Minimum Consistent Subset in Trees and Interval Graphs

Aritra Banik
National Institute of Science, Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, India

Sayani Das
Theoretical Computer Science, The Institute of Mathematical Sciences, Chennai, India

Anil Maheshwari
School of Computer Science, Carleton University, Ottawa, Canada

Bubai Manna
Department of Mathematics, Indian Institute of Technology Kharagpur, India

Subhas C. Nandy
Advanced Computing and Microelectronics Unit, Indian Statistical Institute, Kolkata, India

Krishna Priya K. M.
National Institute of Science, Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, India

Bodhayan Roy
Department of Mathematics, Indian Institute of Technology Kharagpur, India

Sasanka Roy
Advanced Computing and Microelectronics Unit, Indian Statistical Institute, Kolkata, India

Abhishek Sahu
National Institute of Science, Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, India

Abstract

In the Minimum Consistent Subset (MCS) problem, we are presented with a connected simple undirected graph G, consisting of a vertex set $V(G)$ of size n and an edge set $E(G)$. Each vertex in $V(G)$ is assigned a color from the set $\{1, 2, \ldots, c\}$. The objective is to determine a subset $V' \subseteq V(G)$ with minimum possible cardinality, such that for every vertex $v \in V(G)$, at least one of its nearest neighbors in V' (measured in terms of the hop distance) shares the same color as v. The decision problem, indicating whether there exists a subset V' of cardinality at most l for some positive integer l, is known to be NP-complete even for planar graphs.

In this paper, we establish that the MCS problem is NP-complete on trees. We also provide a fixed-parameter tractable (FPT) algorithm for MCS on trees parameterized by the number of colors (c) running in $O(2^{6c}n^6)$ time, significantly improving the currently best-known algorithm whose running time is $O(2^{4c}n^{2c+3})$. In an effort to comprehensively understand the computational complexity of the MCS problem across different graph classes, we extend our investigation to interval graphs. We show that it remains NP-complete for interval graphs, thus enriching graph classes where MCS remains intractable.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability; Theory of computation → Graph algorithms analysis; Theory of computation → Dynamic graph algorithms

Keywords and phrases Nearest-Neighbor Classification, Minimum Consistent Subset, Trees, Interval Graphs, Parameterized complexity, NP-complete

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2024.7

Related Version *Full Version*: https://arxiv.org/abs/2404.15487

Funding *Aritra Banik*: Supported by the Science and Engineering Research Board (SERB) via the project MTR/2022/000253.
Anil Maheshwari: Supported by Natural Sciences and Engineering Research Council of Canada (NSERC)
Bodhayan Roy: Supported by the Science and Engineering Research Board (SERB) via the project MTR/2021/000474.

44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2024).
Editors: Siddharth Barman and Sławomir Lasota; Article No. 7; pp. 7:1–7:15

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

For many supervised learning algorithms, the input comprises a colored training dataset T in a metric space $(\mathcal{X}, d)$ where each element $t \in T$ is assigned a color $C(t)$ from the set of colors $[c]$. The objective is to preprocess T in a manner that enables rapid assignment of color to any uncolored element in $\mathcal{X}$, satisfying specific optimality criteria. One commonly used optimality criterion is the nearest neighbor rule, where each uncolored element x is assigned a color based on the colors of its k nearest neighbors in the training dataset T (where k is a fixed integer). The efficiency of such an algorithm relies on the size of the training dataset. Therefore, it is crucial to reduce the size of the training set while preserving distance information. This concept was formalized by Hart [11] in 1968 as the minimum consistent subset (MCS) problem. In this problem, given a colored training dataset T, the objective is to find a subset $S \subseteq T$ of minimum cardinality such that for every point $t \in T$, the color of t is the same as the color of one of its nearest neighbors in S. Since its inception, the MCS problem has found numerous applications, as can be judged by over 2800 citations to [11] in Google Scholar.

The MCS problem for points in $\Re^2$ under the Euclidean norm is shown to be NP-complete if at least three colors color the input points. Furthermore, it remains NP-complete even for two colors [15, 12]. Recently, it has been shown that the MCS problem is W[1]-hard when parameterized by the output size [5]. For some algorithmic results on the MCS problem in $\Re^2$, see [3, 15].

In this paper, we explore the minimum consistent subset problem when $(\mathcal{X}, d)$ is a graph metric. Without loss of generality, we will use $[n]$ to denote the set of integers $\{1, \ldots, n\}$. For any graph G, we denote the set of vertices of G by $V(G)$ and the set of edges by $E(G)$. Consider any graph G and an arbitrary vertex coloring function $C : V(G) \to [c]$. For a subset of vertices U, let $C(U)$ represent the set of colors of the vertices in U, formally denoted as $C(U) = \{C(u) : u \in U\}$. For any two vertices $u, v \in V(G)$, the shortest path distance between u and v in G is denoted by $d(u, v)$. For a vertex $v \in V(G)$ and any subset of vertices $U \subseteq V(G)$, let $d(v, U) = \min_{u \in U} d(v, u)$. The nearest neighbors of v in the set U are denoted as $\mathrm{NN}(v, U)$, formally defined as $\mathrm{NN}(v, U) = \{u \in U : d(v, u) = d(v, U)\}$. A subset of vertices $S \subseteq V(G)$, is called a consistent subset for (G, C) if for every $v \in V(G)$, $C(v) \in C(\mathrm{NN}(v, S))$. The consistent subset problem on graphs is defined as follows:

Consistent Subset Problem on Graphs(CSPG)

Input: A graph G, a coloring function $C : V(G) \to [c]$, and an integer ℓ.
Question: Does there exist a consistent subset of size $\leq \ell$ for (G, C)?

A consistent subset of minimum size is called a minimum consistent subset (MCS). Banerjee et al. [2] proved that the CSPG is W[2]-hard [6] when parameterized by the size of the minimum consistent set, even when limited to two colors, thus refuting the possibility of an FPT algorithm parameterized by $(c + \ell)$ under standard complexity-theoretic assumptions for general graphs. This naturally raises the question of determining the simplest graph classes where the problem remains computationally intractable. Dey et al. [8] presented a polynomial-time algorithm for CSPG on simple graph classes such as paths, spiders, combs, and caterpillars. The CSPG has gained significant research attention in recent years when the underlying graph is a tree. Dey et al. [7] presented a polynomial-time algorithm for bi-colored trees, and Arimura et al. [1] presented an XP algorithm parameterized by the number of colors c, with a running time of $\mathcal{O}(2^{4c} n^{2c+3})$. Very recently, the paper [4] demonstrated a polynomial time algorithm for the *minimum consistent spanning subset* (a variant of MCS) in trees.

New Results. The most intriguing question yet to be answered is whether CSPG remains NP-hard for trees [1, 7]. In this paper, we decisively answer this question in the affirmative. This is particularly noteworthy given the scarcity of naturally occurring problems known to be NP-hard on trees. Our contribution includes a reduction from the MAX-2SAT problem, detailed in Section 3. Next, we show that CSPG is fixed-parameter tractable (FPT) for trees on n vertices, significantly improving the results presented in Arimura et al. [1]. Our intricate dynamic programming algorithm runs in $\mathcal{O}(2^{6c}n^6)$ time, whereas [1] requires $\mathcal{O}(2^{4c}n^{2c+3})$ time; see Section 4.

Moreover, in Section 5, we show that CSPG on interval graphs is NP-hard. While interval graphs are unrelated to trees, our hardness result for interval graphs raises new questions about the fixed-parameter tractability of CSPG on this distinct graph class.

2 Notation and Preliminary Results

Notations. For any graph G and any vertex $v \in V(G)$, let $N(v) = \{u : u \in V(G),\ (u, v) \in E(G)\}$ denotes the set of neighbours of v and $N[v] = \{v\} \cup N(v)$. We denote the distance between two subgraphs G_1 and G_2 in G by $d(G_1, G_2) = \min\{d(v_1, v_2) : v_1 \in V(G_1), v_2 \in V(G_2)\}$. For any subset of vertices $U \subseteq V(G)$ in a graph G, $G[U]$ denotes the subgraph of G induced on U. Most of the symbols and notations of graph theory used are standard and taken from [9].

As an elementary result, we prove that MCS is log-APX-hard [14]. We reduce the dominating set problem to the consistent subset problem (CSPG). In the *dominating set problem* given a graph G and an integer k the objective is to find out whether there exists a subset $U \subseteq V(G)$ of size k such that for any vertex $v \in V(G)$, $N[v] \cap U \neq \emptyset$. It is known that the set cover problem is log-APX-hard, or in other words, it is NP-hard to approximate the set cover problem within a $c \cdot \log n$ factor [13]. As there exists an L-reduction from set cover to dominating set problem, the dominating set problem is known to be log-APX-hard. Let (G, k) be any arbitrary instance of the dominating set problem with a graph G and an integer k. We construct an instance $(H, C, k + 1)$ of CSPG as follows. $V(H) = V(G) \cup \{x\}$ and $E(H) = E(G) \cup \{(x, v) : v \in V(G)\}$. For all $v \in V(G)$, we set $C(v) = 1$, and set $C(x) = 2$. For the sake of completeness, we state the following lemma.

▶ **Lemma 1.** *There exists a dominating set for G of size at most k if and only if there is a consistent subset of size at most $k + 1$ for the graph H.*

Proof. Let D be a dominating set of size k for the graph G. We claim that $D' = \{x\} \cup D$ is a consistent subset of H. If not, then there is a vertex $v_i \in V(H) \setminus D'$ such that $d(v_i, D) > d(v_i, x) = 1$. This contradicts the assumption that D is a dominating set for G and the claim holds.

On the other hand, suppose D' is a consistent subset of size $k + 1$ in the graph H. Observe that $x \in D'$ as x is the only vertex with color 2. We claim that $D = D' \setminus \{x\}$ is a dominating set of G. If not, then there is a vertex $v \in V(G) \setminus D$ such that $N(v) \cap D = \emptyset$. Thus $d(v, D') > 1$ but $d(v, x) = 1$ and $C(v) \neq C(x)$. This contradicts the assumption that C is a consistent subset and hence the claim holds. ◀

From Lemma 1, we have the following theorem.

▶ **Theorem 2.** *There exists a constant $c > 0$ such that it is NP-hard to approximate the Minimum Consistent Set problem within a factor of $c \cdot \log n$.*

3 NP-hardness of MCS for Trees

In this section, we prove that CSPG is NP-hard even when the input graph is a tree. We present a reduction from MAX-2SAT problem to CSPG. Let θ be a given MAX-2SAT formula with n variables $\{x_1, \ldots, x_n\}$ and m clauses $\{c_1, \ldots, c_m\}$, $n, m \geq 50$. We construct an instance (T_θ, C_θ) of the MCS problem from θ as follows.

Construction of (T_θ, C_θ).

The constructed tree T_θ is composed of variable gadgets, clause gadgets, and central vertex gadgets.

Variable Gadget.

A variable gadget X_i for the variable $x_i \in \theta$ has two components where each component has a literal path and M pairs of stabilizer vertices, as described below (see Figure 1), where M is very large (we will define the exact value of M later).

Literal paths: The two literal paths of the variable gadget X_i are $P_i^\ell = \langle x_i^1, x_i^2, x_i^3, x_i^4 \rangle$ and $\overline{P}_i^\ell = \langle \overline{x}_i^1, \overline{x}_i^2, \overline{x}_i^3, \overline{x}_i^4 \rangle$, each consisting of four vertices; these are referred to as *positive literal path*, and *negative literal path* respectively. Here, by a path of k (> 2) vertices, we mean a connected graph with $k-2$ nodes having degree 2 and the remaining two nodes having degree 1. All the vertices on the path P_i^ℓ are of color c_i^ℓ and all the vertices on the path $\overline{P}_i^\ell$ are of color $\overline{c}_i^\ell$.

Stabilizer vertices: M pairs of vertices $\{s_i^1, \overline{s}_i^1\}, \ldots, \{s_i^M, \overline{s}_i^M\}$, where the color of each pair of vertices $\{s_i^j, \overline{s}_i^j\}$ is $c^s(i,j)$. We denote the set of vertices $S_i = \{s_i^1, \ldots, s_i^M\}$ as *positive stabilizer vertices* and the set of vertices $\overline{S}_i = \{\overline{s}_i^1, \ldots, \overline{s}_i^M\}$ as *negative stabilizer vertices*. Each vertex in S_i is connected to x_i^1 and each vertex in $\overline{S}_i$ is connected to $\overline{x}_i^1$.

The intuition behind this set of stabilizer vertices is that by setting a large value of M we ensure that either $\{s_i^1, \ldots, s_i^M\}$ or $\{\overline{s}_i^1, \ldots, \overline{s}_i^M\}$ is present in any "small" sized solution.

Clause Gadget.

For each clause $c_i = (y_i \vee z_i)$, where y_i and z_i are two (positive/negative) literals, we define the clause gadget C_i as follows. It consists of three paths, namely *left occurrence path* $P_i^Y = \langle y_i^1, \cdots y_i^7 \rangle$, *right occurrence path* $P_i^Z = \langle z_i^1, \cdots z_i^7 \rangle$, and *clause path* $P_i^W = \langle w_i^1, \cdots w_i^7 \rangle$ (see Figure 1). All the vertices in P_i^Y (resp. P_i^Z) have the same color as that of the corresponding literal path in their respective variable gadgets, i.e. for any literal, say y_i in C_i, if $y_i = x_i$ (resp. $\overline{x_i}$) then we set the color of the vertices of P_i^Y as c_i^x (resp. $\overline{c}_i^x$). The color of the vertices on the path P_i^Z is set in the same manner. We color the vertices in P_i^W by c_i^w, which is different from that of the vertices in P_i^Y and P_i^Z. We create the clause gadget C_i by connecting y_i^1 with w_i^2 and z_i^1 with w_i^6 by an edge (see Figure 1).

Central Vertex Gadget.

We also have a central path $P^v = \langle v_1, v_2, v_3 \rangle$. The color of all the vertices in P^v is the same, say c^v, which is different from the color of all other vertices in the construction. For each variable gadget X_i, x_i^1 and $\overline{x}_i^1$ are connected with the vertex v_1 (see Figure 1). For each clause C_i, w_i^4 is connected with v_1. The color of the vertices of P^v is c^v.

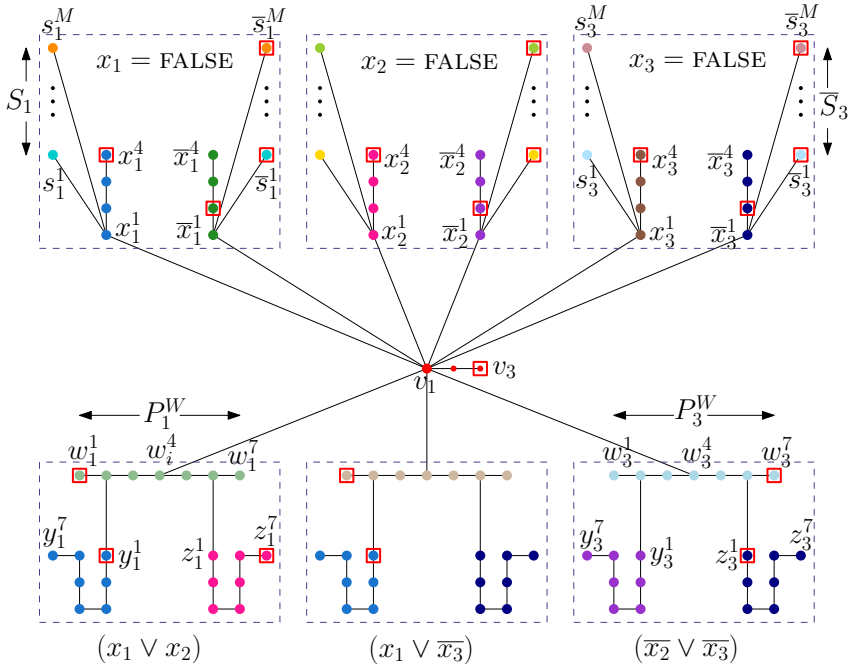

Figure 1 An example of construction of (T_θ, C_θ) where $\theta = (x_1 \vee x_2) \wedge (x_1 \vee \overline{x_3}) \wedge (\overline{x_2} \vee \overline{x_3})$. For the assignment $x_1 = x_2 = x_3 =$ FALSE, we have shown the corresponding CS with a red box around the vertices. For the assignment $x_1 = x_2 = x_3 =$ FALSE, $(x_1 \vee x_2)$ is not satisfied whereas the rest of the clauses are satisfied.

Our objective is to show that there exists an MCS of size at most $N(k) = n(M+2) + 2k + 3(m-k) + 1$ in the tree T_θ if at least k clauses of θ are satisfied; otherwise, the size is strictly greater than $N(k)$. We now prove a set of auxiliary claims about a minimum consistent subset for (T_θ, C_θ).

▶ **Lemma 3.** *For any consistent subset V_C of size at most $N(k) = n(M+2)+2k+3(m-k)+1$ in the tree T_θ, the following are true.*

- *For any variable x_i, exactly one of the following is true.*
  - *$S_i \subset V_C$, $\overline{S}_i \cap V_C = \emptyset$, and $x_i^2, \overline{x}_i^4 \in V_C$.*
  - *$\overline{S}_i \subset V_C$, $S_i \cap V_C = \emptyset$, and $\overline{x}_i^2, x_i^4 \in V_C$*

  , and
- *$v_3 \in V_C$.*

Lemma 3, states that by strategically choosing the vertices in a variable gadget, the vertices of the tree corresponding to that variable gadget can be consistently covered by choosing its only one set of stabilizer vertices.

Proof. For a variable x_i, let $S_i \cap V_C \neq \emptyset$. Let $s_i^j \in S_i \cap V_C$. But then every vertex $v \in S_i \setminus \{s_i^j\}$ must have a vertex within distance 2 of its own color, since $dist(s_i^j, v) = 2$. Hence $S_i \subset V_C$. One can similarly prove that $\overline{S}_i \cap V_C \neq \emptyset \implies \overline{S}_i \subseteq V_C$. Also, every variable gadget contains M uniquely colored vertices and hence has at least M vertices in V_C. So, if $M >> n$, we have that $N(k) < (n+1)M$, and there exists no variable gadget that contains vertices from both S_i and $\overline{S}_i$. In other words exactly one of the following holds for every variable gadget corresponding to a variable x_i:

- $S_i \subset V_C$, $\overline{S}_i \cap V_C = \emptyset$
- $\overline{S}_i \subset V_C$, $S_i \cap V_C = \emptyset$,

Below, we look into one of these cases, and a similar argument may be made for the other case as well.

Case 1: $S_i \subset V_C$, $\overline{S}_i \cap V_C = \emptyset$. Notice that there must be a vertex in the literal path $\{x_i^1, x_i^2, x_i^3, x_i^4\}$ from $\{x_i^1, x_i^2\}$ of the variable gadget X_i since $dist(S_i, x_i^1) = 1$ and the distance to any other vertex of the same color (other than these two vertices) is more than 1. But $x_i^1 \notin V_C$, as $\overline{S}_i \cap V_C = \emptyset$ and $dist(x_i^1, \overline{S}_i) < dist(S_i, \overline{S}_i)$. Hence $x_i^2 \in V_C$.

Similarly there must be a vertex in the literal path $\{\overline{x}_i^1, \overline{x}_i^2, \overline{x}_i^3, \overline{x}_i^4\}$ of the variable gadget X_i since $dist(S_i, \overline{x}_i^1) = 3$ and the distance to any other vertex of the same color (other than $\{\overline{x}_i^2, \overline{x}_i^3, \overline{x}_i^4\}$) is more than 3. And the distance of 4 between S_i and $\overline{S_i}$ eliminates the possibility of belonging any of $\overline{x}_i^1, \overline{x}_i^2$ or $\overline{x}_i^3$ belonging in V_C. Thus, $\overline{x}_i^4 \in V_C$.

Case 2: $\overline{S}_i \subset V_C$, $S_i \cap V_C = \emptyset$. Case 2 may be argued in a manner similar to Case 1.

Moreover, V_C must contain at least one veretx from the set $\{v_1, v_2, v_3\}$. However, the distance of 4 between S_i and $\overline{S_i}$ rules out the possibility of either v_1 or v_2 being in V_C. Consequently, $v_3 \in V_C$. ◀

To satisfy the inequality in the above lemma, we now set the value of M as n^3. In the next lemma, we present a bound on the vertices from each clause gadget that are contained in a consistent subset of size at most $N(k)$. For any clause C_i, denote the corresponding clause gadget by $T_i^C = G[\{w_i^a, y_i^a, z_i^a | 1 \leq a \leq 7\}]$.

▶ **Lemma 4.** *In any consistent subset V_C of the tree T_θ, for each clause C_i, $2 \leq |V(T_i^C) \cap V_C|$.*

Proof. There needs to be a vertex among the vertices $\{w_i^a \mid 1 \leq a \leq 7\}$ since they are distinctly colored from all other vertices. If this vertex belongs to $\{w_i^a \mid 1 \leq a \leq 4\}$, then there must also be a vertex in $\{y_i^a \mid 1 \leq a \leq 7\}$ since the nearest vertex of the same color (any y_i^a) is farther away than the vertex with the color of any w_i^a. Similarly, if this vertex belongs to $\{w_i^a \mid 4 \leq a \leq 7\}$, then there must be a vertex in $\{z_i^a \mid 1 \leq a \leq 7\}$. Therefore, $2 \leq |V(T_i^C) \cap V_C|$. ◀

▶ **Theorem 5.** *There exists a truth assignment of the variables in θ which satisfies at least k clauses if and only if there exists a consistent subset of size at most $N(k)$ for (T_θ, C_θ).*

Proof. ($\Rightarrow$) For the forward direction, let there exist an assignment A to the variables of θ that satisfies k clauses. Consider the following set of vertices V_A. For each variable x_i if x_i is TRUE, include all the vertices of S_i in V_A. Also include x_i^2 and $\overline{x}_i^4$ in V_A. If $x_i =$ FALSE then include all the vertices of $\overline{S}_i$ in V_A. Also include x_i^4 and $\overline{x}_i^2$ in V_A.

For every satisfied clause $C_i = (y_i \vee z_i)$ with respect to A we include the following vertices in V_A. Without loss of generality assume that $y_i =$ TRUE. We include w_i^7 and z_i^1 in V_A. For every unsatisfied clause $C_i = (y_i \vee z_i)$, we include w_i^1, y_i^1 and z_i^7 in V_A. We also include v_3 in V_A.

Observe that the cardinality of V_A is $N(k) = n(M+2) + 2k + 3(m-k) + 1$. Next, we prove that V_A is a consistent subset for T_θ. Observe that for any pair of vertices $(s_i^j, \overline{s}_i^j)$, exactly one of them is in V_A. Without loss of generality assume that $s_i^j \in V_A$. Observe that $d(s_i^j, \overline{s}_i^j) = d(\overline{s}_i^j, V_A) = 4$. If $x_i =$ TRUE then $d(x_i^j, x_i^2) \leq d(x_i^j, V_A)$ and $d(\overline{x}_i^j, \overline{x}_i^4) \leq d(\overline{x}_i^j, V_A)$. The case when $x_i =$ FALSE is symmetric.

For any clause gadget either w_i^1 or w_i^7 is in V_A. Without loss of generality assume that $w_i^1 \in V_A$. Observe that for every vertex w_i^j, $d(w_i^j, w_i^1) = d(w_i^j, V_A)$. Let $C_i = (y_i \vee z_i)$ be a satisfied clause and without loss of generality assume that $y_i = x_j =$ TRUE. Observe

that $d(y_i^1, x_j^2) = 6$, and $d(y_i^3, V_A) = 6$, and $d(y_i^a, x_j^2) = d(y_i^a, V_A)$. For any unsatisfied clause $C_i = (y_i \vee z_i)$ observe that $d(y_i^a, x_j^2) = d(y_i^a, V_A)$. Also for any v_j where $1 \leq j \leq 3$ $d(v_j, v_3) = d(v_j, V_A)$. Therefore V_A is a consistent subset for T_θ.

($\Leftarrow$) In the backward direction, let there be a consistent subset V_C of size at most $N(k)$ for (T_θ, C_θ). We know from Lemma 3 that either $S_i \subset V_C$ or $\overline{S}_i \subset V_C$. From Lemma 3 any such solution has at least $n(M+2)+1$ many vertices from V_C outside the clause gadgets leaving at most $2k + 3(m-k)$ that may be chosen from the clause gadgets.

Each clause gadget comprises of vertices of three distinct colors: one color exclusive to the clause itself and two colors dedicated to literals. An essential insight is that if there are no vertices in V_C of colors specific to the literals from a clause in the variable gadgets, then such a clause gadget must contain at least three vertices from V_C. This assertion is valid because the distance between two sets of vertices of the same color (corresponding to the same literal in two clauses) across any two clauses is at least 8, while vertices in V_C of clause-specific colors are at a distance of at most 6.

This fact, coupled with Lemma 4 implies that there are at least k clauses for whom colors specific to at least one of their literals have the vertices in V_C of the same color from the variable gadgets. Making the same literals true and setting other variables arbitrarily gives us an assignment that satisfies at least k clauses. ◀

4 MCS for Trees: A Parameterized Algorithm

In this section, we consider the optimization version of the MCS problem for the trees.

Minimum Consistent Subset for Trees	**Parameter:** c
Input: A rooted tree T, whose vertices $V(T)$ are coloured with a set C of c colours.	
Question: Find the minimum possible size of a consistent subset (MCS) for T?	

We consider T as a rooted tree by taking an arbitrary vertex r as its root. We use $V(T')$ to denote the vertices of a subtree T' of T, and $C(U) \subseteq C$ to denote the subset of colors assigned to subset of vertices $U \subseteq V$, and $C(u)$ to denote the color attached to the vertex $u \in V(T)$. For any vertex v, let η_v denote the number of children of v and we denote the children of v by $v_1, v_2, \cdots, v_{\eta_v}$. We denote the subtree rooted at a vertex v by $T(v)$. For any vertex v and any integer $i < \eta_v$, we use $T_{i+}(v)$ to denote the union of subtrees rooted at v_{i+1} to v_{η_v}, and $T_i(v)$ to denote the subtree rooted at v and containing first i many children of v. Thus, $T_{i+}(v) = \cup_{i+1 \leq j \leq \eta_v} T(v_j)$, which is a forest, and $T_i(v) = T(v) \setminus T_{i+}(v)$. In Figure 2(a), the light yellow part is $T_i(v)$, and the light sky-colored part is $T_{i+}(v)$. We define $T^{out}(v) = T \setminus T(v)$.

For any positive integer d and for any vertex $v \in V(T)$, a set of vertices $U \subset V(T)$ is called d-equidistant from v if $d(u_i, v) = d$ for all $u_i \in U$. Any subset of vertices U spans a set of colors $C' \subseteq C$ if $C(U) = C'$. For any vertex $v \in V(T)$, we use $\mathcal{E}_i^{\text{SIB}}(v, d, C')$ (resp. $\mathcal{E}_i^{\text{OUT}}(v, d, C')$) to denote the set of subsets of vertices in $T_{i+}(v)$ (resp. $T \setminus T(v)$), which are d-equidistant from v and span the colors in C'. Next, we define a (*partial*) *consistent subset* for a subtree $T_i(v)$.

Intuition. Our dynamic programming (DP) routine exploits the key observation that a (partial) consistent subset (formally defined below) for a subtree $T(v)$ can be computed in FPT time. This computation is possible given the distance to the closest vertex in the consistent subset that lies outside $T(v)$ and the colors of those vertices. The entries in our

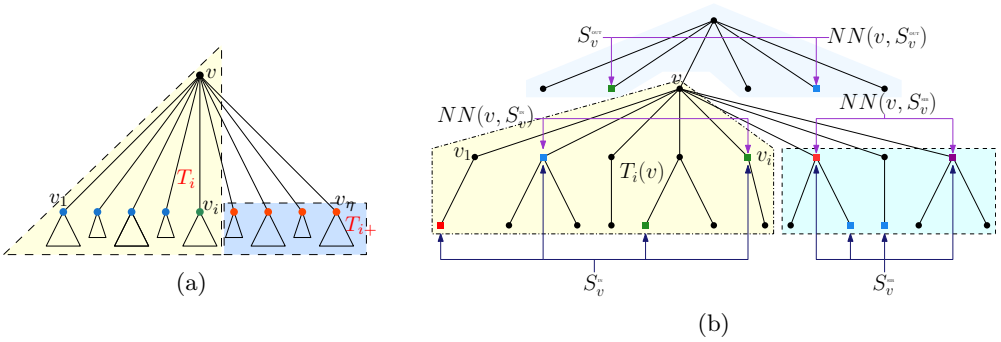

Figure 2 Illustration of the bottom-up dynamic programming routine, ■ vertices denotes consistent subset. $\delta_S^{\text{IN}} = 1, \delta_S^{\text{OUT}} = 2, \delta_S^{\text{SIB}} = 1$.

DP table store the minimum size of partial consistent subsets for all subtrees $T_i(v)$. These subsets are defined based on six parameters: distance to the closest vertex from v in the consistent subset in $T_i(v)$, $T^{out}(v)$ and $T_{i+}(v)$ and colors of these three set of closest vertices.

▶ **Definition 6.** *Let $d^{\text{IN}} \in \mathbb{Z}_0^+$ and $d^{\text{OUT}}, d^{\text{SIB}} \in \mathbb{Z}^+$, and let three subsets of colors $C^{\text{IN}}, C^{\text{OUT}}, C^{\text{SIB}} \subseteq C$. A (partial) consistent subset of the subtree $T_i(v)$ with respect to the parameters d^{IN}, d^{OUT}, d^{SIB}, C^{IN}, C^{OUT}, C^{SIB} is defined as a set of vertices $W \subseteq V(T_i(v))$ such that for any arbitrary subset $X \in \mathcal{E}_i^{SIB}(v, d^{\text{SIB}}, C^{\text{SIB}})$ and $Y \in \mathcal{E}_i^{OUT}(v, d^{\text{OUT}}, C^{\text{OUT}})$ (assuming they exist), W satisfies the following (see Figure 2(b)):*

- $d(v, W) = d^{\text{IN}}$. *(i.e., the distance of v to its nearest member(s) in W is d^{IN})*
- $C(\text{NN}(v, W)) = C^{\text{IN}}$. *(i.e., C^{IN} is the set of colors of the nearest members of v in the set W)*
- *For every vertex $u \in T_i(v)$, $C(u) \in C(\text{NN}(u, W \cup X \cup Y))$.*

Note that for some values of $d^{\text{IN}}, d^{\text{OUT}}, d^{\text{SIB}}, C^{\text{IN}}, C^{\text{OUT}}, C^{\text{SIB}}$ there may not exist any (partial) consistent subset for $T_i(v)$; in such a case we define it is undefined. Also note that, for some values, the (partial) consistent subset can be empty as well, such as when $d^{\text{IN}} = \infty$ and $C(u) \in C(\text{NN}(u, X \cup Y))$ for every vertex $u \in T_i(v)$. For ease of notation, we will denote a (partial) consistent subset for $T_i(v)$ as a consistent subset with respect to the parameters $d^{\text{IN}}, d^{\text{OUT}}, d^{\text{SIB}}, C^{\text{IN}}, C^{\text{OUT}}, C^{\text{SIB}}$.

Consider an arbitrary consistent subset S_T of T, an arbitrary vertex $v \in V(T)$ and an integer $i \in [\eta_v]$ (see Figure 2(b)). For any vertex $v \in V(T)$ and $1 \leq i \leq \eta_v$, define $S_v^{\text{IN}} = S_T \cap V(T_i(v))$, $S_v^{\text{SIB}} = S_T \cap V(T_{i+}(v))$, and $S_v^{\text{OUT}} = S_T \cap V(T \setminus T(v))$. Also define $\delta_S^{\text{IN}} = d(v, S_v^{\text{IN}})$, $C_S^{\text{IN}} = C(\text{NN}(v, S_v^{\text{IN}}))$, $\delta_S^{\text{SIB}} = d(v, S_v^{\text{SIB}})$, $C_S^{\text{SIB}} = C(\text{NN}(v, S_v^{\text{SIB}}))$, $\delta_S^{\text{OUT}} = d(v, S_v^{\text{OUT}})$, $C_S^{\text{OUT}} = C(\text{NN}(v, S_v^{\text{OUT}}))$. Let W be any arbitrary (partial) consistent subset with respect to the parameters $\delta_S^{\text{IN}}, \delta_S^{\text{OUT}}, \delta_S^{\text{SIB}}, C_S^{\text{IN}}, C_S^{\text{OUT}}, C_S^{\text{SIB}}$ (see Definition 6). Next, we have the following lemma.

▶ **Lemma 7.** *$S_W = (S_T \setminus S_v^{\text{IN}}) \cup W$ is a consistent subset for T.*

Proof. Suppose that $A = W \cup \text{NN}(v, S_v^{\text{SIB}}) \cup \text{NN}(v, S_v^{\text{OUT}})$ is the set of vertices which are either in W or in the nearest neighbor of v outside $T_i(v)$ in S_W. We will show that for any vertex $u \in T_i(v)$, $NN(u, S_W) \subseteq A$ and there is a vertex in $NN(u, A)$ of the color same as u. Similarly, let $B = (S_W \setminus W) \cup \text{NN}(v, W)$. We show that for any vertex w outside $T_i(v)$, $NN(w, S_W) \subseteq B$ and there is a vertex in $NN(w, B)$ of the color same as w. Please note that A and B are not necessarily disjoint.

Consider a vertex $u \in T_i(v)$ and $w \in T \setminus T_i(v)$. Since $\text{NN}(v, S_v^{\text{SIB}}) \in \mathcal{E}_i^{\text{SIB}}(v, \delta_S^{\text{SIB}}, C_S^{\text{SIB}})$, $\text{NN}(v, S_v^{\text{OUT}}) \in \mathcal{E}_i^{\text{OUT}}(v, \delta_S^{\text{OUT}}, C_S^{\text{OUT}})$, and W is a consistent subset with respect to the parameters $\delta_S^{\text{IN}}, \delta_S^{\text{OUT}}, \delta_S^{\text{SIB}}, C_S^{\text{IN}}, C_S^{\text{OUT}}, C_S^{\text{SIB}}$, we have $C(u) \in C(\text{NN}(u, W \cup \text{NN}(v, S_v^{\text{SIB}}) \cup \text{NN}(v, S_v^{\text{OUT}}))) = C(\text{NN}(u, A))$. Also, as S_T is a consistent subset, we have $C(w) \in C(\text{NN}(w, S_T \setminus S_v^{\text{IN}}) \cup C_S^{\text{IN}})$. From the properties of W, we have $C(\text{NN}(v, W)) = C_S^{\text{IN}}$. Hence, $C(w) \in C(\text{NN}(w, B))$. Thus, to prove the result, it is enough to show that (i) no vertex from $B \setminus A$ can be the closest to the vertex u in the set S_W, and (ii) no vertex from $A \setminus B$ can be the closest vertex of w in the set S_W. We prove these two claims by contradiction.

Assume that *Claim (i)* is false. Then, there will be a vertex $x \in B \setminus A$, which is closest to u. Note that, $(B \setminus A) \cap ((T_i(v) \cup \text{NN}(v, S_v^{\text{SIB}}) \cup \text{NN}(v, S_v^{\text{OUT}})) = \emptyset$. Hence we have $d(u, x) = d(u, v) + d(v, x)$, $d(v, x) > min(\delta_S^{\text{SIB}}, \delta_S^{\text{OUT}})$. This is a contradiction as the closest vertex from v in $S_W \cup (T \setminus T_i(v))$ (if it exists) is at distance $min(\delta_S^{\text{SIB}}, \delta_S^{\text{OUT}}) = d(v, \text{NN}(v, S_v^{\text{SIB}}) \cup \text{NN}(v, S_v^{\text{OUT}}))$. Hence, x cannot be the closest vertex of u in S_W. Thus, *Claim (i)* follows.

Now, assume that *Claim (ii)* is false. Then there is a vertex $y \in A \setminus B$, which is closest to w. As $w \notin T_i(v)$ and $y \in T_i(v)$, we have $d(w, y) = d(w, v) + d(v, y)$. Since $d(v, \text{NN}(v, W)) = \delta_S^{\text{IN}}$ and $w \in (A \setminus B = W \setminus \text{NN}(v, W))$, we have $d(v, y) > \delta_S^{\text{IN}}$. This contradicts the fact that the closest vertex from v in $S_W \cup T_i(v)$ (if it exists) is at distance δ^{IN} from v. Hence, *Claim (ii)* is true. ◀

Motivated by Lemma 7, we design the following algorithm based on the dynamic programming technique. For each choice of $v \in V(T)$, $i \in [\eta_v]$, $\delta_v^{\text{IN}} \in [n] \cup \{0, \infty\}$, $\delta_v^{\text{OUT}} \in [n] \cup \{\infty\}$, $\delta_v^{\text{SIB}} \in [n] \cup \{\infty\}$, and $C_v, C_v^{\text{OUT}}, C_v^{\text{SIB}} \subseteq C$, we define a subproblem which computes the cardinality of a minimum sized (partial) consistent subset for the subtree $T_i(v)$ with respect to the parameters $\langle \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}, C_v^{\text{SIB}} \rangle$, and denote its size by $P(T_i(v), \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}, C_v^{\text{SIB}})$. Let us use $\delta_v^{\text{MIN}} = \min(\delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}})$.

$$A = \begin{cases} C_v^{\text{IN}} \text{ if } \delta_v^{\text{IN}} = \delta_v^{\text{MIN}} \\ \emptyset \quad \text{otherwise} \end{cases}, \qquad B = \begin{cases} C_v^{\text{SIB}} \text{ if } \delta_v^{\text{SIB}} = \delta_v^{\text{MIN}} \\ \emptyset \quad \text{otherwise} \end{cases}, \qquad D = \begin{cases} C_v^{\text{OUT}} \text{ if } \delta_v^{\text{OUT}} = \delta_v^{\text{MIN}} \\ \emptyset \quad \text{otherwise} \end{cases}$$

We define $C_v^{\text{MIN}} = A \cup B \cup D$. Note that δ_v^{MIN} is the distance of the closest vertex to v in any $X \in \mathcal{E}_i^{\text{SIB}}(v, \delta_v^{\text{SIB}}, C_v^{\text{SIB}})$, any $Y \in \mathcal{E}_i^{\text{OUT}}(v, \delta_v^{\text{OUT}}, C_v^{\text{OUT}})$ or the consistent subset, and that C_v^{MIN} denotes the colors of all such vertices.

To compute any DP entry, we take into account the following six cases. The first two cases are for checking whether a DP entry is valid. The third case considers the scenario in which v is part of the solution; the fourth, fifth, and sixth cases collectively consider the scenario in which v is not in the solution.

Case 1: If $C(v) \notin C_v^{\text{MIN}}$, return undefined.
Case 2: $\delta_v^{\text{IN}} = 0$ and $C_v^{\text{IN}} \neq \{C(v)\}$. Here, return ∞.
Case 3: $\delta_v^{\text{IN}} = 0$ and $C_v^{\text{IN}} = \{C(v)\}$. Here, return $P(T_i(v), \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}, C_v^{\text{SIB}}) =$

$$1 + \sum_{1 \leq j \leq i} \left\{ \min_{\delta, C'} P\Big(T_{\eta_{v_j}}(v_j), \delta, 1, \infty, C', \{C(v)\}, \emptyset\Big) \right\}$$

Explanation. Case 1 and Case 2 are self-explanatory. Case 3 implies that the vertex v is included in the consistent subset. Consequently, for the optimal solution, we need to determine a consistent subset for each tree rooted at a child v_j of v, independently of each other, assuming that v is part of the consistent subset (refer to Figure 2(a)). For every child v_j of v, we iterate through all possible choices of $C' \subseteq C$ and $\delta_{v_j}^{\text{IN}} = \delta \in \{1, \ldots, h(T(v_j)\} \cup \{\infty\}$

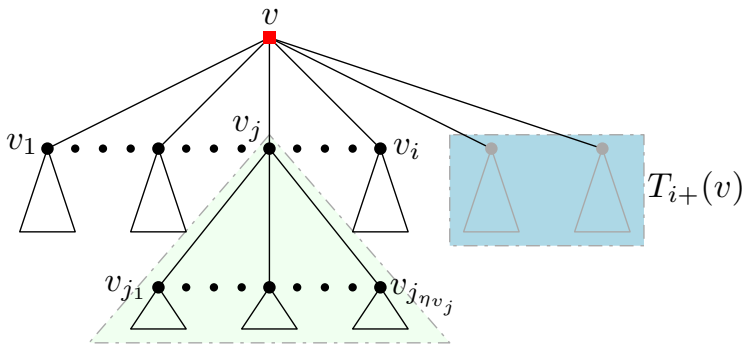

Figure 3 Illustration of the Case 3, where $\delta_v^{\text{IN}} = 0$.

where $h(T(v_j))$ is the height of the tree rooted at v_j, to identify the minimum consistent subset for $T_{\eta_{v_j}}(v_j)$. This is done with the constraints that the closest vertex in the consistent subset inside $T_{\eta_{v_j}}(v_j)$ is at a distance of δ and spans C'. For any vertex in $T(v_j)$, the path of the closest vertex of its own color outside $T_i(v)$ has to pass through v, which is considered to be in the consistent subset and has color $C(v)$. Thus, $\delta_v^{\text{OUT}} = 1$ is taken for the tree $T_{\eta_{v_j}}(v_j)$. Since we are solving for the complete tree rooted at v_j with no siblings, we set the distance to the closest sibling vertex as $\delta_v^{\text{SIB}} = \infty$ and the corresponding color set as $\emptyset$.

Notations for Subsequent Cases. In the rest of the section, we consider three more cases where $\delta_v^{\text{IN}} > 0$, and hence $\delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}} > 0$. Intuitively, while solving the problem recursively, we will recursively solve MCS in $T_{i-1}(v)$ and $T_{\eta_{v_i}}(v_i)$. We try all possible sets of choices of C_a, C_b with $C_v^{\text{IN}} = C_a \cup C_b$, and recursively solve for a solution assuming that the nodes of colors in C_a are present in $T_{i-1}(v)$ at a distance of δ_v^{IN} (if $C_a \neq \emptyset$ and such choices are feasible) and nodes of colors in $C_b \subseteq C_v^{\text{IN}}$ are present in $T_{\eta_{v_i}}(v_i)$ at a distance of $\delta_v^{\text{IN}} - 1$ from v_i (if $C_b \neq \emptyset$ and such choices are feasible).

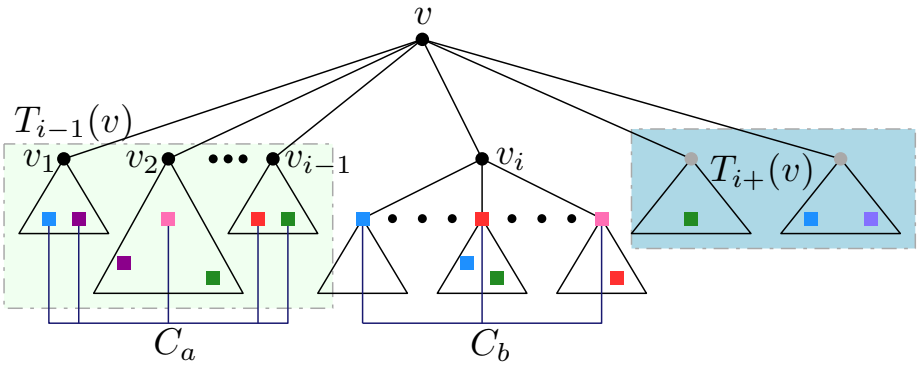

Figure 4 Illustration of the Case 4.

Case 4: $C_a, C_b \neq \emptyset$. In this case, the closest vertices in the consistent subset from v in both $T_{i-1}(v)$ and $T_{\eta(v_i)}(v_i)$ are located at a distance of precisely δ_v^{IN}. We start by defining the following. Let $\delta_x = \min(\delta_v^{\text{IN}}, \delta_v^{\text{SIB}})$ and recall $\delta_v^{\text{MIN}} = \min(\delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}})$.
Observe that both $T_{i+}(v)$ and $T_{\eta_{v_i}}(v_i)$ contains siblings of $T_{i-1}(v)$. Thus, in a hypothetical consistent subset, which is compatible with the current partial consistent subset, for the tree T, the closest vertices from v in $T_{(i-1)+}(v)$ are either in $T_{i+}(v)$ or $T_{\eta_{v_i}}(v_i)$. Here, δ_x is

trying to capture this distance information, and C^{SIB}_{i-1} represents the colors of such vertices. Similarly, from v_i in a hypothetical consistent subset CS for T, which is compatible with the current partial consistent subset, $\delta^{\text{MIN}}_v + 1$ denotes the distance to the vertices in CS contained in $T \setminus T(v_i)$. Note that these vertices can be either in $T_{(i-1)}(v)$ or in $T_{i+}(v)$ or in $T \setminus T(v)$. C^{OUT}_i represents the colors of such vertices.

$$C^{\text{SIB}}_{i-1} = \begin{cases} C_b & \text{if } \delta^{\text{IN}}_v < \delta^{\text{SIB}}_v \text{ and } C_b \neq \emptyset \\ C_b \cup C^{\text{SIB}}_v & \text{if } \delta^{\text{IN}}_v = \delta^{\text{SIB}}_v \\ C^{\text{SIB}}_v & \text{otherwise} \end{cases}$$

Also, define $C^{\text{OUT}}_i = A \cup B \cup D$, where

$$A = \begin{cases} \mathcal{C}_a \text{ if } \delta^{\text{IN}}_v = \delta^{\text{MIN}}_v \\ \emptyset \quad \text{otherwise} \end{cases}, \; B = \begin{cases} C^{\text{SIB}}_v \text{ if } \delta^{\text{SIB}}_v = \delta^{\text{MIN}}_v \\ \emptyset \quad \text{otherwise} \end{cases}, \quad D = \begin{cases} C^{\text{OUT}}_v \text{ if } \delta^{\text{OUT}}_v = \delta^{\text{MIN}}_v \\ \emptyset \quad \text{otherwise} \end{cases}$$

Now we can safely assume that there is a δ_x-equidistant set from v contained in $T_{(i-1)+}(v)$ that spans C^{SIB}_{i-1}.

$$\text{Return } P(T_i(v), \delta^{\text{IN}}_v, \delta^{\text{OUT}}_v, \delta^{\text{SIB}}_v, C^{\text{IN}}_v, C^{\text{OUT}}_v, C^{\text{SIB}}_v) = \min_{C_a, C_b} \Big(P\Big(T_{i-1}(v), \delta^{\text{IN}}_v, \delta^{\text{OUT}}_v, \delta_x, C_a, C^{\text{OUT}}_v, C^{\text{SIB}}_{i-1}\Big) + P\Big(T_{\eta_{v_i}}(v_i), \delta^{\text{IN}}_v - 1, \delta^{\text{MIN}}_v + 1, \infty, C_b, C^{\text{OUT}}_i, \emptyset\Big)\Big)$$

Explanation. In this case we iterate over all possible choices of C_a and C_b, assuming $C_a, C_b \neq \emptyset$ and $C_a \cup C_b = C^{\text{IN}}_v$. In the first part of the recursive formula, we recursively solve the problem for the tree $T_{i-1}(v)$ with the restriction that we have to include a set of vertices of color C_a in $T_{i-1}(v)$ at distance δ^{IN}_v from v (see Figure 2(b)). The restriction on C^{OUT}_v and δ^{OUT}_v among the vertices in $T \setminus T(v)$ remains the same as that of the parent problem. Regarding C^{SIB}_v and δ^{SIB}_v, observe that vertices in $T(v_i)$ and T_{i+} are part of $T_{(i-1)+}$. Therefore the parameters for the sibling depend on the value of δ^{IN}_v and δ^{SIB}_v, and accordingly, we have defined C^{SIB}_{i-1}.

In the second part of the recursive formula, we are solving the problem recursively for the tree $T_{\eta_{v_i}}(v_i)$, with the restriction that, in the consistent set in the consistent set we have to include a set of vertices from $T_{\eta_{v_i}}(v_i)$ which are of colors C_b, and at distance $\delta^{\text{IN}}_v - 1$ from v_i. Observe that the vertices in $T_{i-1}(v)$, $T_{i+}(v)$ and $T \setminus T(v)$ are all outside $T(v_i)$. Thus the restriction on the distance to the vertices on the consistent subset outside $T(v_i)$ and their colors depend on the values of $\delta^{\text{IN}}_v, \delta^{\text{SIB}}_v$ and δ^{OUT}_v. Thus the distance δ^{OUT}_v of this subproblem is defined as $\delta^{\text{MIN}}_v = \min(\delta^{\text{IN}}_v, \delta^{\text{SIB}}_v, \delta^{\text{OUT}}_v)$, and the set of colors C^{OUT}_i is defined accordingly. As we are solving for the whole tree rooted at v_i, there are no siblings; so $\delta^{\text{SIB}}_v = 0$ and $C^{\text{SIB}}_i = \emptyset$.

Case 5: $C_a = \emptyset$ and $C_b = C^{\text{IN}}_v$. Note that in this case, the closest vertices in the consistent subset from v in $T_{\eta(v_i)}(v_i)$ are located at a distance of δ^{IN}_v while in $T_{i-1}(v)$, they are located at a distance of at least $\delta \geq \delta^{\text{IN}}_v + 1$

We iterate over all values $\delta > \delta^{\text{IN}}_v$ and all possible choices of colors to find the size of a minimum consistent subset. We define δ_x and C^{SIB}_{i-1} the same as in Case 4. For any values $\delta > \delta^{\text{IN}}_v$ and $C \subseteq [c]$, we define $\delta^{\text{MIN}}_v(\delta, C) = \min(\delta, \delta^{\text{SIB}}_v, \delta^{\text{OUT}}_v)$, and $C^{\text{OUT}}_i(\delta, C) = A \cup B \cup D$ where

$$A = \begin{cases} \mathcal{C} \text{ if } \delta = \delta^{\text{MIN}}_v \\ \emptyset \text{ otherwise} \end{cases}, \quad B = \begin{cases} C^{\text{SIB}}_v \text{ if } \delta^{\text{SIB}}_v = \delta^{\text{MIN}}_v \\ \emptyset \quad \text{otherwise} \end{cases}, \quad D = \begin{cases} C^{\text{OUT}}_v \text{ if } \delta^{\text{OUT}}_v = \delta^{\text{MIN}}_v \\ \emptyset \quad \text{otherwise} \end{cases}$$

From v_i in a hypothetical consistent subset CS for T, which is compatible with the current partial consistent subset, $\delta_v^{\text{MIN}}(\delta, C)$ denotes the distance to the vertices in CS contained in $T \setminus T(v_i)$. Note that these vertices can be either in $T_{(i-1)}(v)$ or in $T_{i+}(v)$ or in $T \setminus T(v)$. $C_i^{\text{OUT}}(\delta, C)$ represents the colors of such vertices.

$$\text{Return } P(T_i(v), \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}, C_v^{\text{SIB}}) = \min_{\delta > \delta_v^{\text{IN}}, C \subseteq [c]} \Big(P\Big(T_{i-1}(v), \delta, \delta_v^{\text{OUT}}, \delta_x, C, C_v^{\text{OUT}}, C_{i-1}^{\text{SIB}}\Big) + P\Big(T_{\eta_{v_i}}(v_i), \delta_v^{\text{IN}} - 1, \delta_v^{\text{MIN}}(\delta, C) + 1, \infty, C_b, C_i^{\text{OUT}}(\delta, C), \emptyset\Big)\Big)$$

Explanation. The explanation for this case is the same as Case 4 except for the fact that we have to make sure that the closest vertex chosen in the consistent subset from $T_{i-1}(v)$ is at distance at least $\delta_v^{\text{IN}} + 1$.

Case 6: $C_b = \emptyset$ and $C_a = C_v^{\text{IN}}$.

Here we consider the case when $C_b = \emptyset$ and $C_a = C_v^{\text{IN}}$. Note that in this case, the closest vertices in the consistent subset from v in $T_{i-1}(v)$ are located at a distance of δ_v^{IN} while in $T_{\eta(v_i)}(v_i)$, they are located at a distance of at least $\delta \geq \delta_v^{\text{IN}} + 1$

We define $\delta_v^{\text{MIN}}(\delta, C) = \min(\delta_v^{\text{IN}}, \delta_v^{\text{SIB}}, \delta_v^{\text{OUT}})$. We define $C_i^{\text{OUT}}(\delta, C) = A \cup B \cup D$ where

$$A = \begin{cases} \mathcal{C}_a \text{ if } \delta_v^{\text{IN}} = \delta_v^{\text{MIN}} \\ \emptyset \text{ otherwise} \end{cases}, \quad B = \begin{cases} C_v^{\text{SIB}} \text{ if } \delta_v^{\text{SIB}} = \delta_v^{\text{MIN}} \\ \emptyset \text{ otherwise} \end{cases}, \quad D = \begin{cases} C_v^{\text{OUT}} \text{ if } \delta_v^{\text{OUT}} = \delta_v^{\text{MIN}} \\ \emptyset \text{ otherwise} \end{cases}$$

For any values $\delta > \delta_v^{\text{IN}}$ and $C \subseteq [c]$ we define $\delta_x(\delta, C) = \min(\delta_v^{\text{SIB}}, \delta)$ We define $C_{i-1}^{\text{SIB}}(\delta, C) = E \cup F$ where

$$E = \begin{cases} \mathcal{C} \text{ if } \delta = \delta_x(\delta, C) \\ \emptyset \text{ otherwise} \end{cases}, \quad F = \begin{cases} C_v^{\text{SIB}} \text{ if } \delta_v^{\text{SIB}} = \delta_x(\delta, C) \\ \emptyset \text{ otherwise} \end{cases}$$

The meaning and the reasoning behind the defining of $\delta_v^{\text{MIN}}(\delta, C)$ and $C_i^{\text{OUT}}(\delta, C)$ remains the same as in previous cases. Thus, in a hypothetical consistent subset, which is compatible with the current partial consistent subset, for the tree T, the closest vertices from v in $T_{(i-1)+}(v)$ are either in $T_{i+}(v)$ or $T_{\eta_{v_i}}(v_i)$. Here, $\delta_x(\delta, C)$ is trying to capture this distance information, and $C_{i-1}^{\text{SIB}}(\delta, C)$ represents the colors of such vertices.

In this case we return:

$$\text{Return } P(T_i(v), \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}, C_v^{\text{SIB}}) = \min_{\delta > \delta_v^{\text{IN}},\ C \subseteq [c]} \Big(P\Big(T_{i-1}(v), \delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_x(\delta, C), C_a, C_v^{\text{OUT}}, C_{i-1}^{\text{SIB}}(\delta, C)\Big) + P\Big(T_{\eta_{v_i}}(v_i), \delta, \delta_v^{\text{MIN}} + 1, \infty, C_b, C, \emptyset\Big)\Big)$$

Explanation. The explanation for this case is the same as the previous two cases.

Running time of the algorithm. The total number of choices of $\delta_v^{\text{IN}}, \delta_v^{\text{OUT}}, \delta_v^{\text{SIB}}, C_v^{\text{IN}}, C_v^{\text{OUT}}$ and C_v^{SIB} is bounded by $n^3 2^{3c}$. For each choice C_a and C_b, the algorithm takes at-most $n2^c$ time to go through all possible entries of δ and C (in **case 5** and **case 6**) and there are at-most 2^{2c} choices of C_a and C_b. The recursion runs for at most n^2 times. Hence, the worst-case running time of the algorithm is $\mathcal{O}(2^{6c} n^6)$.

5 NP-hardness of MCS for Interval Graphs

A graph H is said to be an interval graph if there exists an interval layout of the graph H, or in other words, for each node, $v_i \in V(H)$ one can assign an interval α_i on the real line such that $(v_i, v_j) \in E(H)$ if and only if α_i and α_j (completely or partially) overlap in the layout of those intervals.

We prove that the Minimum Consistent Subset problem is NP-complete even when the input graph is an interval graph. We present a reduction from the Vertex Cover problem for cubic graphs. It is known that Vertex Cover remains NP-complete even for cubic graphs [10]. For any set of intervals $\mathcal{I}$, let $G(\mathcal{I})$ be the interval graph corresponding to the set of intervals $\mathcal{I}$.

Interval Graph Construction.
Let G be any cubic graph, where $V(G) = \{v_1, \ldots, v_n\}$ is the set of vertices, and $E(G) = \{e_1, \ldots, e_m\}$ is the set of edges in G. We create the set of intervals $\mathcal{I}_G$ for G on a real line $\mathcal{L}$. The set of intervals in $\mathcal{I}_G$ is represented by intervals of three different sizes, *medium*, *small* and *large*, where each medium interval is of unit length, each small interval is of length $\epsilon << \frac{1}{2n^3}$ and the length of the large interval is $\ell >> 2n$.
We define $\mathcal{I}_G = \mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3 \cup \mathcal{I}_4$ where $\mathcal{I}_1$ contains $2m$ *medium* size intervals (two intervals for each edge) and defined as $\mathcal{I}_1 = \{I(e_i, v_j) : e_i = (v_j, y) \in E(G), y \in V(G)\}$. We set color c_i to the interval $I(e_i, v_j)$. $\mathcal{I}_2$ contain $n \cdot n^3$ *small* intervals of color c_{m+1}, and $\mathcal{I}_3$ contain $n \cdot n^4$ *small* intervals of color c_{m+1}. $\mathcal{I}_4$ contains one large interval I_ℓ of color c_1.
We create the following vertex gadget X_i for each vertex $v_i \in V(G)$. X_i contains the following *medium* size intervals $\{I(e, v_i) : e = (v_i, x) \in E(G)\}$ corresponding to the edges that are incident on v_i. These intervals span the same region s_i of unit length on the real line $\mathcal{L}$; hence they are mutually completely overlapping. In the vertex gadget X_i, we also include a total of n^3 mutually non-overlapping small intervals in the set $\mathcal{I}_2$. Span of all the small intervals in X_i is contained in the span s_i of the *medium* sized intervals in X_i (see Figure 5).
Each vertex gadget is placed one after another (in a non-overlapping manner) along the line $\mathcal{L}$ in an arbitrary order, such that a total of n^4 mutually non-overlapping small intervals can be drawn between two consecutive vertex gadgets. Thus, $\mathcal{I}_3$ contains n sets of n^4 non-overlapping small intervals. Finally, $\mathcal{I}_4$ contains a single large interval I_ℓ that contains all the intervals in $\mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3$. This completes the construction.

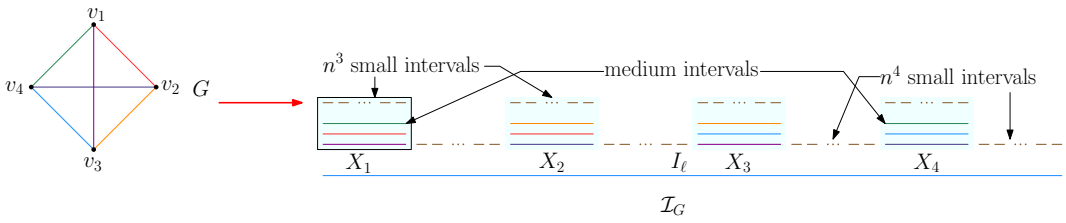

Figure 5 An example reduction.

▶ **Lemma 8.** *The graph G has a vertex cover of size at most k if and only if the corresponding interval graph $G(\mathcal{I}_G)$ has a consistent subset of size at most $K = k(3 + n^3)$.*

Proof. With a slight abuse of notations, we will denote the vertex in $G(\mathcal{I}_G)$ corresponding to an interval $I \in \mathcal{I}_G$ by I. ($\Rightarrow$) Let $A \subseteq V(G)$ be a vertex cover of G. Consider the set of intervals $\mathcal{I}_A = \bigcup_{v_i \in A} X_i$. We prove that $\mathcal{I}_A$ is a consistent subset of $G(\mathcal{I}_G)$. Note that as $|X_i| = 3 + n^3$, $|\mathcal{I}_A| = k(3 + n^3)$.

As the vertices in A cover all the edges in G, $\mathcal{I}_A$ must contain at least one interval of each color $\{c_1, \cdots, c_m\}$. As the unit intervals in X_i associated with a vertex $v_i \in A$ are of colors different from the color of the small intervals in X_i, $\mathcal{I}_A$ contains at least n^3 small intervals. Therefore $\mathcal{I}_A$ contains at least one interval from each color in $\{c_1, \ldots c_{m+1}\}$. Observe that, (i) the interval $I_\ell \in \mathcal{I}_4$ of color c_1 contains an interval of color c_1 that corresponds to the edge e_1, and (ii) the distance between any two nodes corresponding to medium intervals in two different vertex gadgets of $G(\mathcal{I}_G)$ is 2 (via the node corresponding to the interval I_ℓ in $G(\mathcal{I}_G)$). Thus, $\mathcal{I}_A$ is a consistent subset.

($\Leftarrow$) Let $\mathcal{I}_B \subseteq \mathcal{I}_G$ be any consistent subset of $G(\mathcal{I}_G)$ of cardinality $(3 + n^3)k$. Now, if $\mathcal{I}_B$ contains I_ℓ then $|\mathcal{I}_G| - 2 \leq |\mathcal{I}_B| \leq |\mathcal{I}_G|$ because if $e_1 = (v_i, v_j)$ be the edge of color c_1, then we can only do not take the medium intervals of color c_1 from the vertex gadget X_i and X_j in $\mathcal{I}_B$ because they are covered by I_ℓ, which contradicts the fact that $|\mathcal{I}_B| = (3 + n^3)k$. Thus, we have $I_\ell \notin \mathcal{I}_B$.

By the definition, $\mathcal{I}_B$ contains at least one color from $\{c_1, \cdots, c_m, c_{m+1}\}$. Also, if $\mathcal{I}_B$ contains one interval from the gadget X_i of any vertex v_i, then it must contain all the intervals from X_i; otherwise, it can not be a consistent subset. Thus $\mathcal{I}_B$ is the union $\mathcal{X}$ of at most k sets from $\{X_i : i \in [n]\}$, and it contains at least one interval from each color $\{c_1, \cdots, c_m\}$, and a few intervals of color c_{m+1}. Now, consider the set $V_B = \{v_i : X_i \in \mathcal{X}\}$, which is a vertex cover for the graph G of size at most k.

Hence, the lemma is proved. ◀

6 Conclusion

We have shown that MCS is NP-complete for trees and interval graphs, and have given an exact algorithm parameterized w.r.t. the number of colors for trees. As a direction for future research, possibilities of approximating MCS can be explored for interval graphs and related graph classes like circle graphs, circular arc graphs, etc. Parameterized algorithms may also be explored where, in addition to a parameter for the number of colours, there is also a parameter specifying the structural properties of the input graph.

References

1 Hiroki Arimura, Tatsuya Gima, Yasuaki Kobayashi, Hiroomi Nochide, and Yota Otachi. Minimum consistent subset for trees revisited. *CoRR*, abs/2305.07259, 2023. doi:10.48550/arXiv.2305.07259.

2 Sandip Banerjee, Sujoy Bhore, and Rajesh Chitnis. Algorithms and hardness results for nearest neighbor problems in bicolored point sets. In Michael A. Bender, Martín Farach-Colton, and Miguel A. Mosteiro, editors, *LATIN 2018: Theoretical Informatics*, pages 80–93, 2018. doi:10.1007/978-3-319-77404-6_7.

3 Ahmad Biniaz, Sergio Cabello, Paz Carmi, Jean-Lou De Carufel, Anil Maheshwari, Saeed Mehrabi, and Michiel Smid. On the minimum consistent subset problem. *Algorithmica*, 83(7):2273–2302, 2021. doi:10.1007/S00453-021-00825-8.

4 Ahmad Biniaz and Parham Khamsepour. The minimum consistent spanning subset problem on trees. *Journal of Graph Algorithms and Applications*, 28(1):81–93, 2024. doi:10.7155/JGAA.V28I1.2929.

5 Rajesh Chitnis. Refined lower bounds for nearest neighbor condensation. In Sanjoy Dasgupta and Nika 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. URL: https://proceedings.mlr.press/v167/chitnis22a.html.

6 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

7 Sanjana Dey, Anil Maheshwari, and Subhas C. Nandy. Minimum consistent subset problem for trees. In Evripidis Bampis and Aris 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. doi:10.1007/978-3-030-86593-1_14.

8 Sanjana Dey, Anil Maheshwari, and Subhas C. Nandy. Minimum consistent subset of simple graph classes. *Discret. Appl. Math.*, 338:255–277, 2023. doi:10.1016/J.DAM.2023.05.024.

9 Reinhard Diestel. Graph theory, volume 173 of. *Graduate texts in mathematics*, page 7, 2012.

10 Michael R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

11 Peter E. Hart. The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 14(3):515–516, 1968. doi:10.1109/TIT.1968.1054155.

12 Kamyar Khodamoradi, Ramesh Krishnamurti, and Bodhayan Roy. Consistent subset problem with two labels. In B.S. Panda and Partha P. Goswami, editors, *Algorithms and Discrete Applied Mathematics*, pages 131–142, 2018. doi:10.1007/978-3-319-74180-2_11.

13 Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 475–484, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258641.

14 Vijay V. Vazirani. *Approximation Algorithms*. Springer Publishing Company, Incorporated, 2010.

15 Gordon Wilfong. Nearest neighbor problems. In *Proceedings of the Seventh Annual Symposium on Computational Geometry*, SCG '91, pages 224–233, 1991. doi:10.1145/109648.109673.