Clustering with Local Restrictions

Daniel Lokshtanov* and Dániel Marx**

Abstract. We study a family of graph clustering problems where each cluster has to satisfy a certain local requirement. Formally, let μ be a function on the subsets of vertices of a graph G. In the (μ, p, q)-Partition problem, the task is to find a partition of the vertices into clusters where each cluster C satisfies the requirements that (1) at most q edges leave C and (2) $\mu(C) \leq p$. Our first result shows that if μ is an *arbitrary* polynomial-time computable monotone function, then (μ, p, q)-Partition can be solved in time $n^{O(q)}$, i.e., it is polynomial-time solvable *for every fixed* q. We study in detail three concrete functions μ (number of nonedges in the cluster, maximum number of non-neighbours a vertex has in the cluster, the number of vertices in the cluster), which correspond to natural clustering problems. For these functions, we show that (μ, p, q)-Partition can be solved in time $2^{O(p)} \cdot n^{O(1)}$ and in randomized time $2^{O(q)} \cdot n^{O(1)}$, i.e., the problem is fixed-parameter tractable parameterized by p or by q.

1 Introduction

Partitioning objects into clusters or similarity classes is an important task in various applications such as data mining, facility location, interpreting experimental data, VLSI design, and many more. The partition has to satisfy certain constraints: typically, we want to ensure that objects in a cluster are "close" or "similar" to each other and/or objects in different clusters are "far" or "dissimilar." Additionally, we may want to partition the data into a certain prescribed number k of clusters, or we may have upper/lower bounds on the size of the clusters. Different objectives and different distance/similarity measures give rise to specific combinatorial problems.

Correlation clustering [14, 1, 3, 15] deals with a specific form of similarity measure: for each pair of objects, we know that either they are similar or dissimilar. This means that the similarity information can be expressed as an undirected graph, where the vertices represent the objects and similar objects are adjacent. In the ideal situation every connected component of the graph is a clique, in which case the components form a clustering that completely agrees with the similarity information. However, due to inconsistencies in the data or experimental errors, such a perfect partitioning might not always be possible. The goal in correlation clustering is to partition the vertices into an arbitrary number of

* University of California, San Diego, USA. dlokshtanov@cs.ucsd.edu

** Humboldt-Universität zu Berlin, Berlin, Germany. dmarx@cs.bme.hu. Research supported by the Alexander von Humboldt Foundation and OTKA grant 67651.

clusters in a way that agrees with the similarity information as much as possible: we want to minimize the number of pairs for which the clustering disagrees with the input data (i.e., similar pairs that are put into different clusters, or dissimilar pairs that are clustered together).

In many cases, such as in variants of the correlation clustering problem defined in the previous paragraph, the objective is to minimize the total error of the solution. Thus the goal is to find a solution that is good in a global sense, but this does not rule out the possibility that the solution contains clusters that are very bad. In this paper, the opposite approach is taken: we want to find a partition where each cluster is "good" in a certain local sense. This means that the partition has to satisfy a set of local constraints on each cluster, but we do not try to optimize the total fitness of clusters.

The setting in this paper is the following. We want to partition the graph into an arbitrary number of clusters such that (1) at most q edges leave each cluster, and (2) each cluster induces a graph that is "cluster-like." Defining what we mean by the abstract notion of cluster-like gives rise to a family of concrete problems. Formally, let μ be a function that assigns a nonnegative integer to each subset of vertices in the graph and let us require $\mu(X) \leq p$ for every cluster X of the partition. There are many reasonable choices for the measure μ that correspond to natural problems. In particular, in this paper we will obtain concrete results for the following three measures:

1. $\mathsf{nonedge}(X)$: number of nonedges induced by X,
2. $\mathsf{nondeg}(X)$: maximum degree of the *complement* of the graph induced by X.
3. $\mathsf{size}(X) = |X|$: number of vertices of X.

The first two functions express that each cluster should induce a graph that is close to being a clique. The third function only requires that each cluster is small. For a given function μ and integers p and q, we denote by (μ, p, q)-Partition the problem of partitioning the vertices into clusters such that at most q edges leave each cluster and $\mu(X) \leq p$ for every cluster.

Our first result is very simple yet powerful. Let μ be a function satisfying the mild technical conditions that it is polynomial-time computable and monotone (i.e., if $X \subseteq Y$, then $\mu(X) \leq \mu(Y)$). Observe that for example all three functions defined above satisfy these conditions. Our first result shows that for *every function* μ satisfying these conditions and *every fixed integer* q, the problem (μ, p, q)-Partition can be solved in polynomial time (the value p is considered to be part of the input). For example, it can be decided in polynomial time if there is a clustering where at most 13 edges leave each cluster and each cluster induces at most 27 nonedges (or even the more general question, where the maximum number p of nonedges is given in the input). This might be surprising: we believe that most people would guess that this problem is NP-hard. The algorithm is based on a simple application of uncrossing of posimodular functions and on the fact that for fixed q we can enumerate every (connected) cluster with at most q outgoing edges. The crucial observation is that if every vertex can be *covered* by a good cluster, then the vertices can be *partitioned* into good clusters.

Thus the problem boils down to checking if a given v is contained in a suitable cluster.

While the algorithm is simple in hindsight, considerable efforts have been spent on solving some very particular special cases. For example, Heggernes et al. [9] gave a polynomial-time algorithm for $(\mathsf{nonedge}, 1, 3)$-Partition and Langston and Plaut [10] argued that the very deep results of Robertson and Seymour on graph minors and immersions imply that (size, p, q)-Partition is polynomial-time solvable for every fixed p and q. These results follow as straightforward corollaries from our first result.

Although this simple algorithm is polynomial for every fixed q, the running time is about $n^{O(q)}$, thus it is not efficient even for small values of q. To improve the running time, we look at the problem from the viewpoint of parameterized complexity. We show that for several natural measures μ, including the three defined above, the clustering problem can be solved in randomized time $2^{O(q)} \cdot n^{O(1)}$, that is, the problem is fixed-parameter tractable (FPT) parameterized by the bound q on the number of edges leaving a cluster. Moreover, the bound p can be assumed to be part of the input. Thus this algorithm can be efficient for small values of q (say, $O(\log n)$) even if p is large. The algorithm has constant probability of error, but it can be derandomized at the cost of worse dependence on q in the running time (details will appear in the full version). The problem (size, p, q)-Partition appears in the open problem list of the 1999 monograph of Downey and Fellows [8] under the name "Minimum Degree Partition," where it is suggested that the problem is probably W[1]-hard parameterized by q. Our result answers this question by showing that the problem is FPT, contrary to the expectation of Downey and Fellows.

A crucial ingredient of our parameterized algorithm is the notion of *important separators,* which has been used (implicitly or explicitly) to obtain fixed-parameter tractability results for various cut or separator related problems. In particular, we use the "randomized selection of important sets" argument that was introduced very recently in [13] to prove the fixed-parameter tractability of (edge and vertex) multicut. With these tools at hand, we can reduce (μ, p, q)-Partition to a special case that we call the "Satellite Problem." We show that if the Satellite Problem is fixed-parameter tractable parameterized by q for a particular function μ, then (μ, p, q)-Partition is also fixed-parameter tractable parameterized by q. It seems that for many reasonable functions μ, the Satellite Problem can be solved by dynamic programming techniques. In particular, this is true for the three functions defined above, and this results in randomized algorithms with running time $2^{O(q)} \cdot n^{O(1)}$. Note that the reduction to the Satellite Problem works for every monotone μ, and we need arguments specific to a particular μ only in the algorithms for Satellite Problem.

2 Clustering and uncrossing

Given an undirected graph G, we denote by $\Delta(X)$ the set of edges between X and $V(G) \setminus X$, and define $d(X) = |\Delta(X)|$. We will use two well-known and

easily checkable properties of the function d: for $X, Y \subseteq V(G)$, d satisfies the *submodular* and *posimodular* inequalities

$$d(X) + d(Y) \geq d(X \cap Y) + d(Y \cup X) \text{ and } d(X) + d(Y) \geq d(X \setminus Y) + d(Y \setminus X).$$

Let $\mu : 2^{V(G)} \to \mathbb{Z}^+$ be a function assigning nonegative integers to sets of vertices of G. Let p and q be two integers. We say that a set $C \subseteq V(G)$ is a (μ, p, q)-*cluster* if $\mu(C) \leq p$ and $d(C) \leq q$. A (μ, p, q)-*partition* of G is a partition of $V(G)$ into (μ, p, q)-clusters. The main problem considered in this paper is finding such a partition. A necessary condition for the existence of (μ, p, q)-partition is that for every vertex $v \in V(G)$ there is a (μ, p, q)-cluster that contains v. Therefore, we are also interested in the problem of finding a cluster containing a vertex v.

(μ, p, q)-PARTITION
Input: A graph G, integers p, q.
Find: A (μ, p, q)-partition of G.

(μ, p, q)-CLUSTER
Input: Graph G, integers p, q, vertex v.
Find: A (μ, p, q)-cluster C containing v.

The main observation of this section is that if μ is *monotone* (i.e., $\mu(X) \leq \mu(Y)$ for every $X \subseteq Y$), then this is actually a sufficient condition. Therefore, in these cases, it is sufficient to solve (μ, p, q)-CLUSTER.

Lemma 1. *Let G be a graph, let $p, q \geq 0$ be two integers, and let $\mu : 2^{V(G)} \to \mathbb{Z}^+$ be a monotone function. If every $v \in V(G)$ is contained in some (μ, p, q)-cluster, then G has a (μ, p, q)-partition, and given a set of (μ, p, q)-clusters $C_1, \ldots, C_n$ whose union is $V(G)$, a (μ, p, q)-partition can be found in polynomial time.*

Proof. Let us consider a collection $C_1, \ldots, C_n$ of (μ, p, q)-clusters whose union is $V(G)$. If the sets are pairwise disjoint, then they form a partition of $V(G)$ and we are done. If $C_i \subseteq C_j$, then the union remains $V(G)$ even after throwing away C_i. Thus we can assume that no set is contained in another. Suppose that C_i and C_j intersect. Now either $d(C_i) \geq d(C_i \setminus C_j)$ or $d(C_j) \geq d(C_j \setminus C_i)$ must be true: it is not possible that both $d(C_i) < d(C_i \setminus C_j)$ and $d(C_j) < d(C_j \setminus C_i)$ hold, as this would violate the posimodularity of d. Suppose that $d(C_j) \geq d(C_j \setminus C_i)$. Now the set $C_j \setminus C_i$ is also a (μ, p, q)-cluster: we have $d(C_j \setminus C_i) \leq d(C_j) \leq q$ by assumption and $\mu(C_j \setminus C_i) \leq \mu(C_j) \leq p$ from the monotonicity of μ. Thus we can replace C_j by $C_j \setminus C_i$ in the collection: the union of the clusters is still $V(G)$. Similarly, if $d(C_j) \geq d(C_j \setminus C_i)$, then we can replace C_j by $C_j \setminus C_i$.

Repeating these steps (throwing away subsets and resolving intersections), we eventually arrive at a pairwise disjoint collection of (μ, p, q)-clusters. Each step decreases the number of cluster pairs C_i, C_j that have non-empty intersection. Therefore, this process terminates after a polynomial number of steps. □

In light of Lemma 1, it is sufficient to find a (μ, p, q)-cluster C_v for each vertex $v \in V(G)$. If there is a vertex v for which there is no such cluster C_v, then obviously there is no (μ, p, q)-partition; if we have such a C_v for every vertex

v, then Lemma 1 gives us a (μ, p, q)-partition in polynomial time. For fixed q, (μ, p, q)-Cluster can be solved by brute force if μ is polynomial-time computable:enumerate every set F of at most q edges and check if the component of $G \setminus F$ containing v is a (μ, p, q)-cluster. If C_v is a (μ, p, q)-cluster containing v, then we find it when $F = \Delta(C_v)$ is considered by the enumeration procedure.

Theorem 2. *Let μ be a polynomial-time computable monotone function. Then for every fixed q, there is an $n^{O(q)}$ time algorithm for (μ, p, q)-Partition.*

As we have seen, an algorithm for (μ, p, q)-Cluster gives us an algorithm for (μ, p, q)-Partition. In the rest of the paper, we devise more efficient algorithms for (μ, p, q)-Cluster than the $n^{O(q)}$ time brute force method described above.

3 Parameterization by q

The main result of this section is that (μ, p, q)-Partition is (randomized) FPT parameterized by q for the three functions nonedge, nondeg, and size.

Theorem 3. *There is an algorithm for* (size, p, q)-Partition, $(\mathsf{nonedge}, p, q)$-Partition *and* (nondeg, p, q)-Partition *using $2^{O(q)}|V(G)|^{O(1)}$ randomized time. If the input instance is a yes-instance the algorithm incorrectly returns no with probability less than $\frac{1}{2}$. On no-instances the algorithm always answers no.*

By Lemma 1, all we need to show is that (μ, p, q)-cluster is fixed-parameter tractable parameterized by q. We introduce a somewhat technical variant of this question, the Satellite Problem, and show that for *every monotone function* μ, if Satellite Problem is FPT, then (μ, p, q)-cluster is FPT as well. Thus we need arguments specific to a particular μ only for the Satellite Problem.

Satellite Problem
Input: A graph G, integers p, q, a vertex $v \in V(G)$, a partition V_0, V_1, $\ldots$, V_n of $V(G)$ such that $v \in V_0$ and there is no edge between V_i and V_j for any $1 \leq i < j \leq n$.
Find: A (μ, p, q)-cluster C with $V_0 \subseteq C$ such that for every $1 \leq i \leq n$, either $C \cap V_i = \emptyset$ or $V_i \subseteq C$.

That is, for every V_i, we have to decide whether to include or exclude it from the solution C. If we exclude V_i from C, then $d(C)$ increases by the number of edges between V_0 and V_i. If we include V_i into C, then $\mu(C)$ increases accordingly. Thus we need to solve the knapsack-like problem of including sufficiently many V_i such that $d(C) \leq q$, but not including too many to ensure $\mu(C) \leq p$. As we shall see in Section 3.3, in many cases this problem can be solved by dynamic programming (and some additional arguments). The important fact that we use is that there are no edges between V_i and V_j, thus for many reasonable functions μ, the way $\mu(C)$ increases by including V_i is fairly independent from whether V_j is included in C or not.

The reduction to Satellite Problem uses the concept of important separators (Section 3.1). The reduction itself is given in Section 3.2. In Section 3.3,

we show how the SATELLITE PROBLEM can be solved for the three functions nonedge, nondeg, size.

3.1 Important separators and Important Sets

The notion of *important separators* was introduced in [12] to prove the fixed-parameter tractability of multiway cut problems. This notion turned out to be useful in other applications as well [5, 6, 17]. The basic idea is that in many problems where terminals need to be separated in some way, it is sufficient to consider separators that are "as far as possible" from one of the terminals. Let s, t be two vertices of a graph G. An $s-t$ *separator* is a set $S \subseteq E(G)$ of edges separating s and t, i.e., there is no $s-t$ path in $G \setminus S$. An $s-t$separator is *inclusionwise minimal* if there is an $s-t$ path in $G \setminus S'$ for every $S' \subset S$.

Definition 4. *Let $s, t \in V(G)$ be vertices, $S \subseteq E(G)$ be an $s-t$ separator, and let K be the component of $G \setminus S$ containing s. We say that S is an* important $s-t$ separator *if it is inclusionwise minimal and there is no $s-t$ separator S' with $|S'| \leq |S|$ such that $K \subset K'$ for the component K' of $G \setminus S'$ containing s.*

We now define *important sets*, which are natural companions to important separators.

Definition 5. *We say that a set $X \subseteq V(G)$, $v \notin X$ is* important *if (1) $d(X) \leq q$, (2) $G[X]$ is connected and (3) there is no $Y \supset X$, $v \notin Y$ such that $d(Y) \leq d(X)$ and $G[Y]$ is connected.*

It is easy to see that X is an important set if and only if $\Delta(X)$ is an important $u-v$ separator for every $u \in X$. As there are differences between edge and vertex separators, and some of the results appear only implicitly in previous papers, the full version of this article [11] contains proofs of Theorem 6 and Lemma 7. Since X is an important set if and only if $\Delta(X)$ is an important $u-v$ separator, we can use Theorem 6 and Lemma 7 to enumerate important sets.

Theorem 6 ($\star$). [1] *Let $s, t \in V(G)$ be two vertices in graph G. For every $k \geq 0$, there are at most 4^k important $s-t$ separators of size at most k. Furthermore, these important separators can be enumerated in time $4^k \cdot n^{O(1)}$.*

Lemma 7 ($\star$). *Let $s, t \in V(G)$. If $\mathcal{S}$ is the set of all important $s-t$ separators, then $\sum_{S \in \mathcal{S}} 4^{-|S|} \leq 1$. Thus $\mathcal{S}$ contains at most 4^k separators of size at most k.*

3.2 Reduction to the Satellite Problem

In this section we reduce (μ, p, q)-CLUSTER to the SATELLITE PROBLEM.

Lemma 8. *If* SATELLITE PROBLEM *can be solved in time $f(q) \cdot n^{O(1)}$ for some monotone μ, then there is a randomized $2^{O(q)} \cdot f(q) \cdot n^{O(1)}$ algorithm with constant error probability that finds a (μ, p, q)-cluster containing v (if one exists).*

[1] Proofs of results labelled with $\star$ have been omitted due to space restrictions.

The following lemma establishes the connection between important sets and finding (μ, p, q)-clusters: we can assume that the components of $G \setminus C$ for the solution C are important sets. In Lemma 10, we show that by randomly choosing important sets, with some probability we can obtain an instance of the Satellite Problem where $V_1, \ldots, V_n$ contain all the components of $G \setminus C$. This gives us the reduction stated in Lemma 8 above.

Lemma 9. *Let C be an inclusionwise minimal (μ, p, q)-cluster containing v. Then every component of $G \setminus C$ is an important set.*

Proof. Let X be a component of $G \setminus C$. It is clear that X satisfies the first two properties of Definition 5 (note that $\Delta(X) \subseteq \Delta(C)$). Thus let us suppose that there is a $Y \supset X$, $v \notin Y$ such that $d(Y) \leq d(X)$ and $G[Y]$ is connected. Let $C' := C \setminus Y$. Note that C' is a proper subset of C: every neighbor of X is in C, thus a connected superset of X has to contain at least one vertex of C. It is easy to see that C' is a (μ, p, q)-cluster: we have $\Delta(C') \subseteq (\Delta(C) \setminus \Delta(X)) \cup \Delta(Y)$ and therefore $d(C') \leq d(C) - d(X) + d(Y) \leq d(C) \leq q$ and $\mu(C') \leq \mu(C) \leq p$ (by the monotonicity of μ). This contradicts the minimality of C. □

Lemma 10. *Given a graph G, vertex $v \in V(G)$, integers p, q, and a monotone function $\mu : 2^{V(G)} \to \mathbb{Z}^+$, we can construct in time $2^{O(q)} \cdot n^{O(1)}$ an instance I of the* Satellite Problem *such that*

- *If some (μ, p, q)-cluster contains v, then I is a yes-instance with probability $2^{-O(q)}$,*
- *If there is no (μ, p, q)-cluster containing v, then I is a no-instance.*

Proof. For every $u \in V(G)$, $u \neq v$, let us use the algorithm of Lemma 7 to enumerate every important $u - v$ separator of size at most q. For every such separator S, let us put the component K of $G \setminus S$ containing u into the collection $\mathcal{S}$. Note that a component K can be obtained for more than one vertex u, but we put only one copy into $\mathcal{S}$.

Let $\mathcal{S}'$ be a subset of $\mathcal{S}$, where each member K of $\mathcal{S}$ is chosen with probability $2^{-d(K)}$ independently at random. Let Z be the union of the sets in $\mathcal{S}'$, let $V_1, \ldots, V_n$ be the connected components of $G[Z]$, and let $V_0 = V(G) \setminus Z$. It is clear that $V_0, V_1, \ldots, V_n$ give an instance I of Satellite Problem, and a solution for I gives a (μ, p, q)-cluster containing v. Thus we only need to show that if there is a (μ, p, q)-cluster C containing v, then I is a yes-instance with probability $2^{-O(q)}$.

Let C be an inclusionwise minimal (μ, p, q)-cluster containing v. Let B be the vertices on the boundary of C, i.e., the vertices of C incident to $\Delta(C)$. Let $K_1, \ldots, K_t$ be the components of $G \setminus C$. Note that every edge of $\Delta(C)$ enters some K_i, thus $\sum_{i=1}^{t} d(K_i) = d(C) \leq q$. By Lemma 9, every K_i is an important set, and hence it is in $\mathcal{S}$. Consider the following two events:

(1) Every component K_i of $G \setminus C$ is in $\mathcal{S}'$ (and hence $K_i \subseteq Z$).
(2) $Z \cap B = \emptyset$.

The probability that (1) holds is $\prod_{i=1}^{t} 4^{-d(K_i)} = 4^{-\sum_{i=1}^{t} d(K_i)} \geq 4^{-q}$. Event (2) holds if for every $b \in B$, no set $K \in \mathcal{S}$ with $b \in K$ is selected into $\mathcal{S}'$. It follows directly from the definition of important separators that for every $K \in \mathcal{S}$ with $b \in K$, $\Delta(K)$ is an important $b - v$ separator. Thus by Lemma 7, $\sum_{K \in \mathcal{S}, b \in K} 4^{-|d(K)|} \leq 1$. The probability that $Z \cap B = \emptyset$ can be bounded by

$$\prod_{K \in \mathcal{S}, K \cap B \neq \emptyset} (1-4^{-d(K)}) \geq \prod_{b \in B} \prod_{K \in \mathcal{S}, b \in K} (1-4^{-d(K)}) \geq \prod_{b \in B} \prod_{K \in \mathcal{S}, b \in K} \exp(\frac{-4^{-d(K)}}{(1-4^{-d(K)})})$$

$$\geq \prod_{b \in B} \prod_{K \in \mathcal{S}, b \in K} \exp(-\frac{4}{3} \cdot 4^{-d(K)}) = \prod_{b \in B} \exp\left(-\frac{4}{3} \cdot \sum_{K \in \mathcal{S}, b \in K} 4^{-d(K)}\right) \geq (e^{-\frac{4}{3}})^{|B|} \geq e^{-4q/3}.$$

In the first inequality, we use that every term is less than 1 and every term on the right hand side appears at least once on the left hand side; in the second inequality, we use that $1 + x \geq \exp(x/(1+x))$ for every $x > -1$. Events (1) and (2) are independent: (1) is a statement about the selection of subsets of $\mathcal{S}$ that are disjoint from B, while (2) involves only sets intersecting B. Thus by probability $2^{-O(q)}$, both (1) and (2) hold.

Suppose that both (1) and (2) hold, we show that instance I of the Satellite Problem is a yes-instance. In this case, every component K_i of $G \setminus C$ is a component V_j of $G[Z]$: $K_i \subseteq Z$ by (1) and every neighbor of K_i is outside Z. Thus C is a solution of I, as it can be obtained as the union of V_0 and some components of $G[Z]$. □

3.3 Solving the Satellite Problem

In this section, we give efficient algorithms for solving the Satellite Problem when the function μ is size, nonedge and nondeg. We describe the three algorithms by increasing difficulty. In the case when μ is size, solving the Satellite Problem turns out to be equivalent to the classical Knapsack problem with polynomial bounds on the values and weights of the items.

Recall that the input to the Satellite Problem is a graph G, integers p, q, a vertex $v \in V(G)$, a partition V_0, V_1, ..., V_n of $V(G)$ such that $v \in V_0$ and there is no edge between V_i and V_j for any $1 \leq i < j \leq n$. The task is to find a vertex set C, such that $C = V_0 \cup \bigcup_{i \in S} V_i$ for a subset S of $\{1, \ldots, n\}$ and C satisfies $d(C) \leq q$ and $\mu(C) \leq p$. For a subset S of $\{1, \ldots, n\}$ we define $C(S) = V_0 \cup \bigcup_{i \in S} V_i$.

Lemma 11. *The* Satellite Problem *for measure* size *can be solved in time* $O(q|V(G)| \log |V(G)|)$.

Proof. Notice that $d(C) = d(V_0) - \sum_{i \in S} d(V_i)$. Hence, we can reformulate the Satellite Problem with $\mu =$ size as finding a subset S of $\{1, \ldots, n\}$ such that $\sum_{i \in S} d(V_i) \geq d(V_0) - q$ and $\sum_{i \in S} |V_i| \leq p - |V_0|$. Thus, we can associate with every i an item with value $d(V_i)$ and weight $|V_i|$. The objective is to find a set of items with total value at least $d(V_0) - q$ and total weight at most $p - |V_0|$. This

problem is known as KNAPSACK and can be solved in $O(nv \log w)$ time by a classical dynamic programming [4, 7] algorithm, where n is the number of items, v is the value we seek to attain and w is the weight limit. Since the value is bounded from above by q and the weight by $|V(G)|$, the statement of the lemma follows. □

The case that $\mu = \mathsf{nonedge}$ is slightly more complicated, however we can still solve it using a dynamic programming algorithm. For the version of SATELLITE PROBLEM when $\mu = \mathsf{nondeg}$ we do not have a polynomial time algorithm. Instead, we give a $2^q|V(G)|^{O(1)}$ time randomized algorithm.

Lemma 12 ($\star$). *The* SATELLITE PROBLEM *for* nonedge *can be solved in time* $O(pn|E(G)||V(G)|)$. *There is a randomized algorithm which given an instance of* nondeg-SATELLITE PROBLEM *runs in* $2^q|V(G)|^{O(1)}$ *time, correctly answers* no *on all* no*-instances and answers* yes *on* yes*-instances with probability at least* e^{-2q}.

Repeating the algorithm for nondeg-SATELLITE PROBLEM $O(e^{2q})$ times will decrease the probability of false negatives from $1 - e^{-2q}$ to $\frac{1}{2}$. Lemmata 10, 11, and 12 give Theorem 3.

4 Parameterization by p

Theorem 13. *There is a* $8e^{p+o(p)}|V(G)|^{O(1)}$ *time algorithm for the problem* (size, p, q)-PARTITION *and a* $8e^{3p+o(p)}|V(G)|^{O(1)}$ *time algorithm for the problems* $(\mathsf{nonedge}, p, q)$-PARTITION *and* (nondeg, p, q)-PARTITION.

Because of Lemma 1, it is sufficient to solve the corresponding (μ, p, q)-CLUSTER problem within the same time bound. The setting is as follows. We are given a graph G, integers p and q and a vertex v in G. The objective is to find a set C *not* containing v such that $d(C \cup \{v\}) \leq q$ and, depending on which problem we are solving, either $|C \cup \{v\}| = \mathsf{size}(C \cup \{v\}) \leq p$, $\mathsf{nonedge}(C \cup \{v\}) \leq p$ or $\mathsf{nondeg}(C \cup \{v\}) \leq p$.

For a set S and vertex v, define $\Delta(S, v)$ to be the set of edges with one endpoint in S and one in $\{v\}$. Define $\overline{\Delta}(S, v)$ to be $\Delta(S) \setminus \Delta(S, v)$, and let $d(S, v) = |\Delta(S, v)|$ and $\overline{d}(S, v) = |\overline{\Delta}(S, v)|$. We will say that a set C is v-*minimal* if $v \notin C$ and $d(C' \cup \{v\}) > d(C \cup \{v\})$ for every $C' \subset C$. As size, nonedge and nondeg are monotone we can focus on v-minimal sets C. The following fact uses that there are no parallel edges:

Observation 14. Let C be a v-minimal set. Then $\overline{d}(C, v) < d(C, v) \leq |C|$

In particular, if $\overline{d}(C, v) \geq d(C, v)$, then $d(v) \leq d(C \cup \{v\})$, contradicting that C is minimal. Since $\overline{d}(C, v) < |C|$, it follows that C must contain a vertex u such that $N[u] \subseteq C \cup \{v\}$. Now we show that there are not too many v-minimal sets C of size at most p such that $G[C]$ is connected.

Lemma 15. *For any graph* G, *vertex* v *and integer* p, *there are at most* $4^p|V(G)|$ v*-minimal sets* C *such that* $|C| \leq p$ *and* $G[C]$ *is connected. Furthermore, all such sets can be listed in* $O(4^p|V(G)|)$ *time.*

Proof. By Observation 14, any v-minimal set C of size at most p satisfies $\overline{d}(C, v) < p$. Let S be a set such that $|S| \leq p$ and $G[S]$ is connected. Let F be a subset of $N(S) \setminus \{v\}$ of size at most $p-1$. We prove by downward induction on $|S|$ and $|F|$ that there are at most $2^{2p-|S|-|F|-1}$ v-minimal sets such that $|C| \leq p$, $G[C]$ is connected, $S \subseteq C$, and $F \cap C = \emptyset$. If $|S| = p$ then the only possibility for C is S, while $2^{2p-|S|-|F|-1} \geq 1$. Similarly, consider the case that $|F| = p-1$. Now, every vertex of F has at least one edge into C and hence $\overline{d}(C, v) = p-1$. Hence $N(C) = F \cup \{v\}$ and the only possibility for C is the connected component of $G \setminus (F \cup \{v\})$ that contains S. Hence there is one possibility for C and $2^{2p-|S|-|F|-1} \geq 1$.

For the inductive step, consider a set S such that $|S| \leq p$ and $G[S]$ is connected and a subset F of $N(S) \setminus \{v\}$ of size at most $p-1$. We want to bound the number of v-minimal sets such that $|C| \leq p$ and $G[C]$ is connected, $S \subseteq C$ and $F \cap C = \emptyset$. If $N(S) \setminus (F \cup \{v\})$ is empty, then there is only one choice for C, namely S, and $2^{2p-|S|-|F|-1} \geq 1$. Otherwise, consider a vertex $u \in N(S) \setminus (F \cup \{v\})$. By the induction hypothesis, the number of v-minimal sets such that $|C| \leq p$ and $G[C]$ is connected, $S \cup \{u\} \subseteq C$ and $F \cap C = \emptyset$ is at most $2^{2p-|S|-|F|-2}$. Similarly, the number of v-minimal sets such that $|C| \leq p$ and $G[C]$ is connected, $S \subseteq C$ and $(F \cup \{u\}) \cap C = \emptyset$ is at most $2^{2p-|S|-|F|-2}$. Since either $u \in C$ or $u \notin C$, the two cases cover all possibilities for C and hence there are at most $2 \cdot 2^{2p-|S|-|F|-2} = 2^{2p-|S|-|F|-1}$ possibilities for C.

For a fixed S and F, the above proof can be translated into a procedure which lists all v-minimal sets such that $|C| \leq p$ and $G[C]$ is connected, $S \subseteq C$ and $F \cap C = \emptyset$. We run the procedure for $S = \{u\}$ and $F = \emptyset$ for every possible choice of u. Hence, there are at most $4^p|V(G)|$ v-minimal sets C such that $|C| \leq p$ and $G[C]$ is connected, and the sets can be efficiently listed. This concludes the proof. □

Observation 16. Let C be a v-minimal set of G and $G[S]$ be a connected component of $G[C]$. Then S is a v-minimal set.

In particular, if S is not a v-minimal set, then it contains a v-minimal set $S' \subset S$ and it is easy to see that $d(\{v\} \cup (C \setminus S) \cup S') \leq d(\{v\} \cup C)$, contradicting the minimality of C. Observation 16 tells us that any v-minimal set is the union of connected v-minimal sets. This makes it possible to use Lemma 15. We are now ready to give an algorithm for (size, p, q)-Cluster, the easiest of the three clustering problems. Our algorithm is based on a combination of *color coding* [2] with a dynamic programming algorithm which uses the observations made in this section.

Proposition 17 ([16]). *For every n, k there is a family of functions $\mathcal{F}$ of size $O(e^k \cdot k^{O(\log k)} \cdot \log n)$ such that every function $f \in \mathcal{F}$ is a function from $\{1, \ldots, n\}$ to $\{1, \ldots, k\}$ and for every subset S of $\{1, \ldots, n\}$ there is a function $f \in \mathcal{F}$ that is bijective when restricted to S. Furthermore, given n and k, $\mathcal{F}$ can be computed in time $O(e^k \cdot k^{O(\log k)} \cdot \log n)$.*

Lemma 18. (size, p, q)-Cluster *can be solved in time $2^{O(p)}|V(G)|^{O(1)}$.*

Proof. We are given as input a graph G together with a vertex v and integers p and q. The task is to find a vertex set C of size at most $p-1$ such that $d(\{v\} \cup C) \leq q$. It is sufficient to search for a v-minimal set C satisfying these properties. By Observation 16, C can be decomposed into $C = S_1 \cup S_2 \ldots \cup S_t$ such that S_i is a connected v-minimal set for every i, $S_i \cap S_j = \emptyset$ for every $i \neq j$ and no edge of G has one endpoint in S_i and the other in S_j for every $i \neq j$. The algorithm of Lemma 15 can be used to list all connected v-minimal sets $S_1 \ldots S_n$; we have $n \leq 4^p|V(G)|$. For a subset Z of $\{1,\ldots,n\}$, define $C(Z) = \{v\} \cup \bigcup_{i \in Z} S_i$. Let $Z \subseteq \{1,\ldots,n\}$ be such that for every $i,j \in Z$ with $i \neq j$, we have $S_i \cap S_j = \emptyset$. We have that $|C(Z)| = 1 + \sum_{i \in Z} |S_i|$ and

$$d\big(C(Z)\big) \leq d(v) + \sum_{i \in Z} (\overline{d}(S_i, v) - d(S_i, v)).$$

If there is no edge with one endpoint in S_i and the other in S_j for some $i \neq j$, $i,j \in Z$, then the inequality above holds with equality. Our algorithm will select a Z such that $C = \bigcup_{i \in Z} S_i$. To ensure that the algorithm picks Z such that the sets S_i and S_j will be disjoint for every pair of distinct integers $i,j \in Z$ we will use color coding. In particular, we construct a family $\mathcal{F}$ of functions from $V(G) \setminus \{v\}$ to $\{1,\ldots,p-1\}$ as described in Proposition 17. The family $\mathcal{F}$ has size $O(e^p \cdot p^{O(\log p)} \cdot \log |V(G)|)$.

For each function $f \in \mathcal{F}$ we will think of the function as a coloring of $V(G) \setminus \{v\}$ with colors from $\{1,\ldots,p-1\}$. We will only look for a v-minimal set C whose vertices have different colors. This will not only ensure that any two sets S_i and S_j that we pick will be disjoint, it also automatically ensures that the size of the set C we return is at most $p-1$. If the input instance was a yes-instance then a solution set C exists, and the construction of $\mathcal{F}$ ensures that there will be a function $f \in \mathcal{F}$ which colors all vertices in C with different colors.

When considering a particular coloring f, we discard all sets from $S_1, \ldots S_n$ which have two vertices of the same color, so from this point, without loss of generality, all sets in $S_1, \ldots S_n$ have at most one vertex of each color. For a vertex set S, define $\mathsf{colors}(S)$ to be the set of colors occuring on vertices on G. For every $0 \leq i \leq n$, $0 \leq j \leq |E(G)|$ and $R \subseteq \{1,\ldots,p-1\}$, we define $T[i,j,S]$ to be true if there is a set $Z \subseteq \{1,\ldots,i\}$ such that all vertices of $C(Z)$ have distinct colors, $d(v) + \sum_{i \in Z}(\overline{d}(S_i,v) - d(S_i,v)) = j$ and $\mathsf{colors}(C(Z)) \subseteq R$. Clearly, there is a v-minimal set C such that $d(\{v\} \cup C) \leq q$ and all vertices of C have different color if and only if $T[n,j,\{1,\ldots,p-1\}]$ is true for some $j \leq q$. We can fill the table T using the following recurrence.

$$T[i,j,R] = \begin{cases} T[i-1,j,R] & \text{if } \mathsf{colors}(S_i) \setminus R \neq \emptyset \\ T[i-1,j,R] \vee T[i-1, & \\ j + d(S_i,v) - \overline{d}(S_i,v), R \setminus \mathsf{colors}(S_i)] & \text{otherwise} \end{cases} \quad (1)$$

Here we initialize $T[0,d(v),\emptyset]$ to true. The table has size $4^p|V(G)|^{O(1)} \cdot 2^p|V(G)|^{O(1)} = 8^p|V(G)|^{O(1)}$ and can be filled in time proportional to its size. Hence the total running time for the algorithm is $(8e)^{p+o(p)}|V(G)|^{O(1)}$. □

For (size, p, q)-Cluster the size of the set C we look for is already bounded by p. For $(\mathsf{nonedge}, p, q)$-Cluster and (nondeg, p, q)-Cluster, we cannot make this assumption, thus further arguments are needed to obtain Theorem 13.

4.1 Hardness results

The algorithmic results in Section 3 still hold when parallel edges are allowed. Interestingly, the positive results in Section 4 do not: in particular, Observation 14 breaks done if there are parallel edges. The following hardness result shows that allowing parallel edges indeed make the problems more difficult:

Theorem 19 ($\star$). $(\mathsf{nonedge}, p, q)$-Partition *and* (nondeg, p, q)-Partition *are NP-complete for* $p = 0$ *on graphs with parallel edges.* (size, p, q)-Partition *is W[1]-hard parameterized by* p *on graphs with parallel edges.*

References

1. N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. In *STOC 2005*, pages 684–693, 2005.
2. N. Alon, R. Yuster, and U. Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.
3. N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.
4. R. Bellman. Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63, 1962.
5. J. Chen, Y. Liu, and S. Lu. An improved parameterized algorithm for the minimum node multiway cut problem. In *WADS*, pages 495–506, 2007.
6. J. Chen, Y. Liu, S. Lu, B. O'Sullivan, and I. Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *J. ACM*, 55(5), 2008.
7. T. Cormen, C. Leiserson, R. Rivest, and C. Stein. Introduction to algorithms, 2001.
8. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
9. P. Heggernes, D. Lokshtanov, J. Nederlof, C. Paul, and J. A. Telle. Generalized graph clustering: Recognizing (o,q)-cluster graphs. In *WG*, pages 171–183, 2010.
10. M. A. Langston and B. C. Plaut. On algorithmic applications of the immersion order : An overview of ongoing work presented at the third slovenian international conference on graph theory. *Discrete Mathematics*, 182(1-3):191–196, 1998.
11. D. Lokshtanov and D. Marx. Clustering with local restrictions. In preparation. Availiable at http://www.ii.uib.no/ daniello/papers/clusteringLocal.pdf.
12. D. Marx. Parameterized graph separation problems. *Theoret. Comput. Sci.*, 351(3):394–406, 2006.
13. D. Marx and I. Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. To appear in STOC 2011.
14. C. Mathieu, O. Sankur, and W. Schudy. Online correlation clustering. In *STACS*, pages 573–584, 2010.
15. C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *SODA*, pages 712–728, 2010.
16. M. Naor, L. J. Schulman, and A. Srinivasan. Splitters and near-optimal derandomization. In *FOCS*, pages 182–191, 1995.
17. I. Razgon and B. O'Sullivan. Almost 2-sat is fixed-parameter tractable (extended abstract). In *ICALP 2008(1))*, pages 551–562, 2008.