Minimum Selective Subset on Some Graph Classes

Bubai Manna*

Abstract

In a connected simple graph $G = (V(G), E(G))$, each vertex is assigned a color from the set of colors $C = \{1, 2, \ldots, c\}$. The set of vertices is partitioned as $V(G) = \bigcup_{\ell=1}^{c} V_\ell$, where all vertices in V_ℓ share the same color ℓ. A subset $S \subseteq V(G)$ is called *Selective Subset* if, for every vertex $v \in V(G)$, if $v \in V_\ell$, at least one of its nearest neighbors in $S \cup (V(G) \setminus V_\ell)$ has the same color as v. The *Minimum Selective Subset* (MSS) problem seeks to find a selective subset of minimum size. The problem was first introduced by Wilfong in 1991 [18] for a set of points in the Euclidean plane, where two major problems, MCS (Minimum Consistent Subset) and MSS, were proposed.

In graph algorithms, the only known result is that the MSS problem is NP-complete, as shown in [2] in 2018. Beyond this, no further progress has been made to date. In contrast, the MCS problem has been widely studied in various graph classes over the years. Therefore, in this work, we also extend the algorithmic study of MSS on various graph classes. We first present a $\mathcal{O}(\log n)$-approximation algorithm for general graphs with n vertices and regardless of the number of colors. We also show that the problem remains NP-complete even for planar graphs when restricted to just two colors. Finally, we provide linear-time algorithms for computing optimal solutions in trees and unit interval graphs for any number of colors[1].

1 Introduction

Many supervised learning methods use a colored training data set T in a metric space (X, d), where each element $t \in T$ has a color from the set of colors $C = \{1, 2, \ldots, c\}$. The goal is to find a subset $S \subseteq T$ with minimum cardinality such that every element of T is either in S or has at least one nearest neighbor in S with the same color. This problem, known as the *Minimum Consistent Subset* (MCS), was first introduced by Hart [13], whose work has received more than 2,800 citations.

The problem is NP-complete for three or more colors [18] and remains NP-complete for two colors [14] in $\mathbb{R}^2$. It is also W[1]-hard when parameterized by output size [6]. Various algorithms for the problem MCS in $\mathbb{R}^2$ have been proposed [2,4,6,18], highlighting its significance in machine learning and computational geometry. However, the problem MCS is closely related to MSS which is discussed by Wilfond [18].

The problem MSS plays a crucial role in optimizing data selection by identifying the smallest subset that preserves essential information. This is particularly useful in applications such as fingerprint recognition, character recognition, and pattern recognition, where it helps reduce redundancy and improve decision-making in classification and feature selection tasks.

Wilfong [18] proved that MSS is also NP-complete even with two colors in $\mathbb{R}^2$. Recently, [2] established an PTAS with c-color points and showed that the problem is W[2]-hard when parameterized by the size of the solution, while MSS is contained in W[1] when the number of colors is two in $\mathbb{R}^2$.

1.1 Notations and Definitions

For any graph $G = (V(G), E(G))$, we denote the set of vertices by $V(G)$ and the set of edges by $E(G)$. Without loss of generality, we use $[n]$ to denote the set of integers $\{1, \ldots, n\}$. We use an arbitrary vertex color function $C : V(G) \to [c]$, such that each vertex is assigned exactly one color from the set $[c]$. For a subset of vertices $U \subseteq V(G)$, let $C(U)$ represent the set of colors of the vertices in U, formally defined as $C(U) = \{C(u) \mid u \in U\}$.

For any two vertices $u, v \in V(G)$, the shortest path distance between u and v in G is denoted by $d(u, v)$. $d(u, v)$ is called *hop-distance* between u and v. For a vertex $v \in V(G)$, the distance between v and the set $U \subseteq V(G)$ in G is given by $d(v, U) = \min_{u \in U} d(v, u)$.

The nearest neighbors of v in the set U is denoted as $\hat{\mathrm{N}}(v, U)$, formally defined as

$$\hat{\mathrm{N}}(v, U) = \{u \in U \mid d(v, u) = d(v, U)\}.$$

Therefore, if $v \in U$, then $\hat{\mathrm{N}}(v, U) = \{v\}$.

$G[U]$ denotes the subgraph of G induced by $U \subseteq V(G)$, and $|U|$ is the cardinality of U. We use standard graph-theoretic notation and symbols as presented in [12].

Suppose $G = (V(G), E(G))$ is a given connected and undirected graph, where the vertices are partitioned into

*Indian Institute of Technology, Kharagpur, bubaimanna11@gmail.com

[1]The full version of this work can be found in [17].

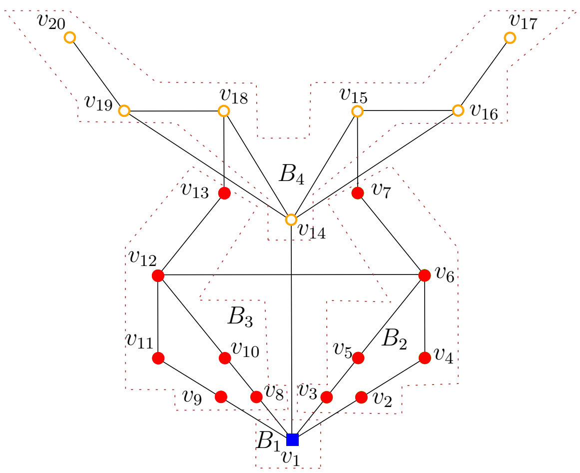

Figure 1: Colors: *blue= square, red= disk, orange=fdisk* and $V(G) = V_{blue} \cup V_{red} \cup V_{orange}$, where $V_{blue} = \{v_1\}$, $V_{red} = \{v_2, \ldots, v_{13}\}$, and $V_{orange} = \{v_{14}, \ldots, v_{20}\}$. The sets $\{v_1, v_4, v_5, v_{10}, v_{11}, v_{17}\}$ and $\{v_1, v_3, v_4, v_8, v_{11}, v_{17}\}$ both are MCS. Similarly, $\{v_1, v_2, v_3, v_7, v_8, v_9, v_{13}, v_{14}\}$ and $\{v_1, v_4, v_5, v_7, v_{10}, v_{11}, v_{13}, v_{14}\}$ are also MSS. Brown-dotted regions indicate the blocks. The complete list of blocks is $B_1 = \{v_1\}$, $B_2 = \{v_2, \ldots, v_7\}$, $B_3 = \{v_8, \ldots, v_{13}\}$, $B_4 = \{v_{14}, \ldots, v_{20}\}$.

c color classes, namely $V_1, V_2, \ldots, V_c$. This means that each vertex in $V(G)$ has a color from the set $[c]$, and each vertex in V_ℓ has color ℓ. A *Minimum Consistent Subset* (MCS) is a subset $S \subseteq V(G)$ of minimum cardinality such that for every vertex $v \in V(G)$, if $v \in V_\ell$, then

$$\hat{\mathrm{N}}(v, S) \cap V_\ell \neq \emptyset.$$

The definition of a selective subset is as follows:

Definition 1 *A subset $S \subseteq V(G)$ is called a* Minimum Selective Subset (MSS) *if, for each vertex $v \in V(G)$, if $v \in V_\ell$, the set of nearest neighbors of v in $S \cup (V(G) \setminus V_\ell)$, contains at least one vertex u such that $C(v) = C(u)$, and $|S|$ is minimum.*

In other words, we are looking for a vertex set $S \subseteq V(G)$ of minimum cardinality such that every vertex v has at least one nearest neighbor of the same color in the graph, excluding those vertices of the same color as v that are not in S.

Figure 1 illustrates that MCS and MSS are distinct and may not be unique for a given graph. The selective subset problem on graphs is defined as follows:

Selective Subset Problem on Graphs

Input: A graph $G = (V(G), E(G))$, a color function $C : V(G) \to [c]$, and an integer s.
Question: Does there exist a selective subset of size $\leq s$ for (G, C)?

Banerjee et al. [2] proved that MCS is W[2]-hard [7] when parameterized by the minimum consistent set size, even with two colors in general graphs. Dey et al. [9–11] provided polynomial-time algorithms for MCS on some simple graph classes including path, spider, caterpillar, comb and trees (for trees, $c = 2$). XP and NP-complete as well as the FPT algorithms (when c is a parameter) on trees, can be found in [1, 3]. MCS is also NP-complete in interval graphs [3] and APX-hard in circle graphs [15]. Variants such as the *Minimum Consistent Spanning Subset* (MCSS) and the *Minimum Strict Consistent Subset* (MSCS) of MCS have been studied in trees [5, 16]. However, the algorithmic results for MSS have not been extensively studied to date. Banerjee et al. [2] only showed that MSS is NP-complete in general graphs.

1.2 Results

Since only a hardness result for MSS in general graphs is known, our work provides new insights into the complexity and approximability of the problem, identifying cases where efficient approximations are achievable and where hardness persists. In Section 3, we present a $\mathcal{O}(\log n)$-approximation algorithm for MSS in general graphs, where the number of vertices is n and regardless of the number of colors.

Planar graphs are fundamental structures in graph theory and computational geometry, with many practical applications and rich structural properties. Therefore, in Section 4, we show that the problem MSS remains NP-complete even when restricted to planar graphs with just $c = 2$, highlighting the inherent difficulty of the problem even on well-behaved graph classes.

Trees are simple graph classes that often admit efficient algorithms, even when such algorithms are not possible for more general graph families. Unit interval graphs, which model intervals of equal length on the real line, are widely studied due to their applications in scheduling and biology and their tractable structure. Since both of these graph classes have been well explored in MCS [1, 3], it is natural and necessary to investigate MSS in the same classes. Therefore, in Sections 5 and 6, respectively, we present linear-time algorithms for finding optimal solutions to the MSS problem on trees and unit interval graphs with any number of colors. All proofs of (*)-marked results are in the full version [17].

2 Preliminaries

If all the vertices of a graph G are of the same color (that is, G is monochromatic), then any vertex of the graph is an MSS. Moreover, at least one vertex from each color class must be included in every selective subset;

otherwise, consider a vertex $v \in V_\ell$, and all vertices in $\hat{\mathrm{N}}(v, S \cup (V(G) \setminus V_\ell))$ must have a different color from that of v, violating the condition for a selective subset.

Definition 2 *A block is defined as a maximal connected subgraph whose vertices share the same color.*

Figure 1 illustrates an example of the blocks.

Lemma 1 *Any selective subset must contain at least one vertex from each block.*

Proof. Suppose, for contradiction, that M is a selective subset of a graph G, but there exists a block B_i such that $M \cap B_i = \emptyset$. Let the color assigned to the vertices of B_i be ℓ, and let $v \in B_i$ be any vertex.

Therefore, there must exist a nearest neighbor (say u) of v in $M \cup (V(G) \setminus V_\ell)$ such that $C(u) = C(v)$; otherwise, M would not be a selective subset. Since $M \cap B_i = \emptyset$, we have $u \notin B_i$. Thus, $u \in B_j$ for some $j \neq i$.

Let P denote the shortest path between v and u. As B_i and B_j are distinct blocks and $C(v) = C(u)$, there must exist at least one vertex $w \in P$ such that $C(w) \neq C(u)$. This implies that w lies closer to v than u does, i.e., $d(v, w) < d(v, u)$, and $w \in M \cup (V(G) \setminus V_\ell)$ with $C(w) \neq C(u)$.

Hence, instead of u, w is the nearest neighbor of v in $M \cup (V(G) \setminus V_\ell)$ with $C(w) \neq C(v)$, violating the definition of a selective subset. This contradiction completes the proof. □

The above proof indicates that, for any vertex $v \in B_i$, either v itself must belong to M, or there must exist a vertex $u \in B_i$ such that $u \in M$ and u is the nearest neighbor of v in $M \cup (V(G) \setminus V_\ell)$. In other words, each vertex $v \in B_i$ must have its nearest neighbor in $M \cup (V(G) \setminus V_\ell)$ that also lies within B_i. Therefore, we have the following observation:

Observation 1 *The blocks are independent of each other in the solution of a selective subset.*

3 $\mathcal{O}(\log n)$-Approximation Algorithm of MSS in General Graphs

Let $G = (V(G), E(G))$ be a graph with c colors and B_1, B_2, ..., B_k be the blocks in G. The *Set Cover* problem is defined as follows: Given a universe I and a collection of m subsets S, the goal is to select a minimum number of subsets from S whose union covers all elements in I. The reduction of MSS in G to the *set cover* problem is as follows (see Figure 2).

Reduction. Let $B_{i,1} \subseteq B_i$ be the set of vertices adjacent to at least one vertex of a different color, for each $i = 1, \ldots, k$. For each vertex $v \in B_i \setminus B_{i,1}$, if v in adjacent to at least one vertex of $B_{i,1}$, include v in $B_{i,2}$. Let $B^{all}_{i,1} = \bigcup_{i=1}^{k} B_{i,1} = \{a_1, \ldots, a_{n_1}\}$ and $B^{all} = \bigcup_{i=1}^{k} (B_{i,1} \cup B_{i,2}) = \{b_1, \ldots, b_{n_2}\}$ where $n_1 \leq n$ and $n_2 \leq n$. Therefore, each b_j is a vertex of either $B_{i,1}$ or $B_{i,2}$ for $j = 1, \ldots, n_2$ and for some $i \in \{1, \ldots, k\}$.

Construct a universe set $I = \{e_1, \ldots, e_{n_1}\}$, where each e_i corresponds to vertex $a_i \in B^{all}_{i,1}$. For each $b_i \in B^{all}$, define a set $S_i \subseteq I$ such that $e_j \in S_i$ if the corresponding vertex a_j and the vertex b_i are either the same vertex or adjacent and in the same block. Let $S = \{S_1, \ldots, S_{n_2}\}$.

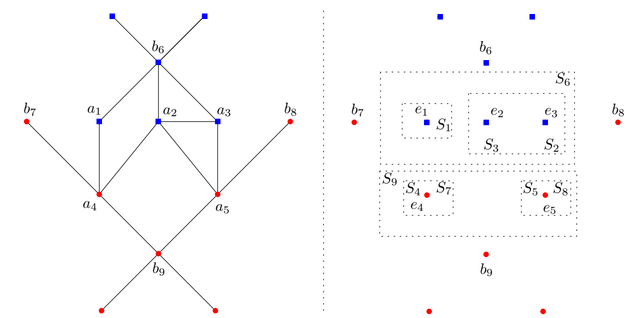

Figure 2: Reduction to set cover. Colors: *blue= square*, *red= disk*. Sets are $B_{1,1} = \{a_1, a_2, a_3\}$, $B_{2,1} = \{a_4, a_5\}$, $B_{1,2} = \{b_6\}$, $B_{2,2} = \{b_7, b_8, b_9\}$. Sets are $B^{all}_{i,1} = \{a_1, \ldots, a_5\}$, $B^{all} = \{b_1, \ldots, b_9\}$, where $b_i = a_i$ for $i = 1, \ldots, 5$. The set elements for the set cover problem is $I = \{e_1, \ldots, e_5\}$ where $e_i = b_i = a_i$ for $i = 1, \ldots, 5$. The subsets are given $S_1 = \{e_1\}$, $S_2 = S_3 = \{e_2, e_3\}$, $S_4 = S_7 = \{e_4\}$, $S_5 = S_8 = \{e_5\}$, $S_6 = \{e_1, e_2, e_3\}$, $S_9 = \{e_4, e_5\}$. The elements inside the dotted rectangles represent the sets. The selective subset is $M = \{b_6, b_9\}$, and the corresponding set cover is $\{S_6, S_9\}$.

Lemma 2 * *If a set cover of I uses some sets of S, then the vertices represented by those sets form a selective subset, and vice versa.*

Theorem 3 *The Minimum Selective Subset problem admits an $\mathcal{O}(\log n)$-approximation in general graphs.*

Proof. By Lemma 2, the MSS problem reduces to a Set Cover instance where the universe size is $n_1 \leq n$, with $n = |V(G)|$.

The greedy algorithm for Set Cover achieves an $\mathcal{O}(\log n_1)$-approximation. Since $n_1 \leq n$, it follows that $\log n_1 \leq \log n$, and therefore the approximation factor becomes $\mathcal{O}(\log n)$ in terms of the original graph size.

As the reduction preserves the approximation guarantee, this yields an $\mathcal{O}(\log n)$-approximation algorithm for the MSS problem in general graphs. □

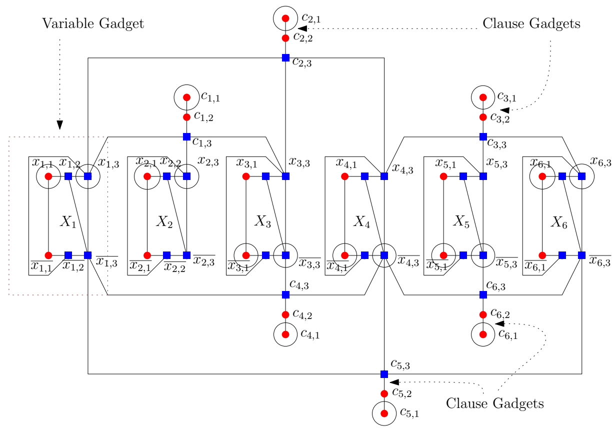

Figure 3: Reduction from PRM-3SAT to MSS in planar graph. Colors: *blue= square*, *red= disk*. The SAT expression is $\theta = c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5 \wedge c_6$. The clauses are $c_1 = (x_1 \vee x_2 \vee x_3), c_2 = (x_1 \vee x_3 \vee x_4), c_3 = (x_4 \vee x_5 \vee x_6), c_4 = (\overline{x_1} \vee \overline{x_3} \vee \overline{x_4}), c_5 = (\overline{x_1} \vee \overline{x_4} \vee \overline{x_6}), c_6 = (\overline{x_4} \vee \overline{x_5} \vee \overline{x_6})$ with variables $x_1 = x_2 = x_6 = 1$ and $x_3 = x_4 = x_5 = 0$. The vertices inside small circles are in the selective subset.

4 NP-Hardness of MSS in Planar Graphs

We reduce an instance of Planar Rectilinear Monotone 3-SAT (PRM-3SAT) θ to a bichromatic planar graph. The definition of PRM-3SAT is provided in [17]. In [8], it is shown that PRM-3SAT is NP-complete. It is also shown that, given a PRM-3SAT formula θ, its embedding described in [8] can be obtained in polynomial time.

Reduction. We embed an instance of PRM-3SAT formula θ with n variables $x_1, x_2, \ldots, x_n$ and m clauses $c_1, c_2, \ldots, c_m$ into a bichromatic planar graph $G = (V(G), E(G))$ (see Figure 3).

Variable gadget: For each variable x_i ($1 \le i \le n$), we construct a variable gadget X_i shown as a brown dotted rectangle in Figure 3, as follows: A path $(x_{i,1}, x_{i,2}, x_{i,3})$ of length two, where $C(x_{i,1}) =$ *red* and $C(x_{i,2}) = C(x_{i,3}) =$ *blue*, is called the *positive literal path* of x_i. Similarly, a path $(\overline{x_{i,1}}, \overline{x_{i,2}}, \overline{x_{i,3}})$ of length two, where $C(\overline{x_{i,1}}) =$ *red* and $C(\overline{x_{i,2}}) = C(\overline{x_{i,3}}) =$ *blue*, is called the *negative literal path* of x_i.

The vertex $x_{i,1}$ is adjacent to $\overline{x_{i,1}}$, and $x_{i,2}$, $x_{i,3}$ are adjacent to $\overline{x_{i,3}}$, $\overline{x_{i,2}}$, respectively. Additionally, $x_{i,3}$ is adjacent to $\overline{x_{i,3}}$.

Clause gadget: For each clause c_j ($1 \le j \le m$), the clause gadget C_j is as follows: A path $(c_{j,1}, c_{j,2}, c_{j,3})$ of length two, where $C(c_{j,1}) = C(c_{j,2}) =$ *red* and $C(c_{j,3}) =$ *blue*, is called *clause path* for the clause c_j.

If a clause c_j consists of three positive literals x_i, x_l, x_t (i.e., $c_j = (x_i \vee x_l \vee x_t)$), then the vertex $c_{j,3}$ adjacent to $x_{i,3}$, $x_{l,3}$, and $x_{t,3}$ from their respective positive literal paths. If c_j has three negative literals $\overline{x_i}, \overline{x_l}, \overline{x_t}$, then $c_{j,3}$ is adjacent to $\overline{x_{i,3}}, \overline{x_{l,3}}, \overline{x_{t,3}}$ from the corresponding negative paths as shown in the Figure 3.

The construction of the graph G is now complete. Notably, G remains planar because its embedding (Figure 3) closely resembles the PRM-3SAT embedding described in [8]. Thus, for n variables and m clauses, the bichromatic planar graph G contains $6n + 3m$ vertices and $8n+5m$ edges. We set $V(G) = V_r \cup V_b$, where V_r is the set of *red* vertices and V_b is the set of *blue* vertices.

Lemma 4 * *θ is satisfied if and only if G has a selective subset of size $2n + m$.*

Theorem 5 *Finding a minimum selective subset is NP-complete for planar graphs with two colors.*

Proof. It is easy to see that the problem is in NP. As for NP-complete, Lemmas 4 establishes a relationship between θ and the size of the selective subset of G in polynomial time. Therefore, MSS is NP-complete in planar graphs. □

Remark. The above reduction remains valid even if the pair of vertices $\{c_{j,1}, c_{j,2}\}$ in each clause gadget is assigned a distinct (except blue color), unique color not shared across gadgets. That is, we may assign a different color to each such pair for $1 \le j \le m$, and the reduction still preserves the equivalence between satisfying the PRM-3SAT formula θ and the existence of a selective subset of size $2n + m$.

5 Linear-time Algorithm of MSS in Trees

We now describe a linear-time algorithm for finding a MSS in tree. The key idea is based on Observation 1, which tells us that we can solve the problem independently on each block. So, to simplify notation and ideas, we focus on just one block.

Let $T = (V(T), E(T))$ be a tree rooted at a vertex r with $|V(T)| = n$ and a total of c colors. For each block B, we compute a minimum selective subset M_B using Lemma 1. The algorithm has two phases: initialization (Algorithm 1 in [17]) and selection (Algorithm 2 in [17]). The final solution is the union of the subsets computed for each block.

The algorithm follows these steps (see Figure 4):

Initialization (Algorithm 1).

- Start with the empty sets: $M_B := \emptyset$, $B^1 := \emptyset$, and $B^2 := \emptyset$.
- For each vertex $v \in B$, if v is adjacent to a vertex of a different color, include v in B^1.

- For each vertex $v \in B \backslash B^1$, if v is adjacent to at least one vertex in B^1, then include v in B^2.
- Define $B^{all} = B^1 \cup B^2$ (note that $B^1 \cap B^2 = \emptyset$).

Subtree Formation (refer to lines 1-4 in Algorithm 2).

- Since $B^{all} \subseteq B$, B^{all} induces one or more connected induced subtrees in T, and each subtree consists of vertices of the same color because $B^{all} \subseteq B$. We consider such induced connected maximal subtrees.
- Let $T^{v_1}, T^{v_2}, \ldots, T^{v_t}$ be such connected maximal subtrees in T formed by the vertices of B^{all} and call their roots $v_1, v_2, \ldots, v_t$, respectively. Note that, since the whole tree is rooted, a root is naturally defined for each subtree.

Selection Process (refer to lines 5-31 in Algorithm 2).

- For each subtree T^{v_j} (where $1 \leq j \leq t$), start from the *lowest-level vertex* u. A lowest-level vertex of a tree is a vertex that is farthest from the root.
- u must belong to either B^1 or B^2.
- If $u \in B^2$:
  - Remove u from B^2, B^{all}, and T^{v_j}, as it must be adjacent to a vertex in B^1.
- If $u \in B^1$:
  - If u has a parent (say v) in T^{v_j}, add v to M_B and remove v along its children from B^1, B^2, B^{all}, and T^{v_j}.
  - If u has no parent in T^{v_j}, add u to M_B and remove it from B^1, B^{all}, and T^{v_j}.
  - If the grandparent of u (if it exists in T^{v_j}) belongs to B^1, move it from B^1 to B^2, and update B^{all} accordingly.
- Repeat the **Selection Process** until T^{v_j} becomes empty. Once T^{v_j} has no vertices, increase $j \leftarrow j+1$ and repeat the **Selection Process** until $j = t$.

We apply the algorithm to each block B and obtain subsets M_B. The final solution is $M = \bigcup_{B \in T} M_B$. We now prove that M is not just selective, but also minimum.

Lemma 6 *For any block B and any vertex $u \in B^1$, either $u \in M$ or at least one adjacent vertex of u in B^{all} must belong to M.*

Proof. Let X be the set that contains u and all its adjacent vertices in B^{all}. Suppose T^{v_j} is the subtree that contains the vertices of X.

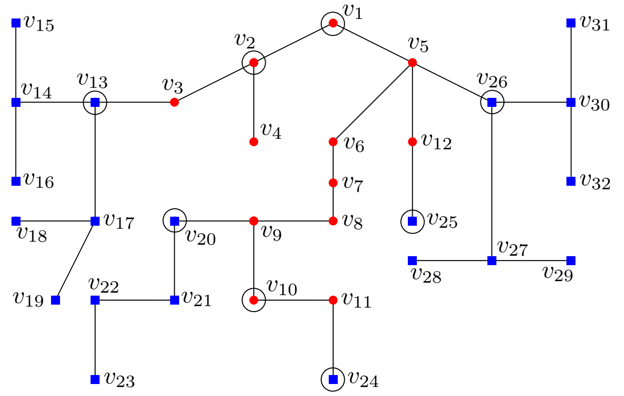

Figure 4: Colors: *blue= square*, *red= disk*. $r = v_1$ is the root of the tree T. The blocks are $B_1 = \{v_1, \ldots, v_{12}\}$, $B_2 = \{v_{13}, \ldots, v_{19}\}$, $B_3 = \{v_{20}, \ldots, v_{23}\}$, $B_4 = \{v_{24}\}$, $B_5 = \{v_{25}\}$, $B_6 = \{v_{26}, \ldots, v_{32}\}$. $B_1^1 = \{v_3, v_5, v_9, v_{11}, v_{12}\}$, $B_1^2 = \{v_1, v_2, v_6, v_8, v_{10}\}$, $B_2^1 = \{v_{13}\}$, $B_2^2 = \{v_{14}, v_{17}\}$, $B_3^1 = \{v_{20}\}$, $B_3^2 = \{v_{21}\}$, $B_4^1 = \{v_{24}\}$, $B_4^2 = \emptyset$, $B_5^1 = \{v_{25}\}$, $B_5^2 = \emptyset$, $B_6^1 = \{v_{26}\}$, $B_6^2 = \{v_{27}, v_{30}\}$. $T_1^{v_1} = \{v_1, v_2, v_3, v_5, v_6, v_{12}\}$, $T_1^{v_8} = \{v_8, v_9, v_{10}, v_{11}\}$, $T_2^{v_{13}} = \{v_{13}, v_{14}, v_{17}\}$, $T_3^{v_{20}} = \{v_{20}, v_{21}\}$, $T_4^{v_{24}} = \{v_{24}\}$, $T_5^{v_{25}} = \{v_{25}\}$, $T_6^{v_{26}} = \{v_{26}, v_{27}, v_{30}\}$. MSS $= \{v_2, v_1, v_{10}, v_{13}, v_{20}, v_{24}, v_{25}, v_{26}\}$. The vertices inside small circles are in the minimum selective subset.

By the process described in Algorithm 2, after a finite number of iterations, a vertex from X will become a lowest-level vertex of T^{v_j} (since we delete vertices in each iteration until the subtree becomes empty).

We analyze two cases:

- If u becomes a lowest-level vertex, then its parent (if it exists in T^{v_j}) must be included in M_B because $u \in B^1$, meaning that an adjacent vertex of u from B^{all} is included in M_B. If the parent does not exist, then u itself must be included in M_B according to Algorithm 2 as $u \in B^1$.
- If a vertex $v \in X \setminus \{u\}$ becomes a lowest-level vertex, we consider two subcases:
  - If $v \in B^1$, then its parent must be u, as u is the only adjacent vertex of v (as v is lowest-level vertex), and u must be included in M_B.
  - If $v \in B^2$, then v is removed from the tree. In subsequent iterations, either u eventually becomes a lowest-level vertex (in which case we proceed as above), or one of its remaining neighbors becomes a lowest-level vertex in B^1, again forcing $u \in M_B$. If all neighbors of u are in B^2 and get deleted, u itself becomes lowest-level, completing the process.

Combining all these cases, we conclude that $X \cap M_B \neq$

$\emptyset$. Thus, either $u \in M$ or at least one adjacent vertex of u in B^{all} belongs to M. □

Lemma 7 * *M is a minimum selective subset of the tree T.*

Lemma 8 * *Algorithm 2 runs in $O(n)$ time.*

Remark. Since our algorithm is described for a single block and applies uniformly to all blocks, the core idea becomes clear: we aim to dominate all vertices in a block that are adjacent to vertices in other blocks. This naturally aligns with an MSOL-expressible formulation. Therefore, the algorithm is not only applicable to trees but also extends to graphs of constant treewidth.

6 Linear-time Algorithm of MSS in Unit Interval Graphs

Let $I = (V(I), E(I))$ be a unit interval graph with $|V(I)| = n$. Each unit interval in $V(I)$ is on the x-axis, which is treated as a vertex in I. Two vertices are adjacent if their corresponding unit intervals intersect, forming an *edge* between them. The set of all such edges is denoted as $E(I)$. Each interval has a *left end* and a *right end*.

An interval v is called a *left adjacent* of an interval u if the x-coordinate of the left endpoint of v is less than that of u, and v intersects u. Similarly, an interval v is called a *right adjacent* of an interval u if the x-coordinate of the left endpoint of v is greater than that of u, and v intersects u. The *leftmost* interval is the one whose left endpoint has the smallest x-coordinate among all adjacent intervals, while the *rightmost* interval has the largest x-coordinate at its right endpoint.

Each interval is assigned a color from a set of c colors. The **Initialization** step is the same as discussed in Section 5. The other steps are also similar but with a little change, which is as follows:

Unit Interval Subgraph Formation.

- Since $B^{all} \subseteq B$, each B^{all} induces one or more connected induced unit interval subgraphs in I, and each such graph consists of vertices of the same color because $B^{all} \subseteq B$. We focus on these induced connected maximal unit interval subgraphs.
- Let $I_1, I_2, \ldots, I_t$ be induced connected maximal unit interval subgraphs in I formed by the vertices of B^{all}.

Selection Process.

- For each unit interval subgraph I_j (where $1 \leq j \leq t$), start from the leftmost interval u.
- u must belong to either B^1 or B^2.
- If $u \in B^2$:
  - Remove u from B^2, B^{all}, and I_j, as it must be adjacent to a interval in B^1.
- If $u \in B^1$:
  - If the rightmost adjacent interval of u exists (say v) in I_j, add v to M_B and remove w along with all of its left adjacent vertices from B^1, B^2, B^{all}, and I_j. Also move all the right adjacent intervals (if exists) of v from B^1 to B^2 and update the sets B^{all} accordingly.
  - If u has no right adjacent interval in I_j, add u to M_B and remove it from B^2, B^{all} and I_j.
- Repeat the **Selection Process** until I_j becomes empty. Once I_j has no vertices, increment $j \leftarrow j+1$ and repeat the **Selection Process** until $j = t$.

Assume $M = \bigcup_{B \in I} M_B$. Since the above algorithm is very similar to the algorithm for trees, Lemma 6 must also hold for unit interval graphs.

Lemma 9 * *M is a minimum selective subset of the interval graph I.*

The proof of this lemma is quite similar to the proof of the Lemma 7. The runtime is also the same as explained in Lemma 8.

7 Remarks

As the MSS problem is NP-complete in planar graphs, developing approximation algorithms and studying its parameterized complexity (specifically, designing FPT algorithms when the number of colors c is the parameter) are important open problems. Additionally, designing approximation algorithms and establishing hardness results for other graph classes, such as circular-arc graphs, present further promising directions for future research.

Acknowledgments

I would like to sincerely thank Bodhayan Roy for his constant guidance and support throughout this work. I am also grateful to Aritra Banik, Anil Maheshwari, Subhas C Nandy and Sasanka Roy for their valuable suggestions and feedback that helped improve the initial version of this paper. Their insights were instrumental during the early stages of this research. I also thank the anonymous reviewers for their thoughtful comments and suggestions, which helped enhance the quality and presentation of the paper.

References

[1] H. Arimura, T. Gima, Y. Kobayashi, H. Nochide, and Y. Otachi. Minimum consistent subset for trees revisited. *CoRR*, abs/2305.07259, 2023.

[2] S. Banerjee, S. Bhore, and R. Chitnis. Algorithms and hardness results for nearest neighbor problems in bicolored point sets. In M. A. Bender, M. Farach-Colton, and M. A. Mosteiro, editors, *LATIN 2018: Theoretical Informatics*, pages 80–93, 2018.

[3] A. Banik, S. Das, A. Maheshwari, B. Manna, S. C. Nandy, K. Priya K. M., B. Roy, S. Roy, and A. Sahu. Minimum Consistent Subset in Trees and Interval Graphs. In S. Barman and S. Lasota, editors, *44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2024)*, volume 323 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:15, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[4] A. Biniaz, S. Cabello, P. Carmi, J. D. Carufel, A. Maheshwari, S. Mehrabi, and M. Smid. On the minimum consistent subset problem. *Algorithmica*, 83(7):2273–2302, 2021.

[5] A. Biniaz and P. Khamsepour. The minimum consistent spanning subset problem on trees. *Journal of Graph Algorithms and Applications*, 28(1):81–93, 2024.

[6] R. Chitnis. Refined lower bounds for nearest neighbor condensation. In S. Dasgupta and N. Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29 March - 1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 262–281. PMLR, 2022.

[7] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[8] M. de Berg and A. Khosravi. Optimal binary space partitions in the plane. In M. T. Thai and S. Sahni, editors, *Computing and Combinatorics*, pages 216–225, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[9] S. Dey, A. Maheshwari, and S. C. Nandy. Minimum consistent subset of simple graph classes. In A. Mudgal and C. R. Subramanian, editors, *Algorithms and Discrete Applied Mathematics*, pages 471–484, Cham, 2021. Springer International Publishing.

[10] S. Dey, A. Maheshwari, and S. C. Nandy. Minimum consistent subset problem for trees. In E. Bampis and A. Pagourtzis, editors, *Fundamentals of Computation Theory - 23rd International Symposium, FCT 2021, Athens, Greece, September 12-15, 2021, Proceedings*, volume 12867 of *Lecture Notes in Computer Science*, pages 204–216. Springer, 2021.

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

[12] R. Diestel. Graph theory, volume 173 of. *Graduate texts in mathematics*, page 7, 2012.

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

[14] K. Khodamoradi, R. Krishnamurti, and B. Roy. Consistent subset problem with two labels. In B. Panda and P. P. Goswami, editors, *Algorithms and Discrete Applied Mathematics*, pages 131–142, 2018.

[15] B. Manna. Minimum Consistent Subset in Interval Graphs and Circle Graphs. *arXiv e-prints*, page arXiv:2405.14493, May 2024.

[16] B. Manna. Minimum Strict Consistent Subset in Paths, Spiders, Combs and Trees. *arXiv e-prints*, page arXiv:2405.18569, May 2024.

[17] B. Manna. Minimum Selective Subset on Some Graph Classes. *arXiv e-prints*, page arXiv:2507.00235, June 2025.

[18] G. Wilfong. Nearest neighbor problems. In *Proceedings of the Seventh Annual Symposium on Computational Geometry*, SCG '91, pages 224—-233, 1991.