with good runtime guarantees. To provide a comprehensive answer to this question, we turn to the parameterized complexity paradigm (Downey and Fellows 2013; Cygan et al. 2015) that has by now been successfully applied to a large number of problems from numerous areas of computer science, including in many settings relevant to artificial intelligence research (Kronegger et al. 2014; Ganian et al. 2020; Dvořák et al. 2021).

Essentially, in the parameterized refinement of classical complexity, one analyzes the performance of algorithms not only with respect to the input size \(n\) but also in view of one or several numerical parameters \(k\) which capture specific properties of the input. If the problem can be solved in time \(f(k) \cdot n^{O(1)}\) for some computable function \(f\), we say that it is fixed-parameter tractable (FPT). The associated algorithm is then also called fixed-parameter, and this represents the most favorable outcome for an \(\text{NP}\)-hard problem. A weaker notion of tractability is captured by the class \(\text{XP}\), which contains all problems that can be solved in polynomial time for every constant value of the parameter. Naturally, it may also happen that a problem remains \(\text{NP}\)-hard even for some constant parameter value, and similarly we may also use the established notion of \(\text{W[1]}\)-hardness to exclude fixed-parameter algorithms under a specific parameterization.

When considering natural parameterizations of NMA and CNMA, two well-motivated parameters are \(d\) (which ensures that the aggregated clusters are coherent) and \(u\) (which ensures that the clusters are small). As our main contribution, we establish the complexity of both problems under all combinations of these two parameters along with structural parameterizations of the network. There, we consider treewidth (Robertson and Seymour 1984) as the most prominent and natural structural graph parameter, as well as the vertex cover number as a restriction of treewidth that has typically been used to achieve tractability for problems which are intractable w.r.t. treewidth. Our main results are summarized in Table 1; in the rest of this section, we provide an overview and context for the individual results.

As an initial starting point, we observe that NMA as well as CNMA remain \(\text{NP}\)-hard even for fixed values of \(d\) and \(u\), which means that to achieve tractability one needs to include at least some parameterization of the network (that is why Table 1 only considers combinations of \(d\) and \(u\) with network parameterizations; it is worth noting that the same considerations also apply to \(\ell\), since \(\ell \leq u\)). Next, when using the treewidth \(\text{tw}\) of the network (as our baseline structural parameter), we show that (a) NMA is \(\text{XP}\) parameterized by \(\text{tw} + d\), (b) CNMA is \(\text{XP}\) parameterized by \(\text{tw}\) alone, and (c) CNMA is \(\text{FPT}\) parameterized by \(\text{tw} + d + u\). We obtain these by designing two new algorithms—one for each problem variant—where results (b) and (c) follow as a corollary of the same algorithm for CNMA. These algorithms utilize the dynamic programming technique employed by virtually all treewidth-based algorithms, but are non-trivial and especially result (a) required the introduction of an advanced cluster-grouping step. We remark that none of these algorithms seem to be obtainable via known algorithmic meta-theorems for treewidth such as Courcelle’s Theorem (Courcelle 1990) or MSOL Partitioning (Rao 2007).

Next, we turn to the question of whether one can obtain more favorable tractability results by exploiting the vertex cover number \(\text{vc}\) of the network. In this setting, we apply a range of techniques to establish the remaining tractability results depicted in Table 1. We begin with a trivial observation that CNMA is \(\text{FPT}\) parameterized by \(\text{vc} + u\). To obtain fixed-parameter tractability w.r.t. the same parameterization for NMA, we make use of a new structural observation which guarantees that every sufficiently large YES-instance of the problem must admit a solution with “homogeneous” clusters, which in turn allows us to apply the classical kernelization technique (Cygan et al. 2015); for a similar approach see, e.g., (Mnich and Wiese 2015). To establish the fixed-parameter tractability of both problems...
w.r.t. vc + d, we then use a different tool, notably exhaustive branching in combination with the construction of an ILP encoding with a bounded number of variables which can then be solved by Lenstra’s celebrated result (Lenstra Jr. 1983) and its subsequent improvements (Kannan 1987; Frank and Tardos 1987). Interestingly, for our final result—the XP-tractability of NMA when parameterized by vc—we need to combine both the techniques mentioned above: in particular, we show that every sufficiently large YES-instance admits a solution which is “well-structured” (albeit in a different way than we required for fixed-parameter tractability), and then use this fact to design an ILP which captures the problem.

Finally, we complement all of the aforementioned algorithms with lower bounds that justify the presence of every parameter in each of the algorithms; in other words, the classification arising from these algorithms is tight. First, we observe that the known intractability of EQUIVALENT CONNECTED PARTITION parameterized by tw almost immediately yields the intractability of CNMA w.r.t. tw + d. Next, we establish W[1]-hardness of both problems parameterized by vc alone, as well as the W[1]-hardness of CNMA parameterized by tw + u using two reductions from the classical INDEPENDENT SET and MULTICOLORED CLIQUE problems; the main difficult here is that it was necessary to encode the structure of the input graph via distances in the network. For the last two open cases, notably NMA parameterized by tw + u and by tw + u + d, it is far from obvious how one could reduce from the usual problems used to establish intractability: indeed, the intractability of these variants stems from the possible presence of many “incomplete” clusters which need to be paired up with each other. Hence, to establish our final lower bounds, we use a more recent reduction technique and start from the MULTIDIMENSIONAL SUBSET SUM problem (Ganian, Klute, and Ordyniak 2021).

### Table 1: Our results—the full complexity-theoretic landscape of (CONNECTED) NETWORK MICROAGGREGATION under all combinations of considered input-specified parameters (rows) and structural parameters (columns); NP-h means that the problem remains NP-hard even for a fixed value of the parameters.

<table>
<thead>
<tr>
<th></th>
<th>NMA</th>
<th>CNMA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>tw</td>
<td>vc</td>
</tr>
<tr>
<td>—</td>
<td>NP-h</td>
<td>W[1]-h (Thm. 13), XP (Thm. 9)</td>
</tr>
<tr>
<td>d</td>
<td>W[1]-h, XP (Thm. 1)</td>
<td>FPT (Thm. 8)</td>
</tr>
<tr>
<td>u</td>
<td>NP-h (Thm. 12)</td>
<td>FPT (Thm. 5)</td>
</tr>
<tr>
<td>d + u</td>
<td>W[1]-h (Thm. 11), XP</td>
<td>FPT</td>
</tr>
</tbody>
</table>

**Related Work.** The general concept of k-anonymity, i.e., replacing a dataset with a similar one but where each entry is indistinguishable from at least k − 1 others, was introduced by Sweeney (2002). Aggarwal et al. (2010) proposed the specific model of k-anonymity where the entries are represented by their respective centers in a clustering. In that work, the particular optimization problem is called r-GATHER; the difference to NETWORK MICROAGGREGATION is only that r-GATHER does not specify an upper bound on the size of the cluster, and the authors showed that r-GATHER is NP-complete, but admits a 2-approximation. In the Euclidean space with the sum-of-squares clustering objective, it is known that an analogue of r-GATHER admits a O(r^3)-factor approximation (Domínguez-Ferrer, Sebé, and Solanas 2008), and a 2-approximation for r = 2 (Domínguez-Ferrer and Sebé 2006). An overview of various versions of r-GATHER on networks was conducted (Abu-Khzam et al. 2018). The authors of that work considered several variants of inter- and intra-cluster distance measure, and showed a number of approximation algorithms and NP-hardness results.

The NETWORK MICROAGGREGATION problem is related to the well-studied k-Center clustering problem, where clusters have no size restrictions, but instead the number of clusters k is predetermined. In particular, Feldmann and Marx (2020) studied parameterized algorithms and complexity of k-Center clustering on networks for various structural parameters of the network. A number of recent works studied parameterized algorithms for clustering with size constraints on the clusters, among others with k-Median as the optimization objective (Cohen-Addad and Li 2019; Bandyapadhyay, Fomin, and Simonov 2021). It is worth to note that all of the results on k-Center/k-Median mentioned above offer little insight in our setting, as they heavily exploit small number of clusters. Last but not least, we mention that there is a large body of work exploring the computational complexity of data clustering in various settings, e.g., when data is missing (Ganian et al. 2020, 2022).

**Preliminaries**

We use bold-face letters (e.g., \( x \)) for vectors and normal font for their entries (i.e., \( x_2 \) is the second entry of vector \( x \)). For a positive integer \( n \) we denote by \([n]\) the set of all positive integers up to (and including) \( n \), that is, \([n] = \{1, 2, \ldots, n\}\), and let \([n]_0 = [n] \cup \{0\}\).

**Network Microaggregation.** Since the networks in our problems are typically modeled as graphs, we employ basic graph-theoretic terminology (Matoušek and Nešetřil 2009; Diestel 2012). A graph \( G = (V, E) \) has a vertex set \( V \) and an edge set \( E \). Most graphs in this paper are edge-weighted; we treat the edge-weight \( \omega(e) \) of an edge \( e \) as its length. For two vertices \( u \) and \( v \) the distance \( \text{dist}_G(u, v) \) is the length of
exists). Our problem of interest is:

**Network Microaggregation (NMA)**

**Input:** An undirected \( n \)-vertex graph \( G = (V, E) \), a lower bound \( \ell \in \mathbb{N} \), an upper bound \( u \in \mathbb{N} \), a maximum allowed distance to a cluster center \( d \in \mathbb{N} \) and a length function \( \omega: E \to [d] \).

**Question:** Is there an integer \( m \) and a partition \( \Pi = (C_1, \ldots, C_m) \) of \( V \) together with a list of vertices \( C = (c_1, \ldots, c_m) \) such that:

- \( \forall i \in [m]: \ell \leq |C_i| \leq u \), \( \forall v \in C_i: \text{dist}_G(v, c_i) \leq d \)?

The **Connected Network Microaggregation** problem (CNMA) is defined analogously, but with the additional requirements that the subgraph induced by each cluster \( C \in \Pi \) is connected. We mention that while (C)NMA are formulated as decision problems for complexity-theoretic reasons, all our algorithms are constructive and can also output a microaggregation \( \Pi \) as a witness.

The most crucial parts of the input, apart from the structure of the given network, are arguably those that determine the quality of the considered microaggregation. Clearly, this includes the parameter \( d \) as this one directly governs the quality of the microaggregated data (the closer the centers are to the original datapoints, the higher the quality of the data collected by the algorithm). Furthermore, by the parameter \( u \) we can directly affect the number of datapoints in the aggregated data. Therefore, in many settings it would be desirable to keep the value of both \( d \) and \( u \) rather low. In fact, the same applies to \( \ell \) as this governs the "proportionality" of the collected data. We remark that both NMA and CNMA are \( \mathbf{NP} \)-complete even with respect to treewidth and \( \mathbf{XP} \) when \( |C_i| \leq 2 \) for all nodes in the tree decomposition.

### Structural Parameters.

Let \( G = (V, E) \) be a graph. A set \( M \subseteq V \) is a vertex cover of \( G \) if \( M \cap e \neq \emptyset \) for every \( e \in E \). The **vertex cover number** of \( G \), denoted \( \text{vc} \), is the minimum size of a vertex cover of \( G \).

A **nice tree-decomposition** \( TD \) of an undirected graph \( G = (V, E) \) is a pair \((\text{Tree} = (W, F), \{X^v \mid v \in W\})\), where \( \text{Tree} \) is a tree rooted at a node \( r \in W \) and a **bag** \( X^r \subseteq V \) is a set of vertices of \( G \) associated with node \( x \) such that:

- For every \( vw \in E \) there is a node \( x \) for which \( v, w \in X^x \).
- For every vertex \( v \in V \), the set of nodes \( x \) satisfying \( v \in X^x \) forms a subtree of \( \text{Tree} \).
- \(|X^x| = 0\) for every leaf \( x \) of \( \text{Tree} \) and \(|X^r| = 0\).
- There are only three kinds of non-leaf nodes in \( \text{Tree} \):
  - **Introduce node:** a node \( x \) with exactly one child \( y \) such that \( X^x = X^y \cup \{v\} \) for some vertex \( v \in X^y \).
  - **Forget node:** a node \( x \) with exactly one child \( y \) such that \( X^x = X^y \setminus \{v\} \) for some vertex \( v \in X^y \).
  - **Join node:** a node \( x \) with two children \( y \) and \( z \) such that \( X^x = X^y = X^z \).

The **width** of a nice tree-decomposition \((\text{Tree}, \{X^x \mid x \in V(\text{Tree})\})\) is \( \max_{x \in V(\text{Tree})}(|X^x| - 1) \), and the **treewidth** of the graph \( G \), denoted \( tw \), is the minimum width of a nice tree-decomposition of \( G \). Let \( x \) be a node of a nice tree decomposition. We denote by \( V^x \) the set of all past vertices contained in the subtree rooted at \( x \), while vertices outside of \( V^x \) are said to be in the future. Formally, we have \( V^x = \bigcup_{y \text{ predecessor of } x} X^y \).

It is worth noting that the structure of real-world data has been demonstrated to attain low treewidth in several settings (Maniu, Senellart, and Jog 2019). Moreover, it is well known (and easy to see) that \( vc \geq tw \).

### Algorithms for Tree-Like Networks

In this section, we provide dynamic programming algorithms for NMA and CNMA that establish the tractable fragments of these problems when treewidth is used as a parameter. We remark that while the use of dynamic programming that relies on bags acting as separators in the graph is the "golden standard" for treewidth-based algorithms, the technical details here are far from standard. Among others, in order to obtain dynamic programming tables which are succinct enough for our purposes, we needed to identify a suitable notion of vertex types that capture their properties when used as a center and incorporate these into the tables.

As the types used in this section will crucially depend on vertex distances, it will be useful to introduce some additional terminology to capture this. As we will not need to distinguish distances longer than \( d \), let \( \text{dist}_d(v, w) = \text{dist}(v, w) \) if \( \text{dist}(v, w) \leq d \) and \( \infty \) otherwise. Our operations over distances are additive, assume the result is set to \( \infty \) whenever the value exceeds \( d \).

For each node \( x \) in a tree-decomposition, we can now partition the vertices of \( G \) into **center-types** that are based on their membership to \( V^x \) and their distances to \( X^x \). Formally, let the **center-type** of a vertex \( v \) for \( v \in V(G) \) be

\[
t^x,v = (\text{dist}_d(v, w_1), \text{dist}_d(v, w_2), \ldots, \text{dist}_d(v, w_{|X^x|})),
\]

where \( w_i \) are the vertices of \( X^x \). A core ingredient for the proof is that if two vertices have the same type and are both in the past or both in the future, then they are "indistinguishable" from the viewpoint of the bag. We remark that the notion of center-types is related to the well-studied **metric dimension** problem (Bonnet and Purohit 2021).

Both of the presented algorithm in this section proceed by leaf-to-root dynamic programming along a nice tree-decomposition computed using well-known algorithms (Bodlaender 1996; Korhonen 2021). We begin by establishing the XP-tractability of **Network Microaggregation** with respect to treewidth and \( d \) which, while still involving some technical challenges, is the easier of the two results in this section.

**Theorem 1.** **Network Microaggregation can be solved in time** \( n^{O((d+2)^{tw})} \).
Proof Sketch. The algorithm stores binary records at each node $x$ in the following form: $D[x, f^x, p^x] \in \{T, F\}$, where:

- $f^x : T^x \rightarrow [\mathbb{N}]_0$ is the total number of vertices in clusters in $V^x \setminus X^x$ that have a center with center-type $T^x$ that occurs in the future,
- $p^x : T^x \rightarrow [\mathbb{N}]_0 \times [\mathbb{N}]_0$ is the lower and upper bound on the required number of vertices that are expected to be added to clusters that have a center with respective center-type in the past.

Intuitively, the records hold aggregated information about clusters with centers of the same type. Vertices within clusters that have future centers with the same center-type are interchangeable so we just need their amount. Partially formed clusters that have centers in the past require some number of additional vertices to have the correct size. We can aggregate those clusters that have the same past center-type because it will not matter which one a new vertex satisfies; the total demand still decreases by one. Through the computation, we immediately resolve aggregated clusters with center-type $t = \infty \times \mathbb{N}$ since no more vertices may be added to these.

The semantics of the records are as follows: a record is true if and only if the subinstance induced on $X^x$ admits a partial solution satisfying the conditions given by $f$ and $p$.

Clearly, the number of records is upper-bounded by $n^{O((d+2)^w)}$. Moreover, the records can be trivially determined at each leaf node, and once we obtain the records for the root node $r$ of the tree-decomposition, we can correctly answer “Yes” if and only if $D[r, \emptyset, \emptyset] = T$ since $X^r = \emptyset$.

Hence, to complete the proof it suffices to show that the records can be computed in a leaf-to-root fashion along the nodes of the tree-decomposition. \qed

Next, we show that for Connected Network Microaggregation one can also use treewidth to achieve tractability, albeit with an entirely different dynamic programming algorithm.

Theorem 2. Connected Network Microaggregation can be solved in time $O(n^d) + d^{O(tw^2)} n^{O(tw)}$.

Corollary 3. Connected Network Microaggregation is in XP parameterized by tw only, and is fixed-parameter tractable when parameterized by $tw + u + d$.

Algorithms Using the Vertex Cover Number

In this section, we investigate the complexity of both considered problems with respect to the vertex cover number. It is well known that the vertex cover can be computed by an efficient fixed-parameter algorithm, and hence throughout this section we assume that $G$ already comes equipped with a minimum vertex cover $M$ of size $vc$.

The results of this section are obtained via the application of two distinct techniques, and to streamline the presentation we divide the section into two subsections accordingly. While the techniques differ, an idea that will be crucial in both subsections is that even though the number of vertices we need to deal with may be large, one can group them into equivalence classes based on a suitable notion of “type”. Interestingly, how these types need to be set up in order for the algorithms to work differs from scenario to scenario (and also differs from the previous section). Moreover, in some cases it will be necessary to define not only types for vertices, but also for clusters.

Tractability via Kernelization

Kernelization is a procedure that transforms the instance into an equivalent one whose size is bounded by the parameter value (where the reduced instance can then be solved, e.g., by brute force or any heuristic), and is the core technique used in this subsection. Kernelization is required to run in polynomial time and is typically achieved by using problem-specific reduction rules. For each such rule, one typically needs to establish that it is “safe”, meaning that it preserves Yes and No instances.

We begin with a simple observation for Connected Network Microaggregation: every Yes-instance of Connected Network Microaggregation may only contain at most $vc \cdot u$ vertices.

Observation 4. Connected Network Microaggregation is fixed-parameter tractable parameterized by the vertex-cover number $vc$ plus the upper bound $u$.

Next, we turn our attention to Network Microaggregation with the same parameterization. Before we introduce the reduction rule, we need a few definitions. We define the neighborhood-type of a vertex $v$ for $v \in V \setminus M$, where $M$ is a vertex cover of $G$. Let the neighborhood-type $t^v$ of a vertex be the set of its neighbors, i.e., its open neighborhood. For a cluster $C \subseteq V \setminus M$ we define its cluster-multiplicity $k(C)$ to be the multiset $\{t^w \mid w \in C\}$, i.e., a multiset of neighborhood-types of its vertices. We stress that cluster-multatypes are defined only for those clusters that do not contain any vertices of $M$. We call a cluster $C$ homogeneous if $t^v = t^w$ for all $v, w \in C$, and heterogeneous otherwise. By proving that every Yes-instance also admits a “nice” solution where each heterogeneous cluster-multitype occurs only at most $\ell$ many times, we show:

Reduction Rule 1. Suppose there are at least $u \cdot (2^{vc \cdot u} + \ell \cdot u + vc)$ vertices of the same neighborhood-type. Then, remove $\ell$ such vertices.

The exhaustive application of Reduction Rule 1 then suffices to obtain a kernel of size at most $2^{O(vc \cdot u)}$, and hence:

Theorem 5. Network Microaggregation is fixed-parameter tractable when parameterized by $vc + u$.

Algorithms Based on Branching and ILP

Our next set of algorithms combines exhaustive branching with an encoding into an Integer Linear Program (ILP) where the number of variables is bounded by the parameters; such ILPs are known to be fixed-parameter tractable:

Proposition 6 (Lenstra Jr. 1983; Kannan 1987; Frank and Tardos 1987). There is an algorithm that solves an input ILP instance $I$ with $p$ variables in time $p^{O(p)} \cdot |I|$. 
We start by defining the types of vertices outside the vertex cover and the types of clusters, both of which are based on the distances to the vertices in \( M \).

**Definition 1.** Let \( M = \{ v_1, v_2, \ldots, v_{vc} \} \) be a vertex cover of the graph \( G = (V, E) \) and let \( v \in V \setminus M \). The vertex-type \( t^v \in \{ [d] \cup \{ \infty \} \}^{M} \) of \( v \) is

\[
t^v = \begin{cases} 
\omega(v, v_i) & \text{if } \{ v, v_i \} \in E, \\
\infty & \text{otherwise}.
\end{cases}
\]

By \( T \) we denote the set of all vertex-types of vertices.

**Definition 2.** Let \( C \subseteq V \) be a cluster. The cluster-type \( t(C) \subseteq T \) of \( C \) is defined as \( t(C) = \{ t \in T \mid \exists v \in C : t^v = t \} \). Moreover, let \( e \in E \cup M \). We call \(( t(C), e)\) the extended-cluster-type of cluster \( C \) with center \( c \).

We observe that there are \( |T| \leq (d + 1)^{vc} \) different vertex-types, plus additional \( vc \) vertices in the vertex cover that are dealt with separately. Moreover, the number of different cluster-types is at most \( 2^{vc} + |T| = 2^{vc} + (d + 1)^{vc} \). When also considering the centers, we get at most \( 2^{d^{vc}} \cdot (vc + (d + 1)^{vc}) \) possible extended-cluster types.

**Theorem 7.** Connected Network Microaggregation is fixed-parameter tractable parameterized by the vertex-cover number \( vc \) and the maximum distance \( d \).

**Proof Sketch.** We begin by observing that the number of clusters can be upper bounded by \( vc \). We then branch to determine the number \( m \in [vc] \) of clusters in the solution, their extended-cluster-types, and which vertex cover vertices they contain. In each branch, we perform a set of basic checks to prune out invalid choices (such as when the extended-cluster-types of clusters cannot correspond to a connected cluster). Then we construct an ILP with at most \( |T| \cdot vc \) variables that is a \( \text{Yes}\)-instance if and only if there is a solution that corresponds to the given branch.

Next, we turn to Network Microaggregation. The following algorithm is based on similar ideas as the previous Theorem, but there is an additional complication: the number of clusters is no longer upper bounded by \( vc \). Hence, we use variables in the ILP formulation to capture how often each of the extended-cluster-types occurs.

**Theorem 8.** Network Microaggregation is fixed-parameter tractable parameterized by \( vc + d \).

The final and the most complicated algorithm in this section combines both of the techniques: we first show that there is a solution with certain properties (as was done for Reduction Rule 1), and we then use this to design an ILP formulation as in the previous two algorithms.

**Theorem 9.** Network Microaggregation is in XP parameterized by the vertex-cover number \( vc \).

**Proof Sketch.** As the core ingredient in the proof, we show that every \( \text{Yes}\)-instance also admits a solution with at most \( 2^{O(vc)} \) cluster centers not in \( M \); the proof of this claim is non-trivial and involves a swapping argument on an auxiliary graph representation of the solution. Once we establish the claim, we use exhaustive branching to determine the exact centers which lie outside of \( M \), and at that point we are at a stage where the existence of a solution can be checked by a non-trivial ILP with boundedly-many variables which also relies on a suitable partitioning of vertices into types.

**Lower Bounds**

This section contains proofs for all lower bounds (i.e., hardness results) required to complete Table 1. Before proceeding to the more interesting results, we first provide a straightforward reduction from Equitable Connected Partition—a problem which is well-known to be \( \text{W}[1]\)-hard when parameterized by treewidth (Enciso et al. 2009).

**Theorem 10.** Connected Network Microaggregation is \( \text{W}[1]\)-hard parameterized by \( tw \), even if \( d = 1 \).

**Proof Sketch.** We reduce from \( \text{MSS} \) to Connected Network Microaggregation by constructing a weighted graph \( G \) for which there is a \( \text{MSS}\)-solution (if and only if there is a solution of the \( \text{W}[1]\)-hard Connected Network Microaggregation problem. Assume, for the sake of the argument, that we would be able to use arbitrarily large clusters, and set \( l = u - 1 \). In this case, we could model each
may not exclude that the active vertices of $A$ depicted in Figure 3. 

To guarantee equality, we need to ensure that $|A_j| \geq c_j$ for every $j \in \{\dim\}$, where $A_j$ corresponds with $i \in S$ in the MSS. Every inactive vertex in $P$ of the choice gadget connects to a center of its auxiliary star with $\ell - 1$ leaves. This allows clusters that cover the auxiliary stars to optionally cover the auxiliary star with it as the center and $\ell - 2$ leaves. Setting the edge lengths to be $d - 1$ ensures that each such star must be covered by a separate cluster. We connect the root of $D_j$ to $A_j$ for every $i \in [u]$ making it possible to add active vertices of $A_j$ to the clusters that cover the tree. The gadget $D_j$ requires that at least $c_j$ of the connected vertices are active.

Because of particular lengths of the construction one may not exclude that the active vertices of $A_j$ are also covered by clusters that do not cover the stars in the demand gadget. To guarantee equality, we need to ensure that $\sum_{i \in \text{active}} v_i \leq c_j$ for every $j \in \{\dim\}$ by extending the construction. This would complete the construction if we did not need to preserve a bound on the cluster sizes. To deal with this final obstacle, we replace the gadget by a different one depicted in Figure 3. 

The next reduction from Multidimensional Subset Sum is also based on the idea of using choice and duplication gadgets to capture the selection of vectors in a solution. However, since the treewidth is bounded by a constant, we cannot use $\dim$ distinct demand gadgets to encode the coordinates of each vector; instead, the coordinates are encoded via different edge lengths of their connections.

**Theorem 12.** Network Microaggregation is NP-hard even if $tw$ and $u$ are fixed constants.

**Closing the Final Gaps**

To finalize the complexity picture of NMA and CNMA, we provide two W[1]-hardness reductions. Both reductions are from a multicolored variant of some well-known combinatorial problem. In particular, we reduce from Multicolored Independent Set and Multicolored Clique, respectively.

**Theorem 13.** Network Microaggregation is W[1]-hard parameterized by the vertex-cover number $vc$.

We note that the same construction as the one used above also works for the connected variant of the problem.

**Corollary 14.** Connected Network Microaggregation is W[1]-hard parameterized by the vertex-cover number $vc$.

Our final result establishes the W[1]-hardness of CNMA even when parameterized by treewidth and the upper bound on the cluster size, which finalizes the complexity picture of the studied problems.

**Theorem 15.** Connected Network Microaggregation is W[1]-hard parameterized by $tw + u$.

**Conclusions**

Our algorithms and lower bounds shed light on the non-obvious complexity-theoretic behavior of the microaggregation problem and provide a comprehensive picture for both NMA and CNMA. It is important to note that—as is typical for complexity-theoretic studies of problems central to AI research (Bentert et al. 2021; Grötschel, Komusiewicz, and Morawietz 2021; Dvořák et al. 2021)—the algorithms provided here are not likely to outperform existing heuristics in practically relevant settings. That being said, the techniques introduced here may in the future serve as inspiration for improvements in practice.

Our study does not consider the lower bound $\ell$ as a separate parameter since many of the considerations and results obtained when parameterizing by $u$ immediately carry over to $\ell$ as well; in particular, all algorithmic lower bounds parameterized by $u$ can be directly carried over to $\ell$. Moreover, since it is known that $u$ can be assumed to be at most $2\ell$ in NMA, the two parameters are asymptotically equivalent for this problem. Last but not least, it may be interesting to investigate the complexity of the considered problems under less standard parameterizations, such as the cluster vertex deletion number.
Acknowledgments
The work was performed while VB, JP, and ŠS were visiting Algorithms and Complexity Group, TU Wien. The visit was financially supported by project 92p1 of the AKTION Austria - Czech Republic Program.

VB, DK, JP, and ŠS was supported by the Czech Science Foundation Grant No. 22-19557S. JP and ŠS also acknowledges support from the Grant Agency of the Czech Technical University in Prague funded grant No. SGS23/205/OHK3/3T/18. RG acknowledges support from the Austrian Science Foundation (FWF, project Y1329) and OeAD bilateral Czech-Austrian WTZ-funding Programme, projects No. CZ 05/2021 and 8J21AT02. KS acknowledges support by DFG Research Group ADYN under grant DFG 411362735.

References


