Learning with Structure: Computing Consistent Subsets on Structurally-Regular Graphs

Aritra Banik[1], **Mano Prakash Parthasarathi**[2], **Venkatesh Raman**[3], **Diya Roy**[1], **Abhishek Sahu**[1]

[1]National Institute of Science, Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, India
[2]North Carolina State University, Raleigh, NC, USA
[3](Retd) The Institute of Mathematical Sciences, HBNI, Chennai, India
aritra@niser.ac.in, mpartha@ncsu.edu, vraman@imsc.res.in, diya.roy@niser.ac.in, abhisheksahu@niser.ac.in

Abstract

The Minimum Consistent Subset (MCS) problem arises naturally in the context of supervised clustering and instance selection. In supervised clustering, one aims to infer a meaningful partitioning of data using a small labeled subset. However, the sheer volume of training data in modern applications poses a significant computational challenge. The MCS problem formalizes this goal: given a labeled dataset X in a metric space, the task is to compute a smallest subset S of X such that every point in X shares its label with at least one of its nearest neighbors in S.

Recently, the MCS problem has been extended to graph metrics, where distances are defined by shortest paths. Prior work has shown that MCS remains NP-hard even on simple graph classes like trees, and presented an fixed-parameter tractable (FPT) algorithm parameterized by the number of colors for MCS on trees. This raises the challenge of identifying graph classes that admit algorithms efficient in both input size (n) and the number of colors (c).

In this work, we study the Minimum Consistent Subset problem on graphs, focusing on two well-established measures: the vertex cover number (vc) and the neighborhood diversity (nd). Specifically, we design efficient algorithms for graphs exhibiting small vc or small nd, which frequently arise in real-world domains characterized by local sparsity or repetitive structure. These parameters are particularly relevant because they capture structural properties that often correlate with the tractability of otherwise hard problems. Graphs with small vertex cover sizes are "almost independent sets", representing sparse interactions, while graphs with small neighborhood diversity exhibit a high degree of symmetry and regularity. Importantly, small neighborhood diversity can occur even in dense graphs, a property frequently observed in domains such as social networks with modular communities or knowledge graphs with repeated relational patterns. Thus, algorithms designed to work efficiently for graphs with small neighborhood diversity are capable of efficiently solving MCS in complex settings where small vertex covers may not exist.

We show that MCS is FPT when parameterized by the vertex cover number and by neighborhood diversity. In each case, we present an algorithm whose running time is polynomial in n and c, and the non-polynomial part depends solely on the chosen parameter. Notably, our algorithms remain efficient for arbitrarily many colors, as their complexity is polynomially dependent on the number of colors.

Extended version — https://arxiv.org/abs/2512.12860

Introduction

Clustering lies at the heart of numerous tasks in computer science and machine learning. In its essence, given a set of points in a metric space (P, d), the objective is to partition them such that "proximate" points reside within the same cluster. While various unsupervised approaches exist, supervised learning offers a powerful paradigm for achieving "most appropriate" clustering.

In supervised clustering, a labeled training dataset i.e., a subset of points $P' \subset P$ endowed with a coloring function $\mathfrak{C} : P' \to [c]$ (where each color denotes a class/cluster) is provided to distill underlying patterns. Usually, given the training dataset, a learning algorithm outputs a set of cluster centers $C = \{c_1, \ldots, c_r\}$. Subsequently, an unlabeled point q is assigned the color $\mathfrak{C}(c_i)$ where $c_i = \text{NN}(q, C)$, with $\text{NN}(q, C)$ representing the nearest neighbors of q in C.

However, the ever-increasing volume of modern datasets poses significant computational challenges for learning algorithms. Large datasets, while information-rich, often lead to protracted learning times. This has motivated a rich line of work on *instance selection*, where the goal is to extract a small, yet representative, subset of the training data that preserves classification behavior. A classical formulation of this idea is the *Minimum Consistent Subset* (MCS) problem, introduced in 1968 (Hart 1968). Given a colored training dataset T, the MCS problem seeks a minimum cardinality subset $S \subseteq T$ such that for every point $t \in T$, the color of t is same as the color of at least one of its nearest neighbors in S. Despite its apparent simplicity, the MCS problem poses significant computational hurdles and is known to be computationally hard in Euclidean spaces (Wilfong 1991; Khodamoradi, Krishnamurti, and Roy 2018), and also hard to approximate in general settings (Chitnis 2022).

The MCS problem has recently been extended to *graph metrics*, motivated by applications where similarity is naturally modeled by graphs, such as social or knowledge networks. In the *Consistent Subset Problem on Graphs* (CSPG), we are given a graph $G = (V, E)$ with a vertex coloring $\mathfrak{C} : V \to [c]$. The distance metric is defined as the shortest path distance in G, denoted by $d(u, v)$. For a vertex $v \in V$ and a subset $U \subseteq V$, let $d(v, U) = \min_{u \in U} d(v, u)$. The set

of nearest neighbors of v in U is denoted by $\text{NN}(v, U) = \{u \in U : d(v, u) = d(v, U)\}$.

A subset of vertices $S \subseteq V(G)$ is called a *consistent subset* for $(G, \mathfrak{C})$ if, for every vertex $v \in V(G)$, the color of v is present among the colors of its nearest neighbors in S, i.e., $\mathfrak{C}(v) \in \mathfrak{C}(\text{NN}(v, S))$.

CONSISTENT SUBSET PROBLEM ON GRAPHS (CSPG)

Input: A graph G and a coloring function $\mathfrak{C} : V(G) \to [c]$.
Question: Compute a minimum consistent subset for $(G, \mathfrak{C})$.

This graph-theoretic version of the MCS problem (i.e., CSPG) has recently drawn attention for both its theoretical appeal and practical relevance. Polynomial-time algorithms have been discovered for certain special graph classes, such as paths, spiders, and caterpillars (Dey, Maheshwari, and Nandy 2023), and later for bi-colored trees (Dey, Maheshwari, and Nandy 2021) and for k-colored trees (for fixed k) (Arimura et al. 2023). For more related works, we refer to (Manna 2024a,b; Biniaz and Khamsepour 2024). Although results were known for bi-colored and k-colored trees, the status for the problem when the underlying graph is a general tree was open for a long time. In a recent breakthrough, (Banik et al. 2024) the authors systematically investigated CSPG and resolved this question. Their work led to two major contributions:

- They established that CSPG is NP-complete on trees, resolving a key open question. This result is particularly striking, as many hard graph problems become tractable when restricted to trees.
- They designed a fixed-parameter tractable (FPT) algorithm, i.e., an algorithm running in time $f(k) \cdot n^{\mathcal{O}(1)}$, where k is a chosen parameter and f is a computable function independent of the input size, for trees, with a running time of $\mathcal{O}(2^{6c} \cdot n^6)$, where c is the number of colors (chosen as the parameter) and n is the number of vertices. This significantly improves upon earlier brute-force approaches with super-exponential dependence on c.

The hardness of CSPG on trees, where the minimum feedback vertex set (FVS) is empty set, has significant implications: it precludes the existence of an FPT algorithm parameterized solely by FVS. This calls for stronger structural parameters to recover tractability. In this work, we take this challenge head-on and present FPT algorithms for MCS parameterized by two natural and well-studied graph parameters:

- **Vertex Cover Number** (vc), which measures the minimum number of vertices needed to cover all edges of the graph.
- **Neighborhood Diversity** (nd), which bounds the number of types of neighborhoods across the graph and is strictly stronger than vertex cover in dense graphs.

A formal definition for both parameters is presented in the next section. Our key contribution is that our algorithms are independent of the number of colors c, in stark contrast to prior work where the exponential dependence on c was unavoidable. Specifically, we show:

- MCS admits an FPT algorithm parameterized by vertex cover size, with running time $k^{\mathcal{O}(k)} \cdot \text{poly}(n, c)$, where f is color-independent and k is the size of the vertex cover.
- MCS also admits an FPT algorithm parameterized by neighborhood diversity, again avoiding any exponential dependence on the number of colors.

In particular, we want to bring to the reader's attention that while designing an FPT algorithm with dependence on both neighborhood diversity (r) and the number of colors (c) is straightforward, due to Claim 1.10, removing the dependence on the number of colors is highly non-trivial. This is because when the number of colors is large, the number of possible combinations becomes prohibitively high, resulting in a running time that is no longer FPT in r. However, our key insight is that the interaction of a small number of important or *responsible* colors with the solution is sufficient to determine the interaction of all other colors. While we may not be able to explicitly identify these responsible colors in advance, once we know how they interact with the solution, we can use a color-coding technique to probabilistically isolate and identify a most suitable set of such colors. This allows us to reduce the problem to a collection of independent subproblems, each of which can be solved separately using a greedy algorithm. The solutions to these subproblems can then be combined to obtain a solution to the original instance. To achieve this, we exploit structural properties arising from both the neighborhood diversity of the graph and the specific characteristics of the problem.

At a high level, our algorithmic technique departs from the conventional use of color coding. Typically, color coding is employed to mark objects or structural features of a problem instance in a way that enables their independent resolution. In contrast, our approach involves color coding the elements themselves (in our case, the colors), with the goal of ensuring that a *greedily selected* subset of solution elements remains well separated under the resulting color distribution. We believe that this perspective introduces a novel and potentially widely applicable direction for color coding, with possible extensions to a broader class of combinatorial problems beyond the specific context addressed in this work.

The parameter neighborhood diversity (nd) is particularly relevant in the context of AI and machine learning applications on graphs. While vertex cover captures a notion of "sparseness" around edges, neighborhood diversity provides a finer-grained measure of structural regularity. Graphs with small neighborhood diversity are those where most vertices have neighborhoods that are structurally similar, even if the graph is dense. Such structures appear in various real-world networks, including social networks with distinct community structures, or knowledge graphs where entities often share common relational patterns. An FPT algorithm parameterized by nd is significant because it indicates tractability not just for sparse graphs (like those with small vertex

cover), but also for certain types of dense graphs that exhibit high regularity in their local connectivity patterns, a characteristic often observed in complex systems modeled as graphs in AI. This allows for efficient solutions in scenarios where a small vertex cover might not exist, but the underlying structure still permits algorithmic leverage.

Notations and Definitions

Graph Notations and Definitions: Let G be a graph. We use $V(G)$ and $E(G)$ to denote the set of vertices and edges of G, respectively. For a set of vertices S, by $G \setminus S$ we mean $G[V(G) \setminus S]$, i.e., the subgraph of G induced on $V \setminus S$. For a vertex v, $N(v)$ denotes the set of neighbors of v in G and $N[v] = N(v) \cup \{v\}$ denotes the closed neighborhood of v. We call a graph G a *complete* graph if every pair of vertices in G is adjacent. A *clique* in G is an induced subgraph that is complete. In contrast, a set $I \subseteq V(G)$ is an *independent set* if no two vertices in I are adjacent in G. A set $M \subseteq V(G)$ is a *vertex cover* if for every edge in G, at least one of its endpoints lies in M.

Two vertices u and v are of the *same type* if $N(v) \setminus \{u\} = N(u) \setminus \{v\}$. Note that, this defines an equivalence relation on $V(G)$ (Matsumoto, Kurita, and Kiyomi 2025). A *neighborhood decomposition* of a graph G is a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_w\}$ of $V(G)$ such that all vertices in each C_i are of same type. Each C_i is a *neighborhood class*, and w is the size of the decomposition. The *neighborhood diversity*, $\mathrm{ND}(G)$, is the minimum size of a neighborhood decomposition of G.

Observation 0.1. *Given a graph G, $\mathrm{ND}(G)$ can be computed in polynomial time (Lampis 2012).*

We define the set of vertices at distance ℓ from a vertex v by $N^{\ell}(v) = \{u \in V : d(u, v) = \ell\}$ and the set of vertices at distance ℓ from a vertex v of color a by $N_a^{\ell}(v) = \{u \in V : d(u, v) = \ell \text{ and } \mathfrak{C}(u) = a\}$. For any vertex v, let $N_a(v)$ denotes the set of neighbors of v, with color a. For $X \subseteq V(G)$, we define $N(X)$ to be the neighbors of vertices in X. Most of the symbols and notations of graph theory used are standard and taken from (Diestel 2012).

Parameterized Complexity: Parameterized complexity offers a framework for solving NP-hard problems more efficiently by isolating the combinatorial explosion to a parameter that is small in practice. A problem is fixed-parameter tractable (FPT) if it can be solved in time $f(\ell) \cdot |I|^{\mathcal{O}(1)}$, where ℓ is the parameter, $|I|$ is the input size, and f is a computable function. Safe reduction rules are polynomial-time preprocessing steps that simplify the instance without changing its answer. For a detailed background, readers can refer to (Cygan et al. 2015).

Hitting Set: Given a set system $(\mathcal{U}, \mathcal{F})$, we say that $\mathcal{H} \subseteq \mathcal{U}$ is a *hitting set* for $(\mathcal{U}, \mathcal{F})$ if $\forall F \in \mathcal{F}$, $\mathcal{H} \cap F \neq \emptyset$ and a set of subsets $\mathcal{F}' \subseteq \mathcal{F}$ is called a *set cover* for $(\mathcal{U}, \mathcal{F})$ if $\bigcup_{F \in \mathcal{F}'} F = \mathcal{U}$. From (Cygan et al. 2015)[Theorem 6.1], we have the following proposition.

Proposition 1. *Given a hitting set instance $\mathcal{HS}(\mathcal{U}, \mathcal{F})$, a hitting set of minimum size can be found in time $2^{|\mathcal{F}|}(|\mathcal{U}| + |\mathcal{F}|)^{\mathcal{O}(1)}$.*

The $\mathcal{O}^*$ notation suppresses polynomial factors in the input size. Specifically, $\mathcal{O}^*(f(n)) = \mathcal{O}(f(n) \cdot \mathrm{poly}(n))$, where polynomial factors are omitted for clarity when they are not the focus of the analysis.

FPT Algorithm Parameterized by Vertex Cover Size

In this section, we present a fixed-parameter tractable (FPT) algorithm for the MCS problem parameterized by the size of the vertex cover. For completeness, we begin with a formal definition of the problem.

CONSISTENT SUBSET PROBLEM PARAMETERIZED BY VERTEX COVER SIZE

Input: A graph $G = (V = M \sqcup I, E)$ where $|M| = k$ and $G[I]$ is an independent set, along with a coloring function $\mathfrak{C} : V(G) \to [c]$.
Question: Compute a minimum consistent subset (MCS) S for $(G, \mathfrak{C})$.
Parameter: k

It is well-known that the VERTEX COVER problem is FPT when parameterized by the solution size k (Cygan et al. 2015). Let k be the size of the minimum vertex cover. As a preprocessing step, we compute a vertex cover M of size k. We define $I = V(G) \setminus M$. Observe that the induced subgraph $G[I]$ is edgeless.

Observation 1.1. *For any two vertices u and v in G, $0 \leq d(u, v) \leq 2k$. In particular, if at least one of $u, v \in M$, then $0 \leq d(u, v) \leq (2k - 1)$.*

Proof. Let P be a shortest path between vertices u and v. Since I is an independent set, no two consecutive vertices on P can belong to I. Thus, between any two vertices from I, there must be at least one vertex from M.

The path P can contain at most k vertices from M, as $|M| = k$. Therefore, the number of vertices from I on P is at most $k+1$. This gives an upper bound on the total number of vertices in P as $k + (k + 1) = 2k + 1$.

In the case where either $u \in M$ or $v \in M$, the number of vertices from I on P can be at most k, and thus the total number of vertices in P is at most $2k$. Therefore, the observation follows. □

Next, we make two guesses with respect to a minimum consistent subset S and attempt to find a solution that respects the guesses.

Guess 1: We guess the distances from each vertex u_i in M to S. More specifically, we assume that an array $\mathcal{D} = [d_1, d_2, \cdots d_k]$ is given where d_i denote the distance between u_i and S. By Observation 1.1, each entry d_i can take a value between 0 and $(2k - 1)$. Thus, the total number of guesses for $\mathcal{D}$ is bounded by $(2k)^k$.

Guess 2: We guess the set of vertices $M_1 \subseteq M$, which consists of the neighbors of the vertices $S \cap I$. Formally, $M_1 = \{u \mid u \in N(S \cap I) \setminus (S \cap M)\}$. The number of choices is bounded by 2^k.

Let $I^{\text{OUT}}(\mathcal{D})$ be the set of vertices in I that are at a distance at most $d_i - 1$ from some vertex $u_i \in M$. For any choice of $(\mathcal{D}, M_1)$, we say that a set of vertices $X \subseteq V(G)$ respects the choice $(\mathcal{D}, M_1)$, if $\forall u_i \in M$, $d(u_i, X) = d_i$ and $N(X \cap I) = M_1$. Therefore, given $(\mathcal{D}, M_1)$, our aim is to find a minimum cardinality consistent subset $S \subseteq V(G)$ that respects the choice $(\mathcal{D}, M_1)$.

Observation 1.2. *For any minimal consistent subset S respecting $(\mathcal{D}, M_1)$, $S \cap I^{\text{OUT}}(\mathcal{D}) = \emptyset$.*

Proof. Assume, for the sake of contradiction, that there exists a vertex $v \in S \cap I^{\text{OUT}}(\mathcal{D})$. By the definition of $I^{\text{OUT}}(\mathcal{D})$, there exists a vertex $u_i \in M$ such that

$$d(u_i, v) \leq d_i - 1,$$

where $d_i = d(u_i, S)$ by definition. Since $v \in S$, it follows that

$$d(u_i, S) \leq d(u_i, v) = d_i - 1,$$

which is a contradiction to $d_i = d(u_i, S)$ as defined in $\mathcal{D}$. Therefore, our assumption is false, and hence we conclude that $S \cap I^{\text{OUT}}(\mathcal{D}) = \emptyset$. □

We define $M_0 = S \cap M$, i.e. $M_0 = \{u_i \in M \mid d_i = 0\}$ and $M_x = M \setminus (M_0 \cup M_1)$. Recall, for any vertex v, we denote the set of vertices at distance d from v by $N^d(v)$, the set of vertices of color a in the neighbor of v by $N_a(v)$ and the set of vertices of color a at distance d from v by $N_a^d(v)$.

We extend the scope of $\mathcal{D}$ and define d_i for the vertices u_i in I as follows. Let $d_{u_i}^{min}$ be the minimum distance in $\mathcal{D}$ among the set of vertices $N(u_i)$, i.e. $d_{u_i}^{min} = \min_{u_j \in N(u_i)} d_j$. Note that all the neighbors of u_i are in M and hence $d_i = d_{u_i}^{min} + 1$ is well defined. For any vertex $u_i \in I$, we define C_i to be the set of colors of all those vertices that are at distance d_i from u_i and do not belong to the set $(M_x \cup M_1)$, i.e. $C_i = \{\mathfrak{C}(u_j) \mid u_j \in N^{d_i}(u_i) \setminus (M_x \cup M_1 \cup I^{\text{OUT}}(\mathcal{D}))\}$. Let $I^{\text{IN}} \subseteq I$ be the set of vertices u_i such that $\mathfrak{C}(u_i) \notin C_i$.

Observation 1.3. *For any consistent subset S respecting $(\mathcal{D}, M_1)$, $I^{\text{IN}}(\mathcal{D}) \subseteq S$.*

Proof. Suppose not. Let $u_i \in I^{\text{IN}}(\mathcal{D})$ but $u_i \notin S$. Also, let x be the closest vertex in S from u_i such that $\mathfrak{C}(x) = \mathfrak{C}(u_i)$. Consider P as the shortest path between u_i and x, also let $u_j \in M$ be the vertex next to u_i in path P. Observe that, $d(u_j, x) < (d_{u_i}^{min} + 1) - 1 = d_{u_i}^{min} \leq d_j$, which contradicts the assumption that S respects the choice $\mathcal{D}$. □

Observation 1.4. *Given $\mathcal{D}$, in polynomial time, we can find out the set of vertices in $I^{\text{IN}}(\mathcal{D})$ and $I^{\text{OUT}}(\mathcal{D})$.*

Proof. For a given choice of $\mathcal{D}$, both $I^{\text{OUT}}(\mathcal{D})$ and $I^{\text{IN}}(\mathcal{D})$ can be constructed in polynomial time using shortest path algorithms. □

We have established that for any consistent subset S respecting $\mathcal{D}$, $I^{\text{IN}}(\mathcal{D}) \subseteq S$ and $I^{\text{OUT}}(\mathcal{D}) \cap S = \emptyset$. If $I^{\text{IN}}(\mathcal{D}) \cap I^{\text{OUT}}(\mathcal{D}) \neq \emptyset$, then we simply discard the guess $\mathcal{D}$.

We denote a vertex u_i to be *satisfied* if $\exists$ a vertex $v \in N^{d_i}(u_i) \cap (M_0 \cup I^{\text{IN}}(\mathcal{D}))$ such that $\mathfrak{C}(u_i) = \mathfrak{C}(v)$. If a vertex is not satisfied, we call it *unsatisfied* and let $\overline{U}$ denote the set of all unsatisfied vertices. For any color a, let $\overline{U}_a \subseteq \overline{U}$ denote the subset of vertices in $\overline{U}$ that are colored a.

Let S be any solution that respects $(\mathcal{D}, M_1)$. For each color a, define $S_a \subseteq S \setminus (I^{\text{IN}}(\mathcal{D}) \cup M_0)$ to be the set of vertices in S of color a, excluding those in $I^{\text{IN}}(\mathcal{D})$ and M_0. Let $S'_a \subseteq I \setminus I^{\text{OUT}}(\mathcal{D})$ be any set of vertices of color a such that for every vertex $u_i \in \overline{U}_a$, $d(u_i, S'_a) \leq d_i$.

Lemma 1.5. *The set $S' = (S \setminus S_a) \cup S'_a$ is a consistent subset respecting $(\mathcal{D}, M_1)$, when S is consistent with $(\mathcal{D}, M_1)$.*

Proof. For the sake of contradiction, suppose that the set S' is not consistent. Then, by the definition, there exists at least one vertex $u_i \in V(G)$ such that $\mathfrak{C}(u_i) \notin \mathfrak{C}(\text{NN}(u_i, S'))$. Note that $u_i \in \overline{U}$. Now, if $u_i \notin \overline{U}_a$ i.e., $\mathfrak{C}(u_i) = b$ (say). In this case, as S is a consistent subset and by the construction of S', $\exists$ a vertex $u_j \in S'$ of color b such that $d(u_i, u_j) = d_i$. Hence, $u_i \in \overline{U}_a$ and by the definition of S'_a, there exists a vertex u_j of color $\mathfrak{C}(u_i)$ such that $d(u_i, u_j) \leq d_i$. Hence, we have $d'_i = d(u_i, S') < d_i$.

Observe that $u_i \in I$; otherwise, the fact that $d(u_i, S') < d_i$ would imply that S' does not respect *Guess 1*. Let u_b be any vertex in $\text{NN}(u_i, S')$. Let $u_c \in M$ be the neighbor of u_i in M lying on the path from u_i to u_b. Then, $d(u_c, u_b) < d_i - 1$.

We know that $d_i = d_{u_i}^{min} + 1$ and $d_c \geq (d_{u_i}^{min} + 1) - 1 = d_i - 1 > d(u_c, u_b)$. Hence, S'_a contains a vertex at distance at most $d_c - 1$. Thus S'_a contains a vertex from $I^{\text{OUT}}(\mathcal{D})$, which contradicts the definition of S'_a, completing the proof. □

Therefore, from Lemma 1.5, given $\mathcal{D}$ and M_1, for each color $a \in \mathfrak{C}(\overline{U})$, our objective reduces to independently computing a minimum-size set $S_a^* \subseteq I \setminus I^{\text{OUT}}(\mathcal{D})$ of color a, such that for every vertex $u_i \in \overline{U}_a$, it holds that $d(u_i, S_a^*) \leq d_i$.

Recall, we define the set of vertices at distance ℓ from a vertex v by $N^\ell(v) = \{u \in V : d(u, v) = \ell\}$.

For any vertex $u_i \in \overline{U}_a$, let $M_1^a(u_i) \subseteq N^{d_i - 1}(u_i) \cap M_1$ to be the set of vertices such that each vertex in $M_1^a(u_i)$ has at least one neighbor of color a in $I \setminus I^{\text{OUT}}(\mathcal{D})$. Formally,

$$M_1^a(u_i) = \{u_j \in N^{d_i - 1}(u_i) \cap M_1 : N_a(u_j) \cap (I \setminus I^{\text{OUT}}(\mathcal{D})) \neq \emptyset\}$$

The intuition behind the definition of $M_1^a(u_i)$ is as follows. In order to satisfy any unsatisfied vertex u_i of color a, any solution must include at least one vertex $u \in I \setminus I^{\text{OUT}}(\mathcal{D})$ of color a where u is a neighbor of a vertex in $M_1^a(u_i)$. We define the following set system with ground set M_1, $\mathcal{M}_a(\mathcal{D}, M_1) = \{M_1^a(u_i)\}$.

Let $X_a^* \subseteq N(S_a) \cap M_1$ be the *minimal* set of vertices such that every vertex in S_a has a neighbor in X_a^*.

Observation 1.6. *X_a^* must be a minimal hitting set for $\mathcal{M}_a(\mathcal{D}, M_1)$.*

Towards finding S_a^*, we make the following final guess:

Guess 3: For each color a, guess the minimal hitting set $X_a \subseteq N(S_a) \cap M_1$. The total number of such choices is bounded by 2^k.

Given X_a, consider the set system $(I, \mathcal{F}(X_a))$ where for each vertex $u_i \in X_a$ we include set of vertices $N(u_i) \cap (I \setminus (I^{\text{IN}}(\mathcal{D}) \cup I^{\text{OUT}}(\mathcal{D})))$ of color a as a subset in the family $\mathcal{F}(X_a)$ i.e.

$$\mathcal{F}(X_a) = \{N(u_i) \cap (I \setminus (I^{\text{IN}} \cup I^{\text{OUT}})) \cap \mathfrak{C}^{-1}(a) : u_i \in X_a\}$$

For any choice X_a, let $S(X_a)$ denotes the minimum hitting set for $(I, \mathcal{F}(X_a))$.

Observation 1.7. *$(S \setminus S_a) \cup S(X_a)$ is a solution.*

Proof. By construction for any vertex $u_i \in \overline{U}_a$, $d(u_i, S_a) \leq d_i$. Observe that from Lemma 1.5, we know that $(S \setminus S_a) \cup S(X_a)$ is a solution. Note that S_a is a hitting set for $(I, \mathcal{F}(X_a))$. This completes the proof. □

Observation 1.8. *Given $\mathcal{D}$ and M_1, we can find out $S^*_a \subseteq I \setminus I^{\text{OUT}}(\mathcal{D})$ in time $2^{\mathcal{O}(k)}$.*

Proof. Observe that there are at most 2^k possible choices for X_a. For each choice of X_a, $\mathcal{F}(X_a)$ contains at most $|X_a| \leq k$ sets. Since the HITTING SET problem is solvable in time $Poly(n) \cdot 2^m$ with n variables and m sets (By Proposition 1), S^*_a can be found in $2^{\mathcal{O}(k)}$ time. □

All the sets $\{S^*_a \mid a \in \mathfrak{C}(\overline{U})\}$ can be found in time at most $c \cdot poly(n) \cdot 2^{\mathcal{O}(k)}$, leading to $\mathcal{O}^*(k^{\mathcal{O}(k)})$ overall running time.

Theorem 1.9. *MCS is FPT parameterized by vertex cover number, admitting an algorithm running in time $\mathcal{O}^*(k^{\mathcal{O}(k)})$, where k is the size of the vertex cover.*

MCS Parameterized by Neighborhood Diversity/Types of Vertices

We are given a graph $G = (V, E)$. Let $V = \bigsqcup_{i \in [r]} T_i$ be a neighborhood decomposition of a graph G of minimum size. Note that, $\forall i \in [r]$, the induced graph $G[T_i]$ is either an independent set or a clique, and for distinct $i, j \in [r]$, either there is no edge between T_i and T_j (or) all possible edges exist between vertices in T_i and T_j.

CONSISTENT SUBSET PROBLEM PARAMETERIZED BY NEIGHBORHOOD DIVERSITY

Input: A graph $G = (V = \bigsqcup_{i \in [r]} T_i, E)$ where for each $u, v \in T_i$, $N(u) \setminus \{v\} = N(v) \setminus \{u\}$ along with a coloring function $\mathfrak{C} : V(G) \to [c]$.
Question: Compute a minimum consistent subset (MCS) S for $(G, \mathfrak{C})$.
Parameter: r

We show that MCS is FPT parameterized by neighborhood diversity r. To that end, we prove the following claim, which we use in the correctness proof of our algorithm at the end of this section.

Claim 1.10. *(†)[1] Given a graph $G = (V, E)$ with neighborhood diversity r (i.e., $V = \bigsqcup_{i \in [r]} T_i$), there is an MCS S for $(G, \mathfrak{C})$ such that for each type T_i and for each color j, the set S has $0, 1$ or all the vertices of color j from T_i. Formally, $\forall i \in [r]$ and $\forall j \in [c]$, we have $|T_i \cap \mathfrak{C}^{-1}(j) \cap S| \in \{0, 1, |(T_i \cap \mathfrak{C}^{-1}(j))|\}$.*

The above claim essentially states that there exists a minimum consistent subset (MCS) that, for each color from any type, includes either 0, 1, or all vertices of that color. With this claim in place, we are now ready to present the first step of our algorithm.

Step 1: Identifying the Nature of Responsible Colors

We start by defining *partitions* and sets of *responsible colors* with respect to a potential MCS S below. Notice that while we may guess (i.e., generate all possible) partitions required for a desired MCS, generating all sets of responsible colors may not be possible in FPT time. We use a clever approach to bypass the exhaustive generation of responsible color sets, as described at the end of these definitions.

Partitions (w.r.t. an MCS S): We begin by guessing a partition $\mathcal{T}$ of the r types into 3 sets, namely $\mathcal{T}_0$, $\mathcal{T}_1$, and $\mathcal{T}_2$ with respect to a potential MCS S for $(G, \mathfrak{C})$ as follows:

- $\mathcal{T}_0 = \{T_i \mid i \in [r] \text{ and } T_i \cap S = \emptyset\}$
- $\mathcal{T}_1 = \{T_i \mid i \in [r] \text{ and } |\mathfrak{C}(T_i \cap S)| = 1\}$
- $\mathcal{T}_2 = \{T_i \mid i \in [r] \text{ and } |\mathfrak{C}(T_i \cap S)| > 1\}$

In other words, $\mathcal{T}_0$ is the set of types that contain no vertex from S, $\mathcal{T}_1$ is the set of types from which all vertices selected into S are of the same color, and $\mathcal{T}_2$ is the set of types from which vertices of multiple colors are selected into S.

Responsible Colors (w.r.t an MCS S): Given an MCS S and a corresponding 3-partition $\mathcal{T}$, a small inclusion-wise-minimal set of colors $\mathcal{R}$ is a set of *responsible colors* if and only if it satisfies the following.

- For each type $T_i \in \mathcal{T}_1$, $\mathcal{R}$ contains the color $\mathfrak{C}(T_i \cap S)$.
- For each type in $\mathcal{T}_2$, the set $\mathcal{R}$ contains at least two distinct colors from $\mathfrak{C}(T_i \cap S)$.

Observe that any set of responsible colors has size at most $2r$, due to the minimality property. Moreover, any such set is sufficient to determine the partition $\mathcal{T}$ of types. And, for a given S, there may exist multiple sets of responsible colors, possibly more than polynomially (in n) many and finding one may not even be possible in FPT time. Nevertheless, let $R = \{\mathsf{c}_1, \ldots, \mathsf{c}_k\}$ denote an arbitrary set of responsible colors for S where $k \leq 2r$. We prove the following property of a set of responsible colors which we use in the final correctness proof of our algorithms. The property is that basically for every vertex v, its closest distance to a solution vertex in S can be determined (same as) by its closest distance to a solution vertex whose color is from the set of responsible colors.

Claim 1.11. *For a set of responsible colors R of S and an arbitrary vertex v, $d(v, S \setminus (S \cap \mathfrak{C}^{-1}(\mathfrak{C}(v)))) = d(v, S \cap (\cup_{j \in R \setminus \mathfrak{C}(v)} \mathfrak{C}^{-1}(j)))$.*

Proof. Let z be a vertex in S of a color other than $\mathfrak{C}(v)$ such that the distance from v to z is minimized over all vertices in S whose colors are different from that of v, i.e.,

[1] Proofs for Theorems, Lemmas, and Claims marked with † have been moved to the extended version due to space limitations.

$d(v, z) = d(v, S \setminus (S \cap \mathfrak{C}^{-1}(\mathfrak{C}(v))))$. If $z \in \mathcal{T}_1$, then by definition of $\mathcal{T}_1$, we have $\mathfrak{C}(z) \in R$, satisfying the claim. Otherwise, if $z \in T_i$ for some $T_i \in \mathcal{T}_2$ and $\mathfrak{C}(z) \notin R$, then by the definition of responsible colors, there must exist a $y \in S \cap T_i$ of a different color (i.e., $\mathfrak{C}(y) \in R$, $\mathfrak{C}(y) \neq \mathfrak{C}(v)$). But then, we have $d(v, z) = d(v, y)$, proving the claim . □

We reiterate that although we may not be able to decide on an R, we can guess whether the vertices corresponding to the colors in R are included in the solution from each type as described below.

Guessing solution occurrences (nature) of colors in R: We guess the solution occurrence of each responsible color $\mathsf{c} \in R$ in each type via a function $occ : R \to 2^{[r]}$, where $occ(\mathsf{c})$ is the set of types that have vertices in $S \cap \mathfrak{C}^{-1}(\mathsf{c})$. A *valid* occurrence function occ must be consistent with the following partitioning requirements consistent with S and $\mathcal{T}$.

- For each type $T_i \in \mathcal{T}_0$, there is no $j \in [k]$ such that $i \in occ(\mathsf{c}_j)$.
- For each type $T_i \in \mathcal{T}_1$, there is precisely one $j \in [k]$ such that $i \in occ(\mathsf{c}_j)$.
- For each type $T_i \in \mathcal{T}_2$, there are at least two colors $\mathsf{c}_{j_1}, \mathsf{c}_{j_2} \in R$ such that $i \in occ(\mathsf{c}_{j_1}) \cap occ(\mathsf{c}_{j_2})$.

At the end of Step 1, we assume that we have correctly fixed a partition $\mathcal{T}$ (with respect to a potential MCS S), along with a consistent and valid occurrence function occ for some arbitrary set of responsible colors R (for S).

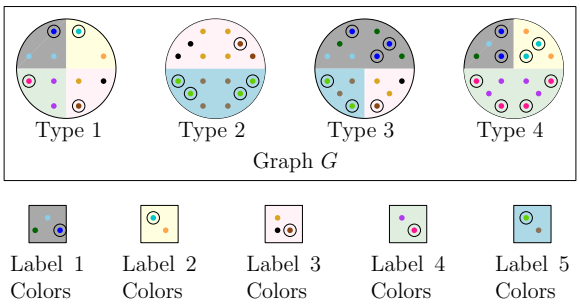

Figure 1: Each disk represents a type in the graph G. Colors are grouped into labels, and each level is indicated by a distinct background color. From each label, a representative color is selected, shown as a point encircled by a circle.

Step 2: Label Coding to Identify a Most Suitable Set of Responsible Colors

In this step, we apply a label-coding function that assigns each color in R a distinct label with *sufficiently high* probability. This allows us to break the problem into simpler subproblems, each of which is structurally easier, solvable in $f(r) \cdot n^{\mathcal{O}(1)}$ time, and independent of the others. In each subproblem, we search for the most appropriate color that can assume the role of a responsible color from R. Since the input instance already associates colors with vertices, we use the term "label coding" instead of "color coding" to avoid confusion, although the two are essentially equivalent.

Formally, this step aims to identify a set of actual colors from the input that can take on the roles of the guessed responsible colors, in a manner consistent with the guessed function occ, and compute the smallest possible consistent subset that realizes this correspondence. We proceed as follows.

Label Coding: We label-code (Cygan et al. 2015) all the colors using k labels and partition the color set $[c]$ as $[c] = C_1 \uplus \cdots \uplus C_k$, such that with *high* probability, each responsible color c_i of R gets the label i. We call such an event a *nice* label-coding. Following a *nice* label-coding, our goal becomes to identify the best responsible (a choice that gives the smallest possible consistent subset) color c_js from each C_i that aligns with the guessed/chosen partitioning constraints and occ function.

Caution Constraints: We select the most suitable responsible color for each C_i, where $i \in [k]$ in the next step. While selecting these most suitable responsible colors independently from each C_i, we impose the following *caution constraints* to ensure correctness. In our (desired) solution, in each C_i,

C1: There is at least one color (denoted by $c_{i'}$) that has solution vertices precisely in all types of $occ(\mathsf{c}_i)$.

C2: There is exactly one color $c_{i'}$ that has solution vertices from any type $T_j \in \mathcal{T}_1$ where $T_j \in occ(\mathsf{c}_i)$.

C3: There is no color $c_{i'}$ that has solution vertices from any type in $\mathcal{T}_0 \cup (\mathcal{T}_1 \setminus occ(\mathsf{c}_i))$.

These *caution constraints* together ensure that, in the desired solution we aim to construct, the types corresponding to the selected vertices satisfy the identified (or guessed) partition $\mathcal{T}$.

Step 3: Selection of a Best Responsible Color from Each Label with Caution Constraints

To determine the most suitable responsible color from each C_i and combine them to return an MCS, we crucially exploit the fact that the subproblems of selecting responsible colors from each C_i are *independent*. At a high level, this independence arises because the partition $\mathcal{T}$, determined by the occurrence function occ over R, essentially fixes, for every vertex, the distance of closest solution vertex of a different color. This, in turn, determines the *minimum number of vertices* required from that particular color in the solution to ensure *consistency* for all vertices of the same color. Since both the partition $\mathcal{T}$ and the function occ are already fixed, we can compute, for each individual color, the smallest subset of vertices that must be included in a solution, as long as the occ function requirements from C_i and the *caution constraints* are satisfied. Below we describe the exact procedure along with a formal correctness argument for the same.

For a fixed label C_i, we go over each $c_j \in C_i$ expecting it to be a most suitable responsible color from C_i and compute the size of a smallest set of vertices required to be in the solution for the consistency of all vertices of colors in C_i. First, from $occ(\mathsf{c}_i)$, we determine the types from which vertices of color c_j are to be selected into S. Recall from Claim 1.10, either one or all vertices of color c_j for each of

the types in $occ(\mathsf{c}_i)$ are selected into a potential solution S. Let O_j be the set of all possible subsets of vertices of color c_j that may appear in S in accordance with $occ(\mathsf{c}_i)$. Thus, $|O_j| \leq 2^r$. For a fixed $o \in O_j$, let $n'_{j,o}$ ($|o|$) be the number of vertices of color c_j in S and $n_{k,o}$ be the minimum number of vertices of color $c_k \in C_i \setminus \{c_i\}$ (again we have at most 3^k such choices) one has to pick into a solution of color c_k adhering to caution constraints while satisfying the consistency requirement of all vertices of color c_k and of all the vertices of color c_j (with respect to the choice o).

We formally check the consistency requirements as follows (in addition to caution constraints). For a choice o of color c_j and any arbitrary subset o' of color c_k (at most 3^k many such choices) that are to be selected into a potential solution, we must ensure that:

$$d(v, o') \leq \min\{d(v, T_i) \mid T_i \in \mathcal{T}_1 \cup \mathcal{T}_2\} \quad \forall v \text{ of color } c_k \tag{1}$$

$$d(u, o) \leq d(u, o') \quad \forall\, u \text{ of color } c_j. \tag{2}$$

Equation (1) ensures the consistency requirement for all vertices of color c_k, and Equation (2) ensures consistency of all vertices of color c_j with respect to color c_k. Note that we do not have to worry about the *consistency requirements* between two colors c_k and c'_k; since ensuring that each color is consistent with respect to a responsible set of colors (in (1)) is sufficient for it to be consistent with all the colors, due to Claim 1.11. Let S_i be a smallest subset of solution vertices of colors in C_i that satisfy the caution constraints along with the above mentioned consistency requirements, i.e.,

$$|S_i| = \min_{c_j \in C_i}\{\min_{o \in O_j}\{n'_{j,o} + \sum_{c_k \in C_i \setminus \{c_j\}} n_{k,o}\}\}$$

We return $S = \bigcup_{i \in k} S_i$ as the desired MCS. Before presenting our final algorithm, we provide a correctness proof of the above statement by establishing the independence of the subproblems, specifically, that the selection of the best responsible color from each C_i can be done independently. The following lemma essentially states that the smallest possible set of solution vertices of colors from C_i, satisfying the caution constraints and consistency requirements, can substitute the vertices of the same colors in an MCS without violating consistency of any vertex or increasing the solution size.

Lemma 1.12. *(†) For any MCS S with partition $\mathcal{T}$, occ, a set of responsible colors R, and following a nice label coding $[c] = C_1 \uplus \cdots \uplus C_k$, let S_i be a smallest possible set of vertices selected from all the colors in C_i with c_j being the responsible color, while adhering to the caution constraints and consistency requirements. $S' = S \cap (\cup_{j \notin C_i} \mathfrak{C}^{-1}(j)) \cup S_i$ is also an MCS.*

Lemma 1.12 ensures that one can compute each S_i of minimum possible size from the corresponding label C_i, independently of the others, and combine them to obtain a desired Minimum Consistent Subset (MCS). A formal algorithm is presented in Algorithm 1.

Runtime Analysis: The algorithm branches over 3^r partitions (choices for $\mathcal{T}$) and $r^{\mathcal{O}(r)}$ possible *occ* functions, each

Algorithm 1: MCS parameterized by Neighborhood Diversity

```
Generate all 3-partitions of the types into T_0, T_1, and T_2.
Generate all valid occurrence functions occ.
for each fixed partition and valid occ do
    Label-code the colors [c] using k labels (k ≤ 2r), and
    partition [c] = C_1 ⊎ ··· ⊎ C_k based on the labels the
    colors receive.
    for i = 1 to k do
        for each c_j ∈ C_i do
            Let c_j be the responsible color in C_i.
            for each o ∈ O_j do
                Compute n'_{j,o} and the corresponding set of
                vertices (call it S'_{j,o}).
                for each c_k ∈ C_i \ {c_j} do
                    Compute n_{k,o} (as described in Step 3) and
                    its vertex set S'_{k,o}.
                end for
            end for
            Keep track of the S'_j = S'_{j,o} ∪
            (⋃_{c_k ∈ C_i \ {c_j}} S'_{k,o}) which minimizes
            n'_{j,o} + Σ_{c_k ∈ C_i \ {c_j}} n_{k,o}.
        end for
        S_i ← arg min_{S'_j : c_j ∈ C_i} |S'_j|.
    end for
    Keep track of S ← ⋃_{i ∈ [k]} S_i of minimum cardinality.
end for
return S.
```

verifiable in polynomial time. A random labeling yields a *nice* label-coding with probability at least k^{-k}, and such codings can be enumerated in $k^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)}$ time. For each component C_i and a responsible color c_j, there are $c \leq n$ choices, and at most 2^r options for $|O_j|$. For each $o \in O_j$, the value $n'_{j,o}$ can be computed in polynomial time. For each of the non-responsible colors c_k, values $n_{k,o}$ can be computed in $3^r \cdot \text{poly}(n)$ time by enumerating all $S \cap \mathfrak{C}^{-1}(c_k)$. Thus, the total runtime is $3^r \cdot r^{\mathcal{O}(r)} \cdot k^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)} \cdot kc \cdot 3^r \cdot n^{\mathcal{O}(1)} \cdot c \cdot 3^r \cdot n^{\mathcal{O}(1)} = r^{\mathcal{O}(r)} \cdot n^{\mathcal{O}(1)}$, where the final bound follows from $k \leq 2r$ and $c \leq n$.

The randomization step (label-coding) can be *derandomized with (n, k)-universal sets* (Cygan et al. 2015), while maintaining the same asymptotic running time.

Theorem 1.13. *MCS is FPT parameterized by neighborhood diversity, admitting an algorithm running in time $\mathcal{O}^*(r^{\mathcal{O}(r)})$, where r is the neighborhood diversity of the input graph.*

Acknowledgments

The first author acknowledges support from the Science and Engineering Research Board (SERB) via the project MTR/2022/000253. The second author would like to thank Akanksha Agrawal (IIT Madras) for formally introducing him to the field of parameterized algorithms, a foundation that helped shape this work.

References

Arimura, H.; Gima, T.; Kobayashi, Y.; Nochide, H.; and Otachi, Y. 2023. Minimum Consistent Subset for Trees Revisited. *CoRR*, abs/2305.07259.

Banik, A.; Das, S.; Maheshwari, A.; Manna, B.; Nandy, S. C.; M., K. P. K.; Roy, B.; Roy, S.; and Sahu, A. 2024. Minimum Consistent Subset in Trees and Interval Graphs. In Barman, S.; and Lasota, S., eds., *44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2024, December 16-18, 2024, Gandhinagar, Gujarat, India*, volume 323 of *LIPIcs*, 7:1–7:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Biniaz, A.; and Khamsepour, P. 2024. The Minimum Consistent Spanning Subset Problem on Trees. *J. Graph Algorithms Appl.*, 28(1): 81–93.

Chitnis, R. 2022. Refined Lower Bounds for Nearest Neighbor Condensation. In Dasgupta, S.; and Haghtalab, N., eds., *International Conference on Algorithmic Learning Theory, 29 March - 1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, 262–281.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer. ISBN 978-3-319-21274-6.

Dey, S.; Maheshwari, A.; and Nandy, S. C. 2021. Minimum Consistent Subset Problem for Trees. In Bampis, E.; and Pagourtzis, A., eds., *Fundamentals of Computation Theory - 23rd International Symposium, FCT 2021, Athens, Greece, September 12-15, 2021, Proceedings*, volume 12867 of *Lecture Notes in Computer Science*, 204–216. Springer.

Dey, S.; Maheshwari, A.; and Nandy, S. C. 2023. Minimum consistent subset of simple graph classes. *Discret. Appl. Math.*, 338: 255–277.

Diestel, R. 2012. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer. ISBN 978-3-642-14278-9.

Hart, P. E. 1968. The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, 14(3): 515–516.

Khodamoradi, K.; Krishnamurti, R.; and Roy, B. 2018. Consistent Subset Problem with Two Labels. In Panda, B. S.; and Goswami, P. P., eds., *Algorithms and Discrete Applied Mathematics - 4th International Conference, CALDAM 2018, Guwahati, India, February 15-17, 2018, Proceedings*, volume 10743 of *Lecture Notes in Computer Science*, 131–142. Springer.

Lampis, M. 2012. Algorithmic Meta-theorems for Restrictions of Treewidth. *Algorithmica*, 64(1): 19–37.

Manna, B. 2024a. Minimum Consistent Subset in Interval Graphs and Circle Graphs. *CoRR*, abs/2405.14493.

Manna, B. 2024b. Minimum Strict Consistent Subset in Paths, Spiders, Combs and Trees. *CoRR*, abs/2405.18569.

Matsumoto, N.; Kurita, K.; and Kiyomi, M. 2025. Space-Efficient FPT Algorithms for Degeneracy. *IEICE Trans. Inf. Syst.*, 108(3): 208–213.

Wilfong, G. 1991. Nearest Neighbor Problems. In *Proceedings of the Seventh Annual Symposium on Computational Geometry*, SCG '91, 224–233.