

Monopolar Graphs: Complexity of Computing Classical Graph Parameters^{*}

Michele Barbato^{*}, Dario Bezzi

Università degli Studi di Milano, Dipartimento di Informatica, OptLab, Via Bramante 65, 26013 Crema (CR), Italy

Abstract

A graph $G = (V, E)$ is monopolar if V can be partitioned into a stable set and a set inducing the union of vertex-disjoint cliques. Motivated by an application of the clique partitioning problem on monopolar graphs to the cosmetic manufacturing, we study the complexity of computing classical graph parameters on the class of monopolar graphs. We show that computing the clique partitioning, stability and chromatic numbers of monopolar graphs is NP-hard. Conversely, we prove that every monopolar graph has a polynomial number of maximal cliques thus obtaining that a maximum-weight clique can be found in polynomial time on monopolar graphs.


Keywords: Computational complexity, Monopolar graph, Maximum-weight clique, Clique partitioning, Stable set, Graph coloring

1. Introduction

We consider simple undirected graphs whose terminology can be found in [3]. Given a graph $G = (V, E)$, a partition (A, B) of V is *monopolar* if A is a stable set and $G[B]$, the graph induced by B in G , is a *cluster*, that is, the union of vertex-disjoint cliques. The graph G is *monopolar* if its vertex set admits a monopolar partition.

Recently, monopolar graphs have been used to detect core-periphery structure of protein interaction networks [4]. ILP formulations and heuristic methods are given in [4] to extract a monopolar subgraph from a general graph by removing as few edges as possible. Here, the input graph represents a protein interaction network measurement affected by independent stochastic errors and the extracted monopolar subgraph gives a finer approximation of the real structure of the observed network.

Our interest in monopolar graphs stems from their relation to another real-world problem, which arises in cosmetic manufacturing and is described at the end of this introduction.

^{*}© 2021. This manuscript version is made available under the [CC-BY-NC-ND 4.0 license](https://creativecommons.org/licenses/by-nc-nd/4.0/) .
Journal version DOI: [10.1016/j.dam.2020.12.023](https://doi.org/10.1016/j.dam.2020.12.023)

^{*}Corresponding author.

Email addresses: michele.barbato@unimi.it (Michele Barbato), dario.bezzi@unimi.it (Dario Bezzi)

From a theoretical perspective, monopolar graphs have been mainly studied in connection with other graph classes, such as polar graphs first defined in [26] and unipolar graphs treated, *e.g.*, in [8, 12, 25]. All these classes can be concisely described by means of the following definition used in [18]. Given Π_A and Π_B two graph properties, $G = (V, E)$ is a (Π_A, Π_B) -graph if V is partitionable into A and B such that $G[A]$ has property Π_A and $G[B]$ has property Π_B . Monopolar graphs are easily seen to be the $(K_2$ -free, P_3 -free)-graphs, see *e.g.*, [4]. Similarly, *polar* graphs can be defined as the $(\overline{P_3}$ -free, P_3 -free)-graphs and the *unipolar* graphs as the $(\overline{K_2}$ -free, P_3 -free)-graphs. Note that polar graphs generalize both unipolar and monopolar graphs.

Most of works concerned with monopolar graphs are focused on the *monopolarity recognition problem*, consisting in deciding whether a given input graph is monopolar. Monopolarity recognition is relevant for solving the analogous problem of recognizing polar graphs. Indeed, for several special classes of input graphs, the monopolarity recognition problem admits polynomial-time algorithms which are also used as subroutines to efficiently recognize polar graphs in those classes, see *e.g.*, [6, 10, 11]. Other efficient algorithms for monopolarity recognition are given if the number of maximal cliques in the cluster induced by a monopolar partition is treated as a fixed parameter [18], for superclasses of chair-free and hole-free input graphs, and for classes of input graphs with bounded clique- or tree-width, see [20] and the references therein. On the other hand, the results in [13] imply that it is **NP**-complete to recognize (mono)polar graphs in general and the same holds for (mono)polarity recognition of K_3 -free input graphs [7, 19] and K_3 -free planar input graphs of maximum degree three [20].

The **NP**-completeness of recognizing (mono)polar graphs contrasts with the fact that unipolar graphs can be recognized in polynomial time, as shown in [8, 12, 25]. In fact, [12] also shows that unipolar graphs are perfect (see *e.g.*, [17, Sect. 9.2] for the definition of perfect graphs). Hence it is well-known [17, Chapt. 9] that the stability, chromatic, clique and clique partitioning numbers of unipolar graphs can be computed in polynomial time and, to this end, specific combinatorial algorithms exploiting the unipolar structure are provided in [12].

Conversely, little seems to be known about the complexity of determining standard parameters on monopolar graphs. Results in this sense are obtained after observing that monopolar graphs generalize both bipartite and split graphs; hence **NP**-hard problems for the latter classes of graphs are also **NP**-hard for the former class. Examples include computing the edge clique partition number [27] and the dominating and independent dominating numbers [2, 9]. Contrary to bipartite and split graphs though, monopolar graphs are easily seen to be non-perfect and the complexity of computing the stability, chromatic, clique and clique partitioning numbers is, to the best of our knowledge, unknown. We are only aware of polynomial-time algorithms for computing the stability number in monopolar $2P_3$ -free graphs, see [21], and the clique and stability numbers of (mono)polar graphs which are trivially perfect [23] (see *e.g.*, [16] for the definition of trivially perfect graphs).

Contribution. We contribute to the investigation on the complexity of computing classical graph parameters on monopolar graphs. We prove that determining the clique partitioning,

stability and chromatic numbers on monopolar graphs is **NP**-hard. The **NP**-hardness of the chromatic number computation is derived from the **NP**-completeness of the **3-COLORABILITY** problem on monopolar graphs. The **NP**-hardness of computing the clique partitioning number is proven along with the **NP**-hardness of recognizing a positive clique partitioning-stability number gap on monopolar graphs. All these complexity results are obtained by reductions of classical **NP**-complete problems and involve graphs whose vertex set is explicitly partitioned in a monopolar fashion. Hence they hold even if a monopolar partition is known. Clearly, they also extend to the more general class of polar graphs and to the weighted versions of the considered problems. Subsequently, we prove that the **MAX WEIGHT CLIQUE** problem can be solved in polynomial time on monopolar graphs. We derive this latter result from the fact that the number of maximal cliques of a monopolar graph is linearly bounded by the number of its vertices and edges.

A Monopolar Graph Model for Manufacturing. We conclude the introduction by describing the aforementioned real-world problem that can be modelled by means of monopolar graphs. The problem, which we call **DISJOINT COVERING**, is as follows. We are given a set of *ingredients* N , a set of *containers* $\mathcal{C} = \{C_1, \dots, C_k\}$ with $C_i \subseteq N$ for every $i \in \{1, \dots, k\}$ and a set of d cosmetic products, each obtained by combining ingredients in the containers. Let $\mathcal{P} = \{P_1, \dots, P_d\}$ be the set of *products*, such that $P_j \subseteq N$ for every $j \in \{1, \dots, d\}$, that is, we identify each product with the list of ingredients it requires. The goal is to decide whether there exist d subsets $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_d$ of \mathcal{C} such that:

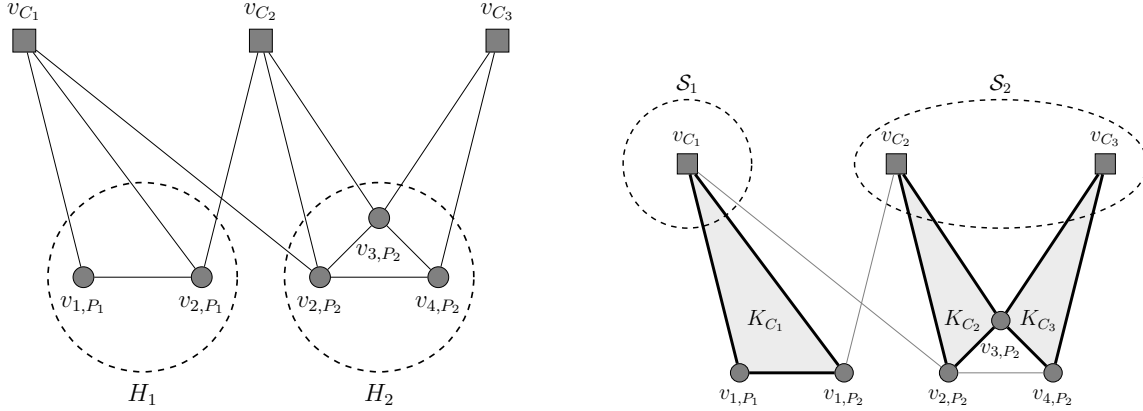
$$P_j \subseteq \bigcup_{C \in \mathcal{S}_j} C \text{ for every } j = 1, \dots, d; \quad (\text{Cov})$$

$$\mathcal{S}_j \cap \mathcal{S}_\ell = \emptyset \text{ for all distinct } j, \ell \in \{1, \dots, d\}. \quad (\text{Disj})$$

The sets N , \mathcal{C} and \mathcal{P} define a *feasible* instance of the **DISJOINT COVERING** problem whenever such disjoint subsets exist.

A feasible instance of the manufacturing problem admits an assignment of each container to at most one product, since every product requires a series of time-consuming tasks to be performed on its assigned container that would otherwise introduce delays. Therefore, the covering condition (**Cov**) guarantees that every product can be obtained by using the ingredients in its assigned containers. The disjointness condition (**Disj**) imposes that an ingredient in a container assigned to product P_j with $j \in \{1, \dots, d\}$ cannot be used to make product P_ℓ with $\ell \neq j$, even if P_j does not require that ingredient.

Let I be an instance of the **DISJOINT COVERING** problem defined by N , \mathcal{C} and \mathcal{P} as above. We model I as a monopolar graph $\mathcal{G}_I = (\mathcal{V}_I, \mathcal{E}_I)$ as follows. The vertex set \mathcal{V}_I is given by the union of two sets A and B such that A contains a vertex v_C for each element of $C \in \mathcal{C}$ and B contains a vertex $v_{i,P}$ for every pair $\{i, P\}$ with $i \in N$ and $P \in \mathcal{P}$ such that $i \in P$. The edge set \mathcal{E}_I is obtained by linking v_C and $v_{i,P}$ whenever $i \in C \cap P$ and by linking $v_{i,P}$ and $v_{j,P}$ whenever $i, j \in P$. Then (A, B) is a monopolar partition of \mathcal{V}_I , since A is a stable set and $\mathcal{G}_I[B]$ is a cluster whose maximal cliques are in one-to-one correspondence with the products in \mathcal{P} . An example of the above correspondence is given in Fig. 1.1a.



(a) A DISJOINT COVERING instance I with ingredient set $N = \{1, 2, 3, 4\}$ represented as a monopolar graph \mathcal{G}_I . Containers in \mathcal{C} are $C_1 = \{1, 2\}$, $C_2 = \{2, 3\}$, $C_3 = \{3, 4\}$; products in \mathcal{P} are $P_1 = \{1, 2\}$ and $P_2 = \{2, 3, 4\}$, respectively corresponding to the maximal cliques H_1 and H_2 of $\mathcal{G}_I[B]$.

(b) The shaded triangles give a clique covering of \mathcal{G}_I and correspond to the solution $\{\mathcal{S}_1, \mathcal{S}_2\}$ to the DISJOINT COVERING problem. Note that K_{C_1} and K_{C_3} are not disjoint but a clique partitioning is obtained by removing v_{3,P_2} from K_{C_3} .

Figure 1.1: Example of correspondence between DISJOINT COVERING instances and monopolar graphs. Square vertices are in A , round vertices in B .

In the remainder of this paper, the symbols $\alpha(G)$, $\chi(G)$ and $\omega(G)$ respectively denote the stability, chromatic and clique number of a graph G , see *e.g.*, [3] for their definitions. The clique partitioning number is indicated by $\bar{\chi}(G)$ to emphasize that it equals the chromatic number of the complement \bar{G} , see *e.g.*, [17, Sect. 9.4].

Prop. 1.1 reveals a relation between the DISJOINT COVERING problem and the clique partitioning number of monopolar graphs.

Proposition 1.1. *Let N , \mathcal{C} and \mathcal{P} define an instance I of the DISJOINT COVERING problem and let \mathcal{G}_I be its corresponding graph. Then I is feasible if and only if $\bar{\chi}(\mathcal{G}_I) = |\mathcal{C}|$.*

Proof. Throughout the proof we use the notation adopted in the description of $\mathcal{G}_I = (\mathcal{V}_I, \mathcal{E}_I)$. We observed that $A = \{v_C : C \in \mathcal{C}\}$ is a stable set of \mathcal{G}_I and (A, B) with $B = \mathcal{V}_I \setminus A$ is a monopolar partition. For $j = 1, \dots, d$, let $H_j = \{v_{i,P_j} : i \in P_j\}$, be the maximal clique of $\mathcal{G}_I[B]$ corresponding to $P_j \in \mathcal{P}$ (see Fig. 1.1a for an example).

Let us first show that if I is feasible then $\bar{\chi}(\mathcal{G}_I) = |\mathcal{C}|$. Since A is a stable set of \mathcal{G}_I , we get $\bar{\chi}(\mathcal{G}_I) \geq |\mathcal{C}| = |A|$. To show that $\bar{\chi}(\mathcal{G}_I) \leq |\mathcal{C}|$ it is enough to exhibit a clique cover of \mathcal{G}_I having cardinality $|\mathcal{C}|$: indeed every clique cover can be turned into a clique partition of the same cardinality by removing the intersection of two cliques from exactly one of them.

If I is feasible there exist d subsets $\mathcal{S}_1, \dots, \mathcal{S}_d$ of \mathcal{C} satisfying conditions (Cov) and (Disj). For every $C \in \mathcal{C}$ we construct a clique K_C of \mathcal{G}_I as follows: if $C \notin \mathcal{S}_1 \cup \dots \cup \mathcal{S}_d$ we set $K_C = \{v_C\}$; otherwise, $C \in \mathcal{S}_j$ for some $j \in \{1, \dots, d\}$ and we define K_C as the clique induced in \mathcal{G}_I by v_C and its neighbors in H_j (see Fig. 1.1b for an illustration). Note that by (Disj) the clique K_C is unique for every $C \in \mathcal{C}$. Then $\mathcal{K} = \{K_C : C \in \mathcal{C}\}$ contains exactly $|\mathcal{C}|$ cliques; let us show that together they cover all vertices of \mathcal{G}_I . First, by definition, each vertex $v_C \in A$ is contained in one clique of \mathcal{K} . Moreover, if $v_{i,P_j} \in H_j$ for some

$j \in \{1, \dots, d\}$ then $i \in P_j$ and by (Cov) $i \in C$ for some $C \in \mathcal{S}_j$; from the definition of \mathcal{E}_I , v_{i,P_j} is a neighbor of v_C , hence $v_{i,P_j} \in K_C$. This concludes the first implication.

Now, we prove the opposite implication. Let \mathcal{K} be a clique partition of \mathcal{G}_I consisting of $|\mathcal{C}|$ cliques. For every $j = 1, \dots, d$, let $\mathcal{K}_j \subseteq \mathcal{K}$ be such that every vertex of H_j belongs to a clique of \mathcal{K}_j . The maximal cliques of $\mathcal{G}_I[B]$ are vertex-disjoint, so $\mathcal{K}_j \cap \mathcal{K}_\ell = \emptyset$ for all distinct $j, \ell \in \{1, \dots, d\}$. Being \mathcal{K} a clique partition, every vertex of A belongs to exactly one clique of \mathcal{K} . Thus the sets $\mathcal{S}_j = \{C \in \mathcal{C} : v_C \in K \text{ for some } K \in \mathcal{K}_j\}$ for $j = 1, \dots, d$ satisfy (DISJ). Let $j \in \{1, \dots, d\}$ and $i \in P_j$. Vertex v_{i,P_j} belongs to one $K \in \mathcal{K}_j$. Since \mathcal{K} covers V by $|\mathcal{C}|$ cliques and A is a stable set of size $|\mathcal{C}|$, there exists C with $v_C \in K$. Then $\{v_C, v_{i,P_j}\} \in \mathcal{E}_I$, that is, $i \in C$. Since C belongs to \mathcal{S}_j , this latter satisfies (Cov). Then I is feasible. \square

2. NP-Hardness Results

Clique Partitioning Monopolar Graphs and Related Problems. The DISJOINT COVERING problem of the introduction is easily seen to be in **NP**. We now prove that it is **NP**-complete. This, together with Prop. 1.1 and the monopolarity of \mathcal{G}_I for every DISJOINT COVERING instance I , implies that it is **NP**-hard to compute the clique partitioning number of generic monopolar graphs.

Our construction relies on a reduction from the well-known MINIMUM COVER problem. An instance of the MINIMUM COVER problem is a triple $(\mathcal{U}, \mathcal{T}, \ell)$ where \mathcal{U} is a set, \mathcal{T} is a collection of k subsets of \mathcal{U} such that $\bigcup_{T \in \mathcal{T}} T = \mathcal{U}$ and $\ell \leq k$ is a positive integer. A subset $\mathcal{T}' \subseteq \mathcal{T}$ such that $\bigcup_{T \in \mathcal{T}'} T = \mathcal{U}$ is said to *cover* \mathcal{U} , and it is called a *feasible cover* if it additionally satisfies $|\mathcal{T}'| \leq \ell$. Deciding whether a generic instance of the MINIMUM COVER problem has a feasible cover is **NP**-complete [14, p. 222].

Given a MINIMUM COVER instance $J = (\mathcal{U}, \mathcal{T}, \ell)$ as above, we construct an instance of the DISJOINT COVERING problem described in the introduction as follows. First, let $E = \{e_1, \dots, e_{k-\ell}\}$ be a set of dummy elements such that $e_i \notin \mathcal{U}$ for $i = 1, \dots, k - \ell$. We define ingredients $N = \mathcal{U} \cup E$, containers $\mathcal{C} = \{T \cup E : T \in \mathcal{T}\}$ and $\mathcal{P} = \{\mathcal{U}, \{e_i\} : i = 1, \dots, k - \ell\}$. Let I_J be the DISJOINT COVERING instance defined by N, \mathcal{C} and \mathcal{P} . We observe that the size of I_J is polynomial in the size of J .

Lemma 2.1. *Instance J has a feasible cover if and only if I_J is feasible. Thus, the DISJOINT COVERING problem is **NP**-complete.*

Proof. It is not restrictive to assume that a feasible cover \mathcal{T}' of J consists of ℓ elements of \mathcal{T} . Let $\mathcal{T} \setminus \mathcal{T}' = \{T_1, \dots, T_{k-\ell}\}$. We consider the partition of \mathcal{C} given by the sets $\mathcal{S}_i = \{T_i \cup E\}$ for every $i = 1, \dots, k - \ell$ and $\mathcal{S}_{k-\ell+1} = \{T \cup E : T \in \mathcal{T}'\}$. We assign \mathcal{S}_i to product $P_i = \{e_i\}$ for every $i = 1, \dots, k - \ell$ and $\mathcal{S}_{k-\ell+1}$ to product $P_{k-\ell+1} = \mathcal{U}$. Then I_J is feasible since $|\mathcal{P}| = k - \ell + 1$ and \mathcal{T}' covers \mathcal{U} .

Conversely, if I_J is feasible, there exist $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{k-\ell+1}$ subsets of \mathcal{C} satisfying conditions (Cov) and (DISJ). Without loss of generality, for every $i = 1, 2, \dots, k - \ell$ the subset \mathcal{S}_i satisfies condition (Cov) on product $P_i = \{e_i\}$ while the same condition holds for $\mathcal{S}_{k-\ell+1}$ and product $P_{k-\ell+1} = \mathcal{U}$. Letting $\mathcal{S}_{k-\ell+1} = \{C_1, C_2, \dots, C_h\}$ for some $h \geq 1$ we have that $\mathcal{U} \subseteq \bigcup_{i=1}^h C_i$. Since $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{k-\ell+1}$ are pairwise disjoint, $|\mathcal{C}| = k$ implies

$h \leq \ell$. Finally, $\mathcal{T}' = \{T_1, \dots, T_h\}$ defined by $T_i = C_i \setminus E$ for every $i \in \{1, \dots, h\}$ is a feasible cover of J , since $T_i \subseteq \mathcal{T}$ and $e_j \notin \mathcal{U}$ for $j = 1, \dots, k - \ell$, so \mathcal{T}' covers \mathcal{U} . Hence the DISJOINT COVERING problem is **NP**-complete. \square

Proposition 2.2. *Computing the clique partitioning number on the class of monopolar graphs is **NP**-hard.*

Proof. Immediate from Prop. 1.1 and Lemma 2.1, the graph \mathcal{G}_I being monopolar for every DISJOINT COVERING instance I , as proven in the introduction. \square

The specific structure of instance I_J constructed for the proof of Lemma 2.1 also allows us to prove that it is **NP**-hard to determine whether $\bar{\chi}(G) = \alpha(G)$ for a monopolar graph G . This latter problem has been shown to be **NP**-hard on generic graphs in [5]. For next proposition, we adapt the proof of [5].

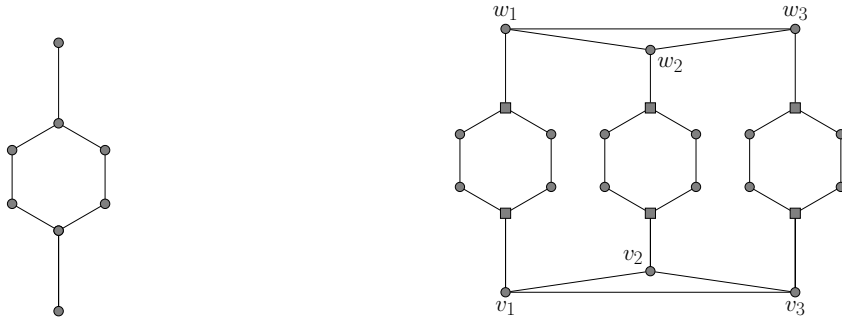
Proposition 2.3. *Deciding whether $\bar{\chi}(G) = \alpha(G)$ for a generic monopolar graph G is **NP**-hard even if some maximum stable set of G is known.*

Proof. Given an instance $J = (\mathcal{U}, \mathcal{T}, \ell)$ of the MINIMUM COVER problem, let I_J be the DISJOINT COVERING instance constructed as above, with \mathcal{C} its set of containers. Let also $\mathcal{G}_{I_J} = (\mathcal{V}_{I_J}, \mathcal{E}_{I_J})$ be the graph corresponding to I_J as described in the introduction. Clearly, \mathcal{G}_{I_J} has size polynomial in the size of J . Moreover, \mathcal{V}_{I_J} has a monopolar partition (A, B) with every vertex in A corresponding to an element of \mathcal{C} and every maximal clique of $\mathcal{G}_{I_J}[B]$ corresponding to an element of \mathcal{P} . In particular, \mathcal{G}_{I_J} has a vertex in B for every set $\{e_i\}$ with $i = 1, \dots, k - \ell$. We call F the set of these vertices. Then $A \cup F$ induces a complete bipartite subgraph of \mathcal{G}_{I_J} . From $\ell \geq 1$ we get $|F| \leq |A| - 1$. By construction, every vertex in the maximal clique of $\mathcal{G}_{I_J}[B]$ corresponding to \mathcal{U} is adjacent to at least one vertex in A , since \mathcal{T} covers \mathcal{U} . Finally, we observe that $|\mathcal{C}| = |\mathcal{T}|$, so $\alpha(\mathcal{G}_{I_J}) = |A| = |\mathcal{C}| = |\mathcal{T}|$. By Prop. 1.1 and Lemma 2.1, a polynomial-time algorithm for deciding whether $\bar{\chi}(G) = \alpha(G)$ for every monopolar graph G allows one to determine whether $\bar{\chi}(\mathcal{G}_{I_J}) = |\mathcal{T}|$ and, as a consequence, whether J has a feasible cover. This proves the result. \square

Stability Number of Monopolar Graphs. We give a reduction of the 3-COLORABILITY problem on general graphs to the stable set problem on monopolar graphs. In the 3-COLORABILITY problem we have to decide whether a given input graph admits a proper coloring with at most three colors. The 3-COLORABILITY problem is **NP**-complete, see [14, p. 191].

For our purposes, we consider the gadget shown in Fig. 2.1a. Its vertices of degree one will be called *extreme*. Let $G = (V, E)$ be a graph. We construct a graph $H_G = (V_G, E_G)$ from G by replacing each vertex $v \in V$ by three vertices v_1, v_2 and v_3 linked to form a K_3 and by joining the two cliques corresponding to v and w as in Fig. 2.1b whenever $\{v, w\} \in E$. More precisely, for every pair $\{v_i, w_i\}$ where $i = 1, 2, 3$ and $\{v, w\}$ is an edge of G , we add a gadget having v_i and w_i as extreme vertices.

Computing $\alpha(H_G)$ is enough to solve the 3-COLORABILITY problem on G , as we prove in next lemma.



(a) Gadget used in H_G .

(b) Transformation of an edge $\{v, w\}$ of G into a monopolar subgraph of H_G . Square vertices are a stable set, round vertices induce a cluster.

Figure 2.1

Lemma 2.4. *Let $G = (V, E)$ be a graph and $H_G = (V_G, E_G)$ be the associated graph defined above. Then G is 3-colorable if and only if $\alpha(H_G) = |V| + 9|E|$.*

Proof. Let $\mathcal{I} = \{1, 2, 3\}$, $C = \{v_i \in V_G : v \in V, i \in \mathcal{I}\}$ and $D = V_G \setminus C$. The graph $H_G[C]$ is a cluster consisting of $|V|$ vertex-disjoint K_3 graphs, hence $\alpha(H_G[C]) = |V|$. The graph $H_G[D]$ is the union of $3|E|$ vertex-disjoint cycles of length six, thus $\alpha(H_G[D]) = 9|E|$. Since C and D partition V_G , we get that $\alpha(H_G) \leq |V| + 9|E|$. The same argument also proves that the right-hand-side value is reached only by the cardinality of stable sets including exactly one vertex for each K_3 corresponding to a vertex of V and exactly three vertices per cycle being part of the gadgets corresponding to the edges of G .

So, if S is a maximum stable set of H_G of cardinality $|V| + 9|E|$, we get that $v_i \in S$ implies $w_i \notin S$ whenever $\{v, w\} \in E$. Otherwise, S would contain at most two vertices in the cycle of the gadget having v_i and w_i as extreme vertices. It follows that, whenever $\{v, w\} \in E$, if $v_i \in S$ for some $i \in \mathcal{I}$ then $w_j \in S$ for some $j \in \mathcal{I} \setminus \{i\}$, as S contains one vertex for each K_3 corresponding to a vertex of V . As a consequence, assigning color $i \in \mathcal{I}$ to vertex v such that $v_i \in S$ yields a proper coloring of G using at most three colors.

Conversely, let G be 3-colorable with colors in \mathcal{I} . We define the stable set $S_1 = \{v_i \in V_G : v \in V \text{ has color } i \in \mathcal{I}\}$. Let us consider the graph H'_G obtained from H_G by removing all vertices in S_1 and their neighbors. Since every vertex $v \in V$ is assigned a color this implies that all K_3 graphs corresponding to the vertices of G are removed. Moreover, at most one vertex per gadget is removed since for every edge $\{v, w\} \in E$ vertices v and w are assigned distinct colors. It follows that H'_G has $3|E|$ connected components each being either a path on five vertices or a cycle on six vertices. All these connected components admit a stable set of size three, hence a maximum stable set S_2 of H'_G has size $9|E|$. Now, $S = S_1 \cup S_2$ is a stable set of H_G of cardinality $|V| + 9|E|$, hence it is a maximum stable set of H_G . \square

Proposition 2.5. *Computing the stability number on the class of monopolar graphs is NP-hard.*

Proof. The size of $H_G = (V_G, E_G)$ is polynomial in the size of G . By Lemma 2.4 it is enough to prove that H_G is monopolar for every graph G . Let us consider the partition (A, B) of V_G where A contains all vertices of degree three of the gadgets corresponding to the edges of G , while B contains all other vertices of V_G . (Fig. 2.1b illustrates this partition on the graph H_{K_2} .) By construction of H_G the vertices of distinct gadgets corresponding to the edges of G are not adjacent except for their extreme vertices, thus A is a stable set. The same argument shows that a P_3 in $H_G[B]$ can only be induced by three extreme vertices of the gadgets used in the construction of H_G . However, every connected subgraph containing three extreme vertices is a K_3 . Hence $H_G[B]$ is a cluster and (A, B) a monopolar partition of V_G . \square

Chromatic Number of Monopolar Graphs. We now prove that the 3-COLORABILITY problem on monopolar graphs is **NP**-complete. This immediately proves that computing the chromatic number of monopolar graphs is **NP**-hard in general. We adapt a well-known reduction of the 3-SAT problem to 3-COLORABILITY problem on general graphs [15, Thm. 2.1].

An instance of the 3-SAT problem is a set of disjunctive clauses each consisting of three literals from a given set of positive and negated variables. The goal is to determine the existence of a *truth assignment* for the instance, *i.e.*, an assignment of boolean values to the variables making all clauses true. The 3-SAT problem is **NP**-complete [14, p. 259].

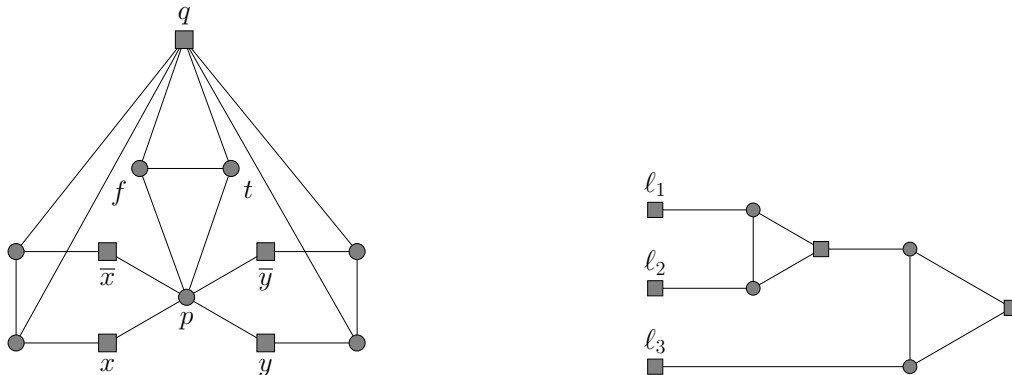
Our reduction relies on two gadgets. The first gadget is constructed by first taking a *diamond* obtained from K_4 by removing an edge. We call p and q the two vertices of the diamond of degree two and f and t the other two vertices. For every variable x of the given instance, we add a cycle of length five having p as a vertex. The neighbors of p in the cycle of variable x will be referred to as x and \bar{x} . Finally, we link the remaining two vertices of the cycle to vertex q . In Fig. 2.2a we illustrate this gadget for two variables x and y .

The second gadget is depicted in Fig. 2.2b and it is the same that is used in [15]. We call it *clause gadget*. A vertex of a clause gadget is a *literal vertex* if it has degree one and a *truth vertex* if it has degree two.

Given an instance I of the 3-SAT problem, we construct a graph G_I as follows. We start with a gadget of the first type as above. Subsequently, for every clause $C = (\ell_1, \ell_2, \ell_3)$ of I we create a clause gadget whose literal vertices are identified with the vertices of the first gadget corresponding to the same literals. Finally, we link the truth vertex of each clause gadget to vertices p and f .

Lemma 2.6. *The graph G_I is monopolar for every instance I of the 3-SAT problem.*

Proof. The vertex set of the gadget of first type admits a monopolar partition (A, B) with $f, p \in B$ and $x, \bar{x} \in A$ for every variable x , see Fig. 2.2a. The vertex set of a clause gadget has a monopolar partition (A, B) in which all literal vertices and the truth vertex belong to A , see Fig. 2.2b. Hence identifying the literal vertices across gadgets of different type and linking all truth vertices to p and f does not break the monopolarity. \square



(a) First monopolar gadget. Square vertices are in A , round vertices in B .

(b) Clause gadget. It is monopolar: square vertices are in A , round vertices in B .

Figure 2.2

We just sketch the proof of next lemma, as the argument is the same as in classical reductions of the 3-SAT problem to the 3-COLORABILITY problem given in [15, Thm. 2.1].

Lemma 2.7. *Given an instance I of the 3-SAT problem, the graph G_I is 3-colorable if and only if there is a truth assignment for I .*

Proof. The gadget of first type is 3-colorable. Let F and T be the colors respectively assigned to f and t and let N be the third color in such a 3-coloring (note that p and q are colored N). A literal is assigned boolean value **true** if the corresponding vertex in the first gadget is colored T , otherwise it is assigned **false**. It is easy to see that, for every variable x of I , vertices x and \bar{x} cannot be colored N and must have distinct colors, so the above is a consistent assignment of boolean values to the variables. As observed in [15], under the above 3-coloring, the truth vertex of a clause gadget can be colored T if and only if at least one literal vertex in the same clause is. Since the truth vertices are all linked to f and p , the graph G_I is 3-colorable if and only if I admits a truth assignment. \square

The proof of the following proposition is now immediate.

Proposition 2.8. *The 3-COLORABILITY problem on monopolar graphs is **NP**-complete. Computing the chromatic number on monopolar graphs is **NP**-hard.*

We conclude the section by considering lower and upper bounds for the chromatic number $\chi(G)$ of a monopolar graph G in terms of its clique number $\omega(G)$. As for general graphs, $\omega(G) \leq \chi(G)$ for every monopolar graph G . However, in general graphs the chromatic number can be arbitrarily larger than the clique number, see [3, pp. 376-377]. Instead let us consider a monopolar graph $G = (V, E)$: if (A, B) is a monopolar partition of V , then the maximal cliques of $G[B]$ can be colored with at most $\omega(G)$ colors. Then we can assign an additional color to all vertices in A to obtain a proper coloring of G . Thus we have $\omega(G) \leq \chi(G) \leq \omega(G) + 1$ for every monopolar graph G . The clique number of a monopolar graph can be computed in polynomial time as we show in next section.

3. Polynomial-Time Algorithms for Clique Problems on Monopolar Graphs

The main result of this section is that for a monopolar graph $G = (V, E)$ the MAX WEIGHT CLIQUE problem $\max\{c(K) : K \text{ is a nonempty clique of } G\}$ is solvable in polynomial time, for all vertex weight functions $c: V \rightarrow \mathbb{R}$. In particular, the clique number of a monopolar graph can be computed efficiently. In the following we consider both the cases in which the monopolar partition is not known in advance or it is. Our starting observation is that in a monopolar graph the number of *maximal* cliques (*i.e.*, those not properly contained in a clique) is linearly bounded above by the number of its vertices and edges.

Lemma 3.1. *Let $G = (V, E)$ be a monopolar graph on n vertices and m edges. The number of maximal cliques of G is in $O(n + m)$.*

Proof. Let (A, B) be a monopolar partition of V . The maximal cliques of $G[B]$ are at most $|B|$. Every $v \in A$ together with its neighborhood in a maximal clique H of $G[B]$ induces a maximal clique of G , whenever the neighborhood of v in H is nonempty. Moreover, an edge incident to $v \in A$ belongs to exactly one such a clique. It follows that there are $O(m)$ maximal cliques of G with a vertex in A and a vertex in B . Every other maximal clique of G is either maximal in $G[B]$ or an isolated vertex of A . Since $|A| + |B| = n$ the above discussion shows that G has $O(n + m)$ maximal cliques. \square

Proposition 3.2. *Let $G = (V, E)$ be a monopolar graph with n vertices and m edges and let $c: V \rightarrow \mathbb{R}$ be a weight function on V . Then the MAX WEIGHT CLIQUE problem defined by c on G can be solved in $O(n^2m + nm^2)$ time. In particular, the size $\omega(G)$ of a largest clique of G can be determined in polynomial time.*

Proof. First, we can check whether all vertices of G have nonpositive weights, in which case a maximum-weight clique of G is a singleton vertex. This preprocessing step can be performed in $O(n)$ by inspection. If at least one vertex has positive weight, we may remove from G all vertices having $c(v) \leq 0$ because, without loss of generality, none of them belongs to a maximum-weight clique. Vertex removals do not affect monopolarity, hence the graph resulting from this preprocessing step is again monopolar with at most n vertices and m edges. The h maximal cliques of a general graph with n vertices and m edges can be listed in $O(hnm)$ time [24]. For each maximal clique K the maximum-weight clique contained in K can be found in $O(n)$ time. Lemma 3.1 guarantees that $h \in O(n + m)$ so the MAX WEIGHT CLIQUE problem on G can be solved in $O(n^2m + nm^2)$ time. \square

When a monopolar partition is known, the MAX WEIGHT CLIQUE problem on a monopolar graph can be solved more quickly.

Proposition 3.3. *Let $G = (V, E)$ be a vertex-weighted monopolar graph with n nodes and m edges and let (A, B) be a monopolar partition of V . Then the MAX WEIGHT CLIQUE problem on G can be solved in $O(n + m)$ time.*

Algorithm 1 Computing a maximum-weight clique of a monopolar graph $G = (V, E)$ with known monopolar partition.

Input: A weight function $c: V \rightarrow \mathbb{R}_+$, a monopolar partition (A, B) of V and for every $v \in B$ its adjacency list $\text{adj}_B(v)$

Output: \mathcal{K}^* and $c(\mathcal{K}^*)$, a maximum-weight clique of G and its weight

```

1:  $\mathcal{K}^* := \emptyset$  and  $c(\mathcal{K}^*) := -1$ 
2:  $\mathcal{I}(v) := 0 \ \forall v \in B$  ▷ index of maximal  $B$  clique containing  $v$ 
3:  $\text{adj}_A(w) := \emptyset \ \forall w \in A$  ▷ adjacency list of  $w$  sorted by the indices of the  $B$  cliques
4:  $i = 0$ 
5: for  $v \in B$  do
6:   if  $\mathcal{I}(v) = 0$  then
7:      $\mathcal{K}_v = \{v\}$  and  $c(\mathcal{K}_v) = c(v)$  ▷ create new clique
8:      $i \leftarrow i + 1$ 
9:      $\mathcal{I}(v) = i$ 
10:   for  $w \in \text{adj}_B(v)$  do
11:     if  $w \in B$  then
12:       Insert  $w$  into  $\mathcal{K}_v$ 
13:        $c(\mathcal{K}_v) = c(\mathcal{K}_v) + c(w)$ 
14:        $\mathcal{I}(w) = i$ 
15:     if  $c(\mathcal{K}_v) > c(\mathcal{K}^*)$  then
16:        $\mathcal{K}^* = \mathcal{K}_v$  and  $c(\mathcal{K}^*) = c(\mathcal{K}_v)$ 
17:   for  $w \in \text{adj}_B(v)$  do
18:     if  $w \in A$  then
19:       Append  $v$  to  $\text{adj}_A(w)$ 
20: for  $v \in A$  do
21:   if  $\text{adj}_A(v) = \emptyset$  and  $c(v) > c(\mathcal{K}^*)$  then
22:      $\mathcal{K}^* = \{v\}$  and  $c(\mathcal{K}^*) = c(v)$ 
23:   else
24:      $l = 0$ 
25:      $\mathcal{K}_{v,0} = \emptyset, c(\mathcal{K}_{v,0}) = -1$ 
26:     for  $w \in \text{adj}_A(v)$  do
27:       if  $\mathcal{I}(w) \neq l$  then
28:         if  $c(\mathcal{K}_{v,l}) > c(\mathcal{K}^*)$  then ▷ check previous clique containing  $v$ 
29:            $\mathcal{K}^* = \mathcal{K}_{v,l}$  and  $c(\mathcal{K}^*) = c(\mathcal{K}_{v,l})$ 
30:          $l = \mathcal{I}(w)$ 
31:          $\mathcal{K}_{v,l} = \{v\}$  and  $c(\mathcal{K}_{v,l}) = c(v)$  ▷ create new clique
32:       Insert  $w$  into  $\mathcal{K}_{v,l}$ 
33:        $c(\mathcal{K}_{v,l}) = c(\mathcal{K}_{v,l}) + c(w)$ 
34:       if  $c(\mathcal{K}_{v,l}) > c(\mathcal{K}^*)$  then ▷ check last clique containing  $v$ 
35:          $\mathcal{K}^* = \mathcal{K}_{v,l}$  and  $c(\mathcal{K}^*) = c(\mathcal{K}_{v,l})$ 
36: return  $\mathcal{K}^*$  and  $c(\mathcal{K}^*)$ 

```

Proof. As in the proof of Prop. 3.2, we assume that all vertices of G have positive weights. Moreover, we assume to be able to check the membership of a vertex to A or B in constant time and to have the adjacency lists of the vertices in B . Then Algorithm 1 solves the MAX WEIGHT CLIQUE problem on G in $O(n + m)$ time.

The first main **for** loop in Algorithm 1 (line 5) inspects the maximal cliques of $G[B]$. It does so in $O(n + m)$ time, because it explores the adjacency list of every vertex of B or considers singleton cliques, when such lists are empty. Each iteration of the first **for** loop keeps track of the maximum-weight clique found so far (lines 15-16). Since the graph G is monopolar, every vertex of A has its neighborhood only in B . Thus the same **for** loop also generates the complete adjacency lists of the vertices in A (lines 17-19). Overall, these lists are constructed in $O(m)$ time in such a way that all neighbors of $v \in A$ belonging to a same clique of $G[B]$ appear consecutively in the adjacency list of v . Exploiting this latter property, the second main **for** loop (line 20) scans in $O(n + m)$ time all maximal cliques of G having one vertex in A (lines 26-32). Indeed every maximal cliques having one vertex in A has all other vertices in precisely one maximal clique of $G[B]$. Also, the maximum-weight clique is updated as in the previous **for** loop.

Since all maximal cliques of G are contained among the maximal cliques of $G[B]$ and those having one vertex in A , our algorithm correctly returns the maximum-weight cliques of G in $O(n + m)$ time. \square

We conclude with three remarks concerning related clique problems on monopolar graphs.

Remark 3.4. At the end of the execution of Algorithm 1, all maximal cliques of G (and possibly some non-maximal cliques) are generated. Thus the algorithm can be easily adapted to solve in polynomial time the problem of enumerating all maximal cliques of a monopolar graph G .

Remark 3.5. Combined with the results shown in [22], Lemma 3.1 also implies that the problem of partitioning the vertices of a monopolar graph into at most k cliques is polynomially solvable whenever k is constant (*i.e.*, it is not part of the input). We sketch the overall idea of a polynomial algorithm, referring the reader to [22, p. 133] for a more detailed treatment. One first shows that only maximal cliques are needed in a k -covering of the vertices into cliques; next, since k is constant and the number of maximal cliques is polynomially bounded, enumerating all subsets of maximal cliques of cardinality k can be done in polynomial time; finally, evaluating whether a set of cliques covers all vertices can be done in quadratic time and this yields the result.

Remark 3.6. Given a graph $G = (V, E)$, the MAX EDGE-WEIGHT CLIQUE problem on G is defined as $\max\{d(K) : K \text{ is a nonempty clique of } G\}$ where $d: E \rightarrow \mathbb{R}$ is an edge-weight function. This problem is **NP**-hard already when $d(e) = 1$ for all $e \in E$ because in this setting it coincides with the problem of determining the clique number of an arbitrary graph. If instead $G = (V, E)$ is monopolar, its MAX EDGE-WEIGHT CLIQUE problem can be solved in polynomial time whenever $d(e) \geq 0$ for all $e \in E$. Indeed, the latter condition on the weights ensures that a maximum-edge-weight clique of G is also maximal,

and Lemma 3.1 guarantees that there are $O(m + n)$ maximal cliques in G , obtainable in polynomial time. For each clique, computing its edge-weight takes $O(m)$ time then we have the result.

When the edge-weight function d has arbitrary sign, the MAX EDGE-WEIGHT CLIQUE problem is **NP**-hard also when restricted to complete graphs: indeed, if $G = (V, E)$ is an arbitrary graph with $d(e) = 1$ for all $e \in E$, we may obtain a complete graph K from G by adding the missing edges and weighting them with suitable negative values so that a maximum-edge-weight clique in K is also a maximum-edge-weight clique in G and vice-versa. This directly implies that the MAX EDGE-WEIGHT CLIQUE problem is **NP**-hard on monopolar graphs when the weight function takes arbitrary sign.

Acknowledgements

This research is partially funded by Regione Lombardia, grant agreement n. E97F17000000009, project AD-COM–Advanced Cosmetic Manufacturing [1].

References

- [1] AD-COM – Advanced Cosmetic Manufacturing. <https://ad-com.net/?lang=en>. Accessed: 2019-01-28.
- [2] A. A. Bertossi. Dominating sets for split and bipartite graphs. *Information processing letters*, 19(1):37–40, 1984.
- [3] J. A. Bondy and U. S. R. Murty. *Graph theory, volume 244 of Graduate Texts in Mathematics*. Springer, New York, 2008.
- [4] S. Bruckner, F. Hüffner, and C. Komusiewicz. A graph modification approach for finding core–periphery structures in protein interaction networks. *Algorithms for Molecular Biology*, 10(1):16, 2015.
- [5] S. Busygin and D. V. Pasechnik. On NP-hardness of the clique partition-independence number gap recognition and related problems. *Discrete Mathematics*, 306(4):460–463, 2006.
- [6] R. Churchley and J. Huang. Line-polar graphs: characterization and recognition. *SIAM Journal on Discrete Mathematics*, 25(3):1269–1284, 2011.
- [7] R. Churchley and J. Huang. On the polarity and monopolarity of graphs. *Journal of Graph Theory*, 76(2):138–148, 2014.
- [8] R. Churchley and J. Huang. Solving partition problems with colour-bipartitions. *Graphs and Combinatorics*, 30(2):353–364, 2014.
- [9] D. G. Corneil and Y. Perl. Clustering and domination in perfect graphs. *Discrete Applied Mathematics*, 9(1):27–39, 1984.

- [10] T. Ekim, P. Heggernes, and D. Meister. Polar permutation graphs. In *International Workshop on Combinatorial Algorithms*, pages 218–229. Springer, 2009.
- [11] T. Ekim, P. Hell, J. Stacho, and D. de Werra. Polarity of chordal graphs. *Discrete Applied Mathematics*, 156(13):2469–2479, 2008.
- [12] E. M. Eschen and X. Wang. Algorithms for unipolar and generalized split graphs. *Discrete Applied Mathematics*, 162:195–201, 2014.
- [13] A. Farrugia. Vertex-partitioning into fixed additive induced-hereditary properties is NP-hard. *The Electronic Journal of Combinatorics.*, 11(1):R46, 2004.
URL: http://www.combinatorics.org/Volume_11/PDF/v11i1r46.pdf.
- [14] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co., New York, 1979.
- [15] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical computer science*, 1(3):237–267, 1976.
- [16] M. C. Golumbic. Trivially perfect graphs. *Discrete Mathematics*, 24(1):105–107, 1978.
- [17] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.
- [18] I. Kanj, C. Komusiewicz, M. Sorge, and E. J. van Leeuwen. Parameterized algorithms for recognizing monopolar and 2-subcolorable graphs. *Journal of Computer and System Sciences*, 92:22–47, 2018.
- [19] H.-O. Le and V. B. Le. The NP-completeness of $(1, r)$ -subcolorability of cubic graphs. *Information Processing Letters*, 81(3):157–162, 2002.
- [20] V. B. Le and R. Nevries. Complexity and algorithms for recognizing polar and monopolar graphs. *Theoretical Computer Science*, 528:1–11, 2014.
- [21] V. V. Lozin and R. Mosca. Polar graphs and maximal independent sets. *Discrete Mathematics*, 306(22):2901–2908, 2006.
- [22] B. Rosgen and L. Stewart. Complexity results on graphs with few cliques. *Discrete Mathematics and Theoretical Computer Science*, 9(1):127–136, 2007.
- [23] M. Talmaciu and E. Nechita. On polar, trivially perfect graphs. *International Journal of Computers Communications & Control*, 5(5):939–945, 2010.
- [24] S. Tsukiyama, M. Ide, H. Ariyoshi, and I. Shirakawa. A new algorithm for generating all the maximal independent sets. *SIAM Journal on Computing*, 6(3):505–517, 1977.

- [25] R. I. Tyshkevich and A. A. Chernyak. Algorithms for the canonical decomposition of a graph and recognizing polarity. *Izvestia Akad. Nauk BSSR, ser. Fiz.-Mat. Nauk*, 6:16–23, 1985.
- [26] R. I. Tyshkevich and A. A. Chernyak. Decomposition of graphs. *Cybernetics*, 21(2):231–242, 1985.
- [27] W. D. Wallis and J. Wu. On clique partitions of split graphs. *Discrete mathematics*, 92(1-3):427–429, 1991.