The (theta, wheel)-free graphs Part III: cliques, stable sets and coloring

Marko Radovanović*, Nicolas Trotignon†, Kristina Vušković‡

April 22, 2019

Abstract

A hole in a graph is a chordless cycle of length at least 4. A theta is a graph formed by three paths between the same pair of distinct vertices so that the union of any two of the paths induces a hole. A wheel is a graph formed by a hole and a vertex that has at least 3 neighbors in the hole. In this series of papers we study the class of graphs that do not contain as an induced subgraph a theta nor a wheel. In Part II of the series we prove a decomposition theorem for this class, that uses clique cutsets and 2-joins, and consequently obtain a polynomial time recognition algorithm for the class. In this paper we further use this decomposition theorem to obtain polynomial time algorithms for maximum weight clique, maximum weight stable set and coloring problems. We also show that for a graph G in the class, if its maximum clique size is ω, then its chromatic number is bounded by $\max\{\omega, 3\}$, and that the class is 3-clique-colorable.

1 Introduction

In this article, all graphs are finite and simple. We say that a graph G *contains* a graph H if H is isomorphic to an induced subgraph of G, and that G is *H-free* if it does not contain H. For a family of graphs $\mathcal{H}$, G is *$\mathcal{H}$-free* if for every $H \in \mathcal{H}$, G is H-free.

A *hole* in a graph is a chordless cycle of length at least 4. A *theta* is a graph formed by three paths between the same pair of distinct vertices so that the union of any two of the

*University of Belgrade, Faculty of Mathematics, Belgrade, Serbia. Partially supported by Serbian Ministry of Education, Science and Technological Development project 174033. E-mail: markor@matf.bg.ac.rs

†CNRS, LIP, ENS de Lyon. Partially supported by ANR project Stint under reference ANR-13-BS02-0007 and by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program Investissements d'Avenir (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR). Also Université Lyon 1, université de Lyon. E-mail: nicolas.trotignon@ens-lyon.fr

‡School of Computing, University of Leeds, and Faculty of Computer Science (RAF), Union University, Belgrade, Serbia. Partially supported by EPSRC grant EP/N0196660/1, and Serbian Ministry of Education and Science projects 174033 and III44006. E-mail: k.vuskovic@leeds.ac.uk

paths induces a hole. A *wheel* is a graph formed by a hole and a vertex that has at least 3 neighbors in the hole.

In this series of papers we study (theta, wheel)-free graphs. This project is motivated and explained in more detail in Part I of the series [3], where two subclasses of (theta, wheel)-free graphs are studied. In Part II of the series [11], we prove a decomposition theorem for (theta, wheel)-free graphs that uses clique cutsets and 2-joins, and use it to obtain an $\mathcal{O}(n^4m)$-time recognition algorithm for the class (where n denotes the number of vertices and m the number of edges of a given graph). In this part we use the decomposition theorem from [11] to obtain further properties of the graphs in the class and to construct polynomial time algorithms for maximum weight clique, maximum weight stable set, and coloring problems. In Part IV of the series [12] we show that the induced version of the k-linkage problem can be solved in polynomial time for (theta, wheel)-free graphs.

The main results and the outline of the paper

Throughout the paper we will denote by $\mathcal{C}$ the class of (theta, wheel)-free graphs. Also, n will denote the number of vertices and m the number of edges of a given graph.

For completeness, in Section 2, we state the decomposition theorem for $\mathcal{C}$ and several other results proved in previous parts that will be needed here. Fundamental for our algorithms are the 2-join decomposition techniques developed in [14] which we also describe here, as well as prove some preliminary lemmas.

In Section 3, we prove that every graph in $\mathcal{C}$ contains a bisimplicial vertex, and use this property to give an $\mathcal{O}(n^2m)$-time algorithm for the maximum weight clique problem on $\mathcal{C}$, as well as to show that the class is 3-clique-colorable.

In Section 4, we give an $\mathcal{O}(n^6m)$-time algorithm for the maximum weight stable set problem on $\mathcal{C}$.

In Section 5, we give an $\mathcal{O}(n^5m)$-time algorithm that optimally colors graphs from $\mathcal{C}$. We also prove that every graph in $\mathcal{C}$, with maximum clique size ω, admits a coloring with at most $\max\{\omega, 3\}$ colors.

Since $\mathcal{C}$ contains all chordal graphs, clearly $\mathcal{C}$ has unbounded clique-width. In Section 6, we show how an example of Lozin and Rauthenbach [8] implies that the class of graphs from $\mathcal{C}$ that have no clique cutset also has unbounded clique-width.

Terminology and notation

A *clique* in a graph is a (possibly empty) set of pairwise adjacent vertices. We say that a clique is *big* if it is of size at least 3. A *stable set* in a graph is a (possibly empty) set of pairwise nonadjacent vertices. A *diamond* is a graph obtained from a complete graph on 4 vertices by deleting an edge. A *claw* is a graph induced by vertices u, v_1, v_2, v_3 and edges uv_1, uv_2, uv_3.

A *path* P is a sequence of distinct vertices $p_1p_2\dots p_k$, $k \geq 1$, such that p_ip_{i+1} is an edge for all $1 \leq i < k$. Edges p_ip_{i+1}, for $1 \leq i < k$, are called the *edges of* P. Vertices p_1 and p_k are the *ends* of P. A cycle C is a sequence of vertices $p_1p_2\dots p_kp_1$, $k \geq 3$, such that $p_1\dots p_k$ is a path and p_1p_k is an edge. Edges p_ip_{i+1}, for $1 \leq i < k$, and edge p_1p_k are called the *edges of* C. Let Q be a path or a cycle. The vertex set of Q is denoted by $V(Q)$. The *length* of Q is the number of its edges. An edge $e = uv$ is a *chord* of Q if $u, v \in V(Q)$, but uv is not an edge of Q. A path or a cycle Q in a graph G is *chordless* if no edge of G is a chord of Q.

Let G be a graph. For $x \in V(G)$, $N(x)$ is the set of all neighbors of x in G, and $N[x] = N(x) \cup \{x\}$. For $S \subseteq V(G)$, $G[S]$ denotes the subgraph of G induced by S. For disjoint subsets A and B of $V(G)$, we say that A is *complete* (resp. *anticomplete*) to B if every vertex of A is adjacent (resp. nonadjacent) to every vertex of B.

In a graph G, a subset S of vertices and/or edges is a *cutset* if its removal yields a disconnected graph.

When clear from the context, we will sometimes write G instead of $V(G)$.

2 Decomposition of (theta, wheel)-free graphs

To state the decomposition theorem for graphs in $\mathcal{C}$ we first define the basic classes involved and then the cutsets used.

Basic classes

We will refer to P-graphs and line graphs of triangle-free chordless graphs (which we now define) as *basic* graphs.

A graph G is *chordless* if no cycle of G has a chord. An edge of a graph is *pendant* if at least one of its endnodes has degree 1. A *branch vertex* in a graph is a vertex of degree at least 3. A *branch* in a graph G is a path of length at least 1 whose internal vertices are of degree 2 in G and whose endnodes are both branch vertices. A *limb* in a graph G is a path of length at least 1 whose internal vertices are of degree 2 in G and whose one endnode has degree at least 3 and the other one has degree 1. Two distinct branches are *parallel* if they have the same endnodes. Two distinct limbs are *parallel* if they share the same vertex of degree at least 3.

Cut vertices of a graph R that are also branch vertices are called the *attaching vertices* of R. Let x be an attaching vertex of a graph R, and let $C_1, \dots, C_t$ be the connected components of $R \setminus x$ that together with x are not limbs of R (possibly, $t = 0$, when all connected components of $R \setminus x$ are limbs). If x is the end of at least two parallel limbs of R, let C_{t+1} be the subgraph of R formed by all the limbs of R with endnode x. The graphs $R[V(C_i) \cup \{x\}]$ (for $i = 1, \dots, t$) and the graph C_{t+1} (if it exists) are the *x-petals* of R.

For any integer $k \geq 1$, a *k-skeleton* is a graph R such that:

(i) R is connected, triangle-free, chordless and contains at least three pendant edges (in particular, R is not a path).

(ii) R has no parallel branches (but it may contain parallel limbs).

(iii) For every cut vertex u of R, every component of $R \setminus u$ has a vertex of degree 1 in R.

(iv) For every vertex cutset $S = \{a, b\}$ of R and for every component C of $R \setminus S$, either $R[C \cup S]$ is a chordless path from a to b, or C contains at least one vertex of degree 1 in R.

(v) For every edge e of a cycle of R, at least one of the endnodes of e is of degree 2.

(vi) Each pendant edge of R is given one label, that is an integer from $\{1, \ldots, k\}$.

(vii) Each label from $\{1, \ldots, k\}$ is given at least once (as a label), and some label is used at least twice.

(viii) If some pendant edge whose one endnode is of degree at least 3 receives label i, then no other pendant edge receives label i.

(ix) If R has no branches then $k = 1$, and otherwise if two limbs of R are parallel, then their pendant edges receive different labels and at least one of these labels is used more then once.

(x) If $k > 1$ then for every attaching vertex x and for every x-petal H of R, there are at least two distinct labels that are used in H. Moreover, if $\overline{H}$ is a union of at least one but not all x-petals, then there is a label i such that both $\overline{H}$ and $(R \setminus \overline{H}) \cup \{x\}$ have pendant edges with label i.

(xi) If $k = 2$, then both labels are used at least twice.

Note that if R is a skeleton, then it edgewise partitions into its branches and its limbs. Also, there is a trivial one-to-one correspondence between the pendant edges of R and the limbs of R: any pendant edge belongs to a unique limb, and conversely any limb contains a unique pendant edge.

If R is a graph, then the *line graph* of R, denoted by $L(R)$, is the graph whose vertices are the edges of R, and such that two vertices of $L(R)$ are adjacent if and only if the corresponding edges are adjacent in R.

A *P-graph* is any graph B that can be constructed as follows:

- Pick an integer $k \geq 1$ and a k-skeleton R.
- Build $L(R)$, the line graph of R. The vertices of $L(R)$ that correspond to pendant edges of R are called *pendant vertices* of $L(R)$, and they receive the same label as their corresponding pendant edges in R.

- Build a clique K with vertex set $\{v_1, \ldots, v_k\}$, disjoint from $L(R)$.
- B is now constructed from $L(R)$ and K by adding edges between v_i and all pendant vertices of $L(R)$ that have label i, for $i = 1, \ldots, k$.

We say that K is the *special clique* of B and R is the *skeleton* of B.

Lemma 2.1 *Every P-graph G contains two distinct branches of length at least 2 (in particular, these two branches both contain a vertex of degree 2).*

PROOF — Let i be a label of G that is used at least twice (it exists by (vii)) and consider two pendant edges of the skeleton R of G that receive this label. Then, by condition (viii) the limbs that contain these pendant edges are of length at least 2, and hence they correspond to branches of length at least 2 in G (note that by (i) the degree of v_i in G is at least 3). □

Lemma 2.2 ([3]) *G is the line graph of a triangle-free chordless graph if and only if G is (wheel, diamond, claw)-free.*

Lemma 2.3 ([11]) *Every P-graph is (theta, wheel, diamond)-free.*

Cutsets

A vertex cutset S is a *clique cutset* if S is a clique. Note that every disconnected graph has a clique cutset: the empty set.

An *almost 2-join* in a graph G is a pair (X_1, X_2) that is a partition of $V(G)$, and such that:

- For $i = 1, 2$, X_i contains disjoint nonempty sets A_i and B_i, such that every vertex of A_1 is adjacent to every vertex of A_2, every vertex of B_1 is adjacent to every vertex of B_2, and there are no other adjacencies between X_1 and X_2.
- For $i = 1, 2$, $|X_i| \geq 3$.

An almost 2-join (X_1, X_2) is a *2-join* when for $i \in \{1, 2\}$, X_i contains at least one path from A_i to B_i, and if $|A_i| = |B_i| = 1$ then $G[X_i]$ is not a chordless path.

We say that $(X_1, X_2, A_1, A_2, B_1, B_2)$ is a *split* of this 2-join, and the sets A_1, A_2, B_1, B_2 are the *special sets* of this 2-join. We often use the following notation: $C_i = X_i \setminus (A_i \cup B_i)$ (possibly, $C_i = \emptyset$).

We are ready to state the decomposition theorem from [11].

Theorem 2.4 ([11]) *If G is (theta, wheel)-free, then G is a line graph of a triangle-free chordless graph or a P-graph, or G has a clique cutset or a 2-join.*

We now describe how we decompose a graph from $\mathcal{C}$ into basic graphs using the cutsets in the above theorem.

Decomposing with clique cutsets

If a graph G has a clique cutset K, then its vertex set can be partitioned into sets (A, K, B), where A and B are nonempty and anticomplete. We say that (A, K, B) is a *split* for the clique cutset K. When (A, K, B) is a split for a clique cutset of a graph G, the *blocks of decomposition* of G with respect to (A, K, B) are the graphs $G_A = G[A \cup K]$ and $G_B = G[K \cup B]$.

A *clique cutset decomposition tree* for a graph G is a rooted tree T defined as follows.

(i) The root of T is G.

(ii) Every non-leaf vertex of T is a graph G' that contains a clique cutset K' with split (A', K', B'). The children of G' in T are the blocks of decomposition $G'_{A'}$ and $G'_{B'}$ of G' with respect to (A', K', B'), and at least one of the graphs $G'_{A'}$ and $G'_{B'}$ do not admit a clique cutset.

(iii) Every leaf of T is a graph with no clique cutset.

(iv) T has at most n leaves.

Theorem 2.5 ([13]) *A clique cutset decomposition tree of an input graph G can be computed in time $O(nm)$.*

Note that for a non-leaf vertex G' of T, the corresponding clique cutset K' of G' is also a clique cutset of G. The following lemmas proved in [3] will also be needed.

Lemma 2.6 ([3]) *If G is a wheel-free graph that contains a diamond, then G has a clique cutset.*

A *star cutset* in a graph is a vertex cutset S that contains a vertex (called a *center*) adjacent to all other vertices of S. Note that a nonempty clique cutset is a star cutset.

Lemma 2.7 ([3]) *If $G \in \mathcal{C}$ has a star cutset, then G has a clique cutset.*

Decomposing with 2-joins

We first state some properties of 2-joins in graphs with no clique cutset. Let $\mathcal{D}$ be the class of all graphs from $\mathcal{C}$ that do not have a clique cutset. By Lemma 2.7, no graph from $\mathcal{D}$ has a star cutset and by Lemma 2.6 no graph from $\mathcal{D}$ contains a diamond. Also, let $\mathcal{D}_{\text{BASIC}}$ be the class of all basic graphs from $\mathcal{C}$ that do not have a clique cutset.

An almost 2-join with a split $(X_1, X_2, A_1, A_2, B_1, B_2)$ in a graph G is *consistent* if the following statements hold for $i = 1, 2$:

(i) Every component of $G[X_i]$ meets both A_i, B_i.

(ii) Every vertex of A_i has a non-neighbor in B_i.

(iii) Every vertex of B_i has a non-neighbor in A_i.

(iv) Either both A_1, A_2 are cliques, or one of A_1 or A_2 is a single vertex, and the other one is a disjoint union of cliques.

(v) Either both B_1, B_2 are cliques, or one of B_1, B_2 is a single vertex, and the other one is a disjoint union of cliques.

(vi) $G[X_i]$ is connected.

(vii) For every vertex v in X_i, there exists a path in $G[X_i]$ from v to some vertex of B_i with no internal vertex in A_i.

(viii) For every vertex v in X_i, there exists a path in $G[X_i]$ from v to some vertex of A_i with no internal vertex in B_i.

Note that the definition contains redundant statements (for instance, (vi) implies (i)), but it is convenient to list properties separately as above.

Lemma 2.8 ([3]) *If $G \in \mathcal{D}$, then every almost 2-join of G is consistent.*

By this lemma every 2-join of a graph of $\mathcal{D}$ is consistent.

We now define the blocks of decomposition of a graph with respect to a 2-join. Let G be a graph and $(X_1, X_2, A_1, A_2, B_1, B_2)$ a split of a 2-join of G. Let k_1 and k_2 be positive integers. The *blocks of decomposition* of G with respect to (X_1, X_2) are the two graphs $G_1^{k_1}$ and $G_2^{k_2}$ that we describe now. We obtain $G_1^{k_1}$ from G by replacing X_2 by a *marker path* $P_2 = a_2 \dots b_2$ of length k_1, where a_2 is a vertex complete to A_1, b_2 is a vertex complete to B_1, and $V(P_2) \setminus \{a_2, b_2\}$ is anticomplete to X_1. The block $G_2^{k_2}$ is obtained similarly by replacing X_1 by a marker path $P_1 = a_1 \dots b_1$ of length k_2.

In [11] the blocks of decomposition w.r.t. a 2-join that we used in construction of a recognition algorithm had marker paths of length 2. In this paper we will use blocks whose marker paths are of length 3. So, unless otherwise stated, when we say that G_1 and G_2 are blocks of decomposition w.r.t. a 2-join we will mean that their marker paths are of length 3.

Lemma 2.9 ([3]) *Let G be a graph with a consistent 2-join (X_1, X_2) and G_1, G_2 be the blocks of decomposition with respect to this 2-join whose marker paths are of length 2. Then the following hold:*

(i) G has no clique cutset if and only if G_1 and G_2 have no clique cutset.

(ii) $G \in \mathcal{C}$ if and only if G_1 and G_2 are in $\mathcal{C}$.

Lemma 2.10 *Let G be a graph from $\mathcal{D}$. Let (X_1, X_2) be a 2-join of G, and G_1, G_2 the blocks of decomposition with respect to this 2-join whose marker paths are of length at least 2. Then G_1 and G_2 are in $\mathcal{D}$ and they do not have star cutsets.*

PROOF — By Lemma 2.8, (X_1, X_2) is consistent. Let G'_1 and G'_2 be blocks of decomposition w.r.t. (X_1, X_2) whose marker paths are of length 2. Then for $i \in \{1, 2\}$, G_i is obtained from G'_i by subdividing (0 or several times) an edge of its marker path. Subdividing an edge whose one endnode is of degree 2 cannot create a clique cutset, nor a theta, nor a wheel, and hence the result follows from Lemma 2.9 and Lemma 2.7. □

A 2-join (X_1, X_2) of G is a *minimally-sided 2-join* if for some $i \in \{1, 2\}$ the following holds: for every 2-join (X'_1, X'_2) of G, neither $X'_1 \subsetneq X_i$ nor $X'_2 \subsetneq X_i$. In this case X_i is a *minimal side* of this minimally-sided 2-join.

A 2-join (X_1, X_2) of G is an *extreme 2-join* if for some $i \in \{1, 2\}$ and all $k \geq 3$ the block of decomposition G_i^k has no 2-join. In this case X_i is an *extreme side* of such a 2-join.

Graphs in general do not necessarily have extreme 2-joins (an example is given in [14]), but it is shown in [14] that graphs with no star cutset do. It is also shown in [14] that if G has no star cutset then the blocks of decomposition w.r.t. a 2-join whose marker paths are of length at least 3, also have no star cutset. This is then used to show that in a graph with no star cutset, a minimally-sided 2-join is extreme. We summarize these results in the following lemma.

Lemma 2.11 ([14]) *Let G be a graph with no star cutset. Let $(X_1, X_2, A_1, A_2, B_1, B_2)$ be a split of a minimally-sided 2-join of G with X_1 being a minimal side, and let G_1 and G_2 be the corresponding blocks of decomposition whose marker paths are of length at least 3. Then the following hold:*

(i) $|A_1| \geq 2$, $|B_1| \geq 2$, and in particular all the vertices of $A_2 \cup B_2$ are of degree at least 3.

(ii) If G_1 and G_2 do not have star cutsets, then (X_1, X_2) is an extreme 2-join, with X_1 being an extreme side (in particular, G_1 has no 2-join).

The following simple lemma is useful and not proved in the previous papers of the series.

Lemma 2.12 *Let G be in $\mathcal{D}$. Let $(X_1, X_2, A_1, A_2, B_1, B_2)$ be a split of a minimally-sided 2-join of G with X_1 being a minimal side, and let G_1 and G_2 be the corresponding blocks of decomposition. If the block of decomposition G_1 is a P-graph, then X_1 contains a vertex that has degree 2 in G.*

PROOF — By Lemma 2.1, G_1 contains a vertex v of degree 2 that is not in the marker path of G_1. We claim that v has also degree 2 in G. If $v \in X_1 \setminus (A_1 \cup B_1)$, then it is clear, so suppose v is in $A_1 \cup B_1$, say in A_1 up to symmetry. Note that (X_1, X_2) is consistent by Lemma 2.8. Since v has degree 2 in G_1, condition (vii) in the definition of consistent 2-joins applied to v implies that v has precisely one neighbor in $X_1 \setminus A_1$ and one neighbor in the marker path of G_1. Since by Lemma 2.11 $|A_1| \geq 2$, it follows that $G[A_1]$ is disconnected. Hence, by condition (iv) in the definition of consistent 2-joins, $|A_2| = 1$. It follows that v has the same degree in G_1 and in G. □

In [14] it is shown that one can decompose a graph with no star cutset using a sequence of 'non-crossing' 2-joins into graphs with no star cutset and no 2-join (which will in our case be basic). This will be particularly important when using 2-join decomposition to solve the stable set problem. We now describe such 2-join decomposition obtained in [14].

A *flat path* of G is any path of G of length at least 3, whose interior vertices are of degree 2, and whose ends do not have a common neighbor. When $\mathcal{M}$ is a collection of vertex-disjoint flat paths of G, a 2-join (X_1, X_2) of G is $\mathcal{M}$-independent if for every path P from $\mathcal{M}$ we have that either $V(P) \subseteq X_1$ or $V(P) \subseteq X_2$.

2-Join decomposition tree T_G of depth $p \geq 1$ of a graph G that has no star cutset and has a 2-join

(i) The root of T_G is $(G^0, \mathcal{M}^0)$, where $G^0 := G$ and $\mathcal{M}^0 = \emptyset$.

(ii) Each vertex of T_G is a pair $(H, \mathcal{M})$, where H is a graph of $\mathcal{D}$ and $\mathcal{M}$ is a set of disjoint flat paths of H.

The non-leaf vertices of T_G are pairs $(G^0, \mathcal{M}^0), \ldots, (G^{p-1}, \mathcal{M}^{p-1})$. Each non-leaf vertex $(G^i, \mathcal{M}^i)$ has two children. One is $(G^{i+1}, \mathcal{M}^{i+1})$, the other one is $(G_B^{i+1}, \mathcal{M}_B^{i+1})$.

The leaf-vertices of T_G are the pairs $(G_B^1, \mathcal{M}_B^1), \ldots, (G_B^p, \mathcal{M}_B^p)$ and $(G^p, \mathcal{M}^p)$. Graphs $G_B^1, G_B^2, \ldots, G_B^p, G^p$ have no star cutset nor 2-join.

(iii) For $i \in \{0, 1, \ldots, p-1\}$, G^i has a 2-join (X_1^i, X_2^i) that is extreme with extreme side X_1^i and that is $\mathcal{M}^i$-independent. Graphs G^{i+1} and G_B^{i+1} are blocks of decomposition of G^i w.r.t. (X_1^i, X_2^i) whose marker paths are of length at least 3. The block G_B^{i+1} corresponds to the extreme side X_1^i, i.e. $X_1^i \subseteq V(G_B^{i+1})$.

Set $\mathcal{M}_B^{i+1}$ consists of paths from $\mathcal{M}^i$ whose vertices are in X_1^i. Note that the marker path used to construct the block G_B^{i+1} does not belong to $\mathcal{M}_B^{i+1}$.

Set $\mathcal{M}^{i+1}$ consists of paths from $\mathcal{M}^i$ whose vertices are in X_2^i together with the marker path P^{i+1} used to build G^{i+1}.

(iv) $\mathcal{M}_B^1 \cup \ldots \cup \mathcal{M}_B^p \cup \mathcal{M}^p$ is the set of all marker paths used in the construction of the vertices $G^1, \ldots, G^p$ of T_G, and the sets $\mathcal{M}_B^1, \ldots, \mathcal{M}_B^p, \mathcal{M}^p$ are pairwise disjoint.

Vertex $(G^p, \mathcal{M}^p)$ is a leaf of T_G and is called the *deepest vertex of* T_G.

The 2-join decomposition tree is described slightly differently in [14], but the following result follows easily from the proofs in [14].

Lemma 2.13 ([14]) *There is an algorithm with the following specification.*

Input: *A graph G that has no star cutset and has a 2-join.*

Output: *A 2-join decomposition tree T_G of depth at most n.*

Running time: $\mathcal{O}(n^4 m)$.

Lemma 2.14 *If $G \in \mathcal{D}$ has a 2-join, then T_G can be constructed, and all graphs $G_B^1, G_B^2, \ldots, G_B^p, G^p$ that correspond to the leaves of T_G are in $\mathcal{D}_{\text{BASIC}}$.*

PROOF — By Lemma 2.7 G has no star cutset, and hence we can construct T_G. By Lemma 2.10 all graphs that correspond to vertices of T_G belong to $\mathcal{D}$. By construction graphs $G_B^1, G_B^2, \ldots, G_B^p, G^p$ have no star cutset nor 2-join. By Lemma 2.7 it follows that none of them has a clique cutset, and hence by Theorem 2.4 all of them are basic. □

3 Maximal cliques and clique coloring

A vertex v of a graph G is *simplicial* if $N(v)$ is a clique, and it is *bisimplicial* if $N(v)$ is a disjoint union of two cliques that are anticomplete to each other. Note that every simplicial vertex is also bisimplicial. We now show that every graph $G \in \mathcal{C}$ has a bisimpicial vertex, which we then use to obtain an algorithm for finding a maximum weight clique of G and to prove that G is 3-clique-colorable.

Theorem 3.1 *If $G \in \mathcal{C}$ then for every clique K of G, either $K = V(G)$ or there is a bisimplicial vertex (of G) in $G \setminus K$.*

PROOF — The proof is by induction on $|V(G)|$.

If R is a triangle-free chordless graph, then $L(R)$ does not contain a claw nor a diamond, and hence every vertex of $L(R)$ is bisimplicial. If G is a P-graph, then by Lemma 2.1 it contains at least two branches of length at least 2. The clique K contains internal vertices of at most one of these branches. Hence, $G \setminus K$ contains a vertex of degree 2, that is therefore bisimplicial. So, when G is basic the result holds.

Let us now suppose that (A, K', B) is a split of a clique cutset K' of G, and let G_A and G_B be the blocks of decomposition w.r.t. this clique cutset. Then clique K is contained in G_A or in G_B. W.l.o.g. suppose that K is contained in G_B. By induction, there is a bisimplicial vertex in $G_A \setminus K'$, and hence in $G \setminus K$.

So, let us suppose that G is not basic and that it does not admit a clique cutset. By Theorem 2.4, G admits a 2-join $(X_1', X_2', A_1', A_2', B_1', B_2')$. Then K is contained in $G[X_1' \cup A_2']$ or in $G[X_2' \cup B_1']$. W.l.o.g. suppose that K is contained in $G[X_2' \cup B_1']$. Let $(X_1, X_2, A_1, A_2, B_1, B_2)$ be a split of a minimally-sided 2-join of G with $X_1 \subseteq X_1'$ being a minimal side, and let G_1 and G_2 be the corresponding blocks of decomposition. By Lemma 2.7, G does not have a star cutset. So by Lemma 2.11 (ii) G_1 does not have a 2-join. By Lemma 2.10, $G_1 \in \mathcal{D}$, and so by Theorem 2.4, G_1 is basic. Additionally, by Lemma 2.11 (i), $|A_1|, |B_1| \geq 2$, and hence, by (iv) and (v) of definition of consistent 2-join, A_2 and B_2 are cliques. Also we may assume that $K \cap A_1 = \emptyset$, since otherwise for $u \in K \cap A_1$ and $v \in K \cap B_1$, $u, v \in B_1'$, and hence any $b \in B_2'$ is a vertex of X_2' that has a neighbor in both A_1 and B_1, contradicting the assumption that (X_1, X_2) is a 2-join of G such that $X_1 \subseteq X_1'$. It follows that $K \subseteq X_2 \cup B_1$. If G_1 is the line graph of a triangle-free chordless graph, then every vertex of A_1 is bisimplicial in G_1 and hence bisimplicial in G (since A_2 is a clique). So, let us assume that G_1 is a P-graph. By Lemma 2.12, a vertex u of X_1 is of degree 2 in G. If B_1 is a clique then, since $|B_1| \geq 2$ and by (viii) of definition of consistent 2-join, it follows that $u \notin B_1$, and therefore $u \notin K$ and the result holds (since A_2 is a clique). If B_1 is not a clique, then b_2 is a center of a claw of G_1, and hence it is contained in the special clique K' of P-graph G_1 (where b_2 is the vertex of the marker path of G_1 that is complete to B_1). So $K' \subseteq B_1 \cup \{b_2\}$. Now, since every vertex of $X_1 \setminus K'$ is bisimplicial in G_1, every vertex of $X_1 \setminus B_1$ is bisimplicial in G (since A_2 is a clique). □

Maximum weight clique

Let G be a graph and $w : V(G) \to [0, +\infty)$ a *weight function on* G. A *maximum weight clique* of G is a clique K of G, such that $\sum_{v \in K} w(v)$ has the maximum value. If K is a maximum weight clique of G, we denote by $\omega_w(G)$ the value of the sum $\sum_{v \in K} w(v)$.

Theorem 3.2 *There is an algorithm with the following specifications:*

Input: *A weighted graph $G \in \mathcal{C}$.*

Output: *A maximum weight clique of G.*

Running time: $\mathcal{O}(n^2 m)$.

PROOF — By Theorem 3.1, G contains a vertex v that is bisimplicial. This vertex can be found in time $\mathcal{O}(nm)$. Let $N(v)$ consist of (possibly empty) cliques K_1 and K_2. Then

$$\omega_w(G) = \max\{\omega_w(G \setminus v), \omega_w(\{v\} \cup K_1), \omega_w(\{v\} \cup K_2)\},$$

and if K is the maximum weight clique of $G \setminus v$, then a maximum weight clique of G is K, $\{v\} \cup K_1$ or $\{v\} \cup K_2$. So it is enough to find a maximum weight clique of $G \setminus v$, which can be done by applying (recursively) the same procedure on $G \setminus v$.

The total running time of this algorithm is $\mathcal{O}(n \cdot nm) = \mathcal{O}(n^2 m)$. □

Clique coloring

A *k-clique-coloring* of a graph G is a function $c : V(G) \to \{1, 2, \ldots, k\}$, such that for every inclusion-wise maximal clique K of size at least 2, $c(K) = \{c(v) : v \in K\}$ has at least 2 elements. We say that G is *k-clique-colorable* if it admits a k-clique-coloring. The *clique-chromatic number* of G, denoted by $\chi_C(G)$, is the smallest number k such that G is k-clique-colorable.

There are graphs in $\mathcal{C}$ that are not 2-clique-colorable, as shown in Figure 1. We now prove that every $G \in \mathcal{C}$ is 3-clique-colorable.

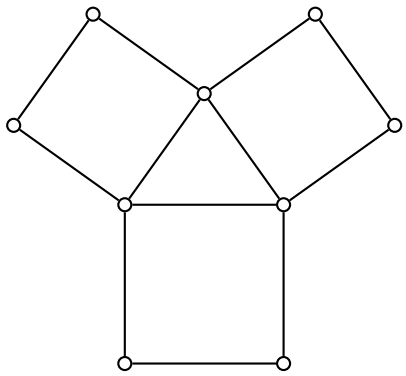

Figure 1: Graph from $\mathcal{C}$ that is not 2-clique-colorable

Theorem 3.3 *If $G \in \mathcal{C}$, then $\chi_C(G) \leq 3$.*

PROOF — The proof is by induction on $|V(G)|$. By Theorem 3.1, G contains a vertex v that is bisimplicial. By induction, we can 3-clique-color $G \setminus v$. Let K_1 and K_2 be disjoint, anticomplete cliques such that $K_1 \cup K_2 = N(v)$. To obtain a 3-clique-coloring of G from the 3-clique-coloring of $G \setminus v$ it is enough to color v with a color different from a vertex of K_1 and a vertex of K_2 (note that if K_i is empty, for some $i \in \{1, 2\}$, then any of the three colors satisfies the property). □

4 Stable set problem

Let G be a graph and $w : V(G) \to [0, +\infty)$ a *weight function on G*. A *maximum weight stable set* of G is a stable set S of G, such that $\sum_{v \in S} w(v)$ has the maximum value. If S is

a maximum weight stable set of G, we denote by $\alpha_w(G)$ the value of the sum $\sum_{v\in S} w(v)$.

In this section we give a polynomial-time algorithm for finding a maximum weight stable set of a weighted graph in $\mathcal{C}$. To do this we first introduce a different way to decompose w.r.t. a 2-join, one that is suited for the stable set problem.

A *gem* Γ is the graph defined with $V(\Gamma) = \{p_1, p_2, p_3, p_4, z\}$ and $E(\Gamma) = \{p_1p_2, p_2p_3, p_3p_4, p_1z, p_2z, p_3, p_4z\}$. Vertex z is the *center* of the gem Γ. Let $(X_1, X_2, A_1, A_2, B_1, B_2)$ be a split of a 2-join of $G \in \mathcal{C}$ and $C_i = X_i \setminus (A_i \cup B_i)$, for $i \in \{1, 2\}$. To build a *gem-block* G_2^g replace X_1 by an induced path $pxyq$ plus a vertex z complete to this path, such that p (resp. q) is complete to A_2 (resp. B_2) and these are the only edges between $\{p, x, y, q, z\}$ and X_2. Note that G_2^g is not necessarily in $\mathcal{C}$. Let $a := \alpha_w(G[A_1 \cup C_1])$, $b := \alpha_w(G[B_1 \cup C_1])$, $c := \alpha_w(G[C_1])$ and $d := \alpha_w(G[X_1])$. We give the following weights to the new vertices of G_2^g: $w(p) = a$, $w(x) = a + b - d$, $w(y) = d$, $w(q) = 2d - a$ and $w(z) = c + d$.

Lemma 4.1 ([14]) *If G_2^g is the gem-block of G, then the weights of G_2^g are non-negative and $\alpha_w(G_2^g) = \alpha_w(G) + d$.*

The gem blocks are useful for computing α, but they are not preserving for our class $\mathcal{C}$, so we cannot recursively decompose using gem blocks. Instead, we will first construct the 2-join decomposition tree T_G (as in Section 2) using marker paths of length 3, and then we will reprocess it by replacing marker paths by gems. As a consequence, the leaves of our decomposition tree may fail to be basic. So, we define *extensions* of basic graphs in the following way.

Let P be a flat path of G of length 3. *Extending* P means adding a new vertex z that is complete to $V(P)$ and anticomplete to the rest of the graph. An *extension* of a pair $(G, \mathcal{M})$, where G is a graph and $\mathcal{M}$ a set of vertex-disjoint flat paths of G of length 3, is any weighted graph obtained by extending the flat paths of $\mathcal{M}$ and giving any non-negative weights to all the vertices. An *extension* of G is any graph that is an extension of $(G, \mathcal{M})$ for some $\mathcal{M}$. We define $\mathcal{D}^{\text{EXT}}_{\text{BASIC}}$ to be the class of all graphs that are an extension of a graph from $\mathcal{D}_{\text{BASIC}}$.

Let us examine the graphs in $\mathcal{D}^{\text{EXT}}_{\text{BASIC}}$. If G is a line graph of a triangle-free chordless graph, then an extension G' of G is again a line graph (but not of a triangle-free chordless graph). Indeed, if R is the root graph of G, then a flat path of G of length 3 correspond to a path $B = b_1b_2b_3b_4b_5$ of R all of whose interior vertices are of degree 2. Hence, $G' = L(R')$, where R' is the graph obtained from R by adding edge b_2b_4 for every such B. Similarly, if G is a P-graph with special clique K and skeleton R, then each flat path $P = a_1a_2a_3a_4$ of G either corresponds to a path $B = b_1b_2b_3b_4b_5$ of R that belongs to a branch or a limb of R, or an endnode of P, say a_4 belongs to K (in the latter case let $C = c_1c_2c_3c_4$ be the subpath of a limb of R such that $L(R[V(C)])$ is the path $a_1a_2a_3$ in G). To obtain R' from R, for each B we add the edge b_2b_4, and for each C we add the edge c_2c_4. Then an extension G' of G is obtained from $L(R')$ by adding clique K and edges between them so

that for every $v \in K$, $N_{G'}(v) = N_G(v) \cup Z_v$, where Z_v is the set of all centers of gems that were used to extend flat paths with endnode v.

Lemma 4.2 *There is an algorithm with the following specifications:*

Input: *A weighted graph $G' \in \mathcal{D}^{\text{EXT}}_{\text{BASIC}}$.*

Output: *A maximum weight stable set of G'.*

Running time: *$\mathcal{O}(n^4)$.*

PROOF — Let G' be an extension of $G \in \mathcal{D}_{\text{BASIC}}$. In order to compute a maximum weight stable set of G', we first need to compute G and then decide if G a line graph of a triangle-free chordless graph or a P-graph. Since G is diamond-free, every diamond of G' is contained in some gem of G'. So, to obtain G from G' it is enough to find all gems contained in G'. This can be done in time $\mathcal{O}(n^4)$. To decide whether G is a line graph of a triangle-free chordless graph or a P-graph it is enough to test whether or not G contains a claw (the line graph of a triangle-free chordless graph does not contain a claw, and a P-graph does). In case G is a P-graph, we find its special clique K by finding all centers of claws and extend them to a maximal clique (in case there is only one center of claw, say u, then we check whether u is contained in a clique of size 3, and if it is we extend that clique to a maximal clique, and otherwise $K = \{u\}$). All this can also be done in time $\mathcal{O}(n^4)$.

If G is the line graph of a triangle-free chordless graph, then G' is also a line graph, so the maximum weighted stable set of G' can be computed in time $\mathcal{O}(n^3)$ using Edmonds' algorithm [4].

If G is a P-graph, then $G' \setminus K$ is a line graph $L(R')$, and hence maximum weight stable set of G' is either contained in $L(R')$, or has exactly one vertex of K. So, it is enough to compute a maximum weight stable set of $L(R')$, and a maximum weight stable set of $G' \setminus N[v]$, for each $v \in K$. Since, for each $v \in K$ the graph $G' \setminus N[v]$ is a line graph, we conclude that using Edmonds' algorithm a maximum weight stable set of G' can be computed in time $\mathcal{O}(n^4 + n \cdot n^3) = \mathcal{O}(n^4)$. □

Lemma 4.3 *There is an algorithm with the following specifications:*

Input: *A weighted graph $G \in \mathcal{D}$.*

Output: *A maximum weight stable set of G.*

Running time: *$\mathcal{O}(n^4 m)$.*

PROOF — Check whether G contains a 2-join (this can be done in time $\mathcal{O}(n^2 m)$ by the algorithm in [1]). If it does not, then by Theorem 2.4 $G \in \mathcal{D}_{\text{BASIC}} \subseteq \mathcal{D}^{\text{EXT}}_{\text{BASIC}}$, and hence we compute maximum weight stable set in $\mathcal{O}(n^4)$ time by Lemma 4.2.

Otherwise, we construct the 2-join decomposition tree T_G (of depth $1 \leq p \leq n$) using marker paths of length 3 in $\mathcal{O}(n^4m)$ time by Lemma 2.13. By Lemma 2.14 all graphs $G^1_B, \ldots, G^p_B, G^p$ that correspond to the leaves of T_G are in $\mathcal{D}_{\text{BASIC}}$. We now reprocess T_G.

Let P^1 be the marker path used in the construction of G^1. We replace G^1 by the corresponding gem block G^{1g}. To do this we need to compute the weights a^1, b^1, c^1, d^1 that need to be assigned to the vertices of the gem, and this amounts to computing four weighted stable set problems on G^1_B. Since $G^1_B \in \mathcal{D}_{\text{BASIC}} \subseteq \mathcal{D}^{\text{EXT}}_{\text{BASIC}}$ this can be done in $\mathcal{O}(n^4)$ time by Lemma 4.2. By Lemma 4.1 $\alpha_w(G^{1g}) = \alpha_w(G) + d^1$. In all the other graphs that correspond to the vertices of T_G and contain P^1, we extend P^1 using weights a^1, b^1, c^1, d^1. We continue this process for $i = 2, \ldots, p$. So if P^i is the marker path used in construction of G^i, we compute the weights needed to transform it into a gem block, by computing four weighted stable set problems on G^i_B whose paths in $\mathcal{M}^i_B$ have all already been extended. Since this graph is in $\mathcal{D}^{\text{EXT}}_{\text{BASIC}}$, this can be done in $\mathcal{O}(n^4)$ time by Lemma 4.2. In all the graphs that correspond to vertices of T_G that contain P^i we extend P^i using calculated weights a^i, b^i, c^i, d^i.

The last graph we reprocess is G^p, and let us denote by G^{pg} the graph that we obtain at the end of the reprocessing procedure. By repeated application of Lemma 4.1, $\alpha_w(G^{pg}) = \alpha_w(G) + d^1 + \ldots + d^p$, and so we deduce $\alpha_w(G)$. The proof of Lemma 4.1 actually shows how to keep track of a stable set of G whose weight is $\alpha_w(G)$. Since $p \leq n$, this algorithm can be implemented to run in time $\mathcal{O}(n^4m + n \cdot n^4) = \mathcal{O}(n^4m)$. □

Theorem 4.4 *There is an algorithm with the following specifications:*

Input: *A weighted graph $G \in \mathcal{C}$.*

Output: *A maximum weight stable set of G.*

Running time: $\mathcal{O}(n^6m)$.

PROOF — By Theorem 2.5 we construct the clique cutset decomposition tree T of G in $\mathcal{O}(nm)$ time. So all the leaves of T are graphs from $\mathcal{D}$. By using Tarjan's method from [13] to compute a maximum weight stable set of G it is enough to compute $\mathcal{O}(n^2)$ maximum weight stable sets on the leaves of T (each one of which can be computed in $\mathcal{O}(n^4m)$ time by Lemma 4.3). Therefore, this algorithm can be implemented to run in time $\mathcal{O}(nm + n^2 \cdot n^4m) = \mathcal{O}(n^6m)$. □

5 Vertex coloring

A *k-coloring* of a graph G is a function $c : V(G) \to \{1, 2, \ldots, k\}$, such that for every $uv \in E(G)$, $c(u) \neq c(v)$. We say that G is *k-colorable* if it admits a k-coloring. The

chromatic number of G, denoted by $\chi(G)$, is the smallest number k such that G is k-colorable. In this section we give a polynomial-time coloring algorithm for $\mathcal{C}$ and prove that every $G \in \mathcal{C}$ is $\max\{3, \omega(G)\}$-colorable.

A graph is *sparse* if every edge is incident with at least one vertex of degree at most 2. Note that every sparse graph is chordless. A proper *2-cutset* of a connected graph G is a pair of non-adjacent vertices a, b such that there is a partition $(X, Y, \{a, b\})$ of $V(G)$ with X and Y anticomplete, both $G[X \cup \{a, b\}]$ and $G[Y \cup \{a, b\}]$ contain a path from a to b and neither $G[X \cup \{a, b\}]$ nor $G[Y \cup \{a, b\}]$ is a chordless path. We say that $(X, Y, \{a, b\})$ is a *split* of this proper 2-cutset. The *blocks of decomposition* G_X and G_Y w.r.t. this cutset are defined as follows. Block G_X (resp. G_Y) is the graph obtained by taking $G[X \cup \{a, b\}]$ (resp. $G[Y \cup \{a, b\}]$) and adding a new vertex u (resp. v) complete to $\{a, b\}$ (and anticomplete to the rest).

A decomposition theorem for the class of chordless graphs is proved in [7]. An improvement of this theorem, that is an *extreme decomposition* for this class, is proved in [9]. We give both results in the following theorem.

Theorem 5.1 ([7, 9]) *If G is a 2-connected chordless graph, then either G is sparse or G admits a proper 2-cutset. Additionally, if $(X, Y, \{a, b\})$ is a split of a proper 2-cutset of G such that $|X|$ is minimum among all such splits, then a and b both have at least two neighbors in X, and G_X is sparse.*

A *k-edge-coloring* of a graph G is a function $c : E(G) \to \{1, 2, \ldots, k\}$, such that for every two distinct edges with a common vertex, say uv and uw, $c(uv) \neq c(uw)$. G is *k-edge-colorable* if it admits a k-edge-coloring. The *edge-chromatic number* of G is the smallest number k such that G is k-edge-colorable.

The edge-coloring of chordless graphs is studied in [9], where the authors obtained the following result. For a graph G, let $\Delta(G) = \max\{\deg(v) : v \in V(G)\}$ and $\delta(G) = \min\{\deg(v) : v \in V(G)\}$.

Theorem 5.2 ([9]) *Every chordless graph G is $\{3, \Delta(G)\}$-edge-colorable. Moreover, there is an $\mathcal{O}(n^3 m)$-time algorithm that finds such an edge-coloring.*

In this section we will prove a variant of the previous theorem (see Lemma 5.5). The following result is an important step towards obtaining a $\max\{3, \omega(G)\}$-coloring for our basic classes.

Lemma 5.3 ([9]) *Let $G = (V, E)$ be a sparse graph and suppose that a list L_{uv} of colors is associated with each edge $uv \in E$. Let S be a stable set of G that contains all vertices of G of degree at least 3. Suppose that for every vertex $u \in S$, all edges incident to u receive the same list. If for each edge $uv \in E$, $|L_{uv}| \geq \max\{\deg(u), \deg(v)\}$ and for each edge $uv \in E$ with no end in S, $|L_{uv}| \geq 3$, then there is an edge-coloring c of G such that, for each edge $uv \in E$, $c(uv) \in L_{uv}$. Furthermore, there is an $\mathcal{O}(nm)$-time algorithm that finds such an edge-coloring c.*

Let $v_1, \ldots, v_k$, where $1 \leq k \leq 3$, be some vertices of a branch B of G, such that they do not induce a path of length 2. Furthermore, let the list of colors L_i, $|L_i| \geq 2$, be associated with v_i, for $1 \leq i \leq k$, such that if v_i and v_j are adjacent, for some $1 \leq i < j \leq k$, then $L_i \cap L_j \neq \emptyset$. Note that branch B can be edge-colored with $\left|\bigcup_{i=1}^{k} L_i\right|$ colors, so that every edge incident with v_i is colored with a color from L_i, for $1 \leq i \leq k$. Indeed, if no two of the vertices from the set $\{v_1, \ldots, v_k\}$ are adjacent, then we can color B greedily (starting from one endnode of B). If w.l.o.g. v_1 and v_2 are adjacent, then we can obtain the desired edge-coloring by first coloring the edge v_1v_2 (with a color from $L_1 \cap L_2$) and then greedily coloring the rest of the branch (starting from the other edge incident with v_1 and the other edge incident with v_2). We say that such an edge-coloring of the branch B is *according to the lists* $L_1, \ldots, L_k$.

A vertex v of G is *free* if it is of degree 2 and both of its neighbors are also of degree 2. Vertices u and v of G are *parallel* if they are of degree 2, and contained in distinct parallel branches of G. A *ring* of G is a hole of G that has at most one vertex that is of degree at least 3 in G. A *small theta* of G is an induced subgraph of G isomorphic to $K_{2,3}$ ($K_{2,3}$ is complete bipartite graph whose sides have sizes 2 and 3). Note that if H is a small theta of a sparse graph G, then only degree 3 vertices of H can have neighbors in $G \setminus H$.

A set $S = \{v_1, \ldots, v_k\}$, $1 \leq k \leq 3$, of vertices is *good* if the following hold:

(i) at most one of $v_1, \ldots, v_k$ is of degree 2 and not free;

(ii) if $k = 3$, then S is not contained in a ring of G of length 5;

(iii) if $k = 3$ and some $v_i \in S$ is of degree 2 and not free, then the two vertices from $S \setminus \{v_i\}$ are not adjacent.

Lemma 5.4 *Let G be a triangle-free sparse graph with $\delta(G) = 2$, and let $v_1, \ldots, v_k \in V(G)$, $1 \leq k \leq 3$, be such that $\{v_1, \ldots, v_k\}$ does not induce a path of length 2. To vertices v_i, for $1 \leq i \leq k$, the lists of colors $L_i \subseteq \{1, \ldots, s\}$, where $s = \max\{3, \Delta(G)\}$, are associated so that $|L_i| \geq \deg(v_i)$. Furthermore, if v_i and v_j are adjacent, for some $1 \leq i < j \leq k$, then $L_i \cap L_j \neq \emptyset$. If one of the following holds:*

(1) $k = 3$, $\{v_1, \ldots, v_k\}$ is contained in a ring of length 5 and $|L_1 \cup L_2 \cup L_3| \geq 3$;

(2) $k \leq 3$ and the set $\{v_1, \ldots, v_k\}$ is good;

(3) $k = 2$, and if v_1 and v_2 are both of degree 2, then $\{v_1, v_2\}$ is not contained in a small theta of G;

then there is an s-edge-coloring of G, such that every edge incident with v_i is colored with a color from L_i, for $1 \leq i \leq k$. Furthermore, there is an $\mathcal{O}(nm)$-time algorithm that finds such an edge-coloring.

PROOF — We prove the result for each of the cases (1), (2) and (3) separately.

(1) Let $B = u_1u_2u_3u_4u_5u_1$ be the ring of G that contains $\{v_1, \ldots, v_k\}$, and w.l.o.g. $v_1 = u_1$, $v_2 = u_2$ and $v_3 = u_4$. Furthermore, let v be vertex of B with maximum degree. As a first step, we s-edge-color B.

First, assume that $|L_1 \cup L_2| \geq 3$. Then, we color u_1u_2 with a color $c \in L_1 \cap L_2$, then color edges u_1u_5 and u_2u_3 with distinct colors from $L_1 \setminus \{c\}$ and $L_2 \setminus \{c\}$, respectively, and finally color edges u_3u_4 and u_1u_5 with distinct colors from L_3. So, w.l.o.g. let $L_1 = L_2 = \{1, 2\}$. Then we may assume $3 \in L_3$, and let $c \in L_3 \setminus \{3\}$. Now, we color the edges u_1u_5 and u_2u_3 with a color $c' \in \{1, 2\} \setminus \{c\}$, u_1u_2 with the color from $\{1, 2\} \setminus \{c'\}$, u_3u_4 in 3 and u_4u_5 in c.

So, we have obtained an s-edge-coloring of B. Let G' be the graph obtained from G by removing all vertices of B except v. Now, to complete the s-edge-coloring of G we s-edge-color G' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors (in this edge-coloring of G') such that all edges incident with v (in G) have different colors.

(2) We prove the claim by induction on $|V(G)|$. If $k = 1$, then to obtain an s-edge-coloring of G we first s-edge-color G by Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors such that edges incident with v_1 have colors from the list L_1. So, we may assume that $k \geq 2$. Also, by induction, we may assume that G is connected. We now consider the following cases.

Case 1. $\Delta(G) = 2$.

If there is an edge e of G that is not incident with v_1, v_2 nor v_3, then to obtain an edge coloring of G we first edge-color the path $G \setminus \{e\}$ according to the lists L_1, L_2 and L_3 and then color the edge e. So, we may assume that every edge of G is incident with at least one of v_1, v_2 or v_3.

Let $k = 2$. Then G is of length 4 and vertices v_1 and v_2 are not adjacent. So, to obtain an edge coloring of G we first color edges incident with v_1 (with colors from L_1) and then edges incident with v_2 (with colors from L_2).

Let $k = 3$. Since G is not a hole of length 5, we have that $G = u_1u_2u_3u_4u_5u_6u_1$, and no two vertices from $\{v_1, v_2, v_3\}$ are adjacent. So, we may assume w.l.o.g. that $v_1 = u_1$, $v_2 = u_3$ and $v_3 = u_5$. If there is a color $c \in L_1 \cap L_2 \cap L_3$, then to obtain an edge-coloring of G we first color edges u_1u_2, u_3u_4 and u_5u_6 with c, and then color the remaining edges according to the lists L_1, L_2 and L_3. So, let us assume that $L_1 \cap L_2 \cap L_3 = \emptyset$. Then to obtain an edge-coloring of G we first greedily edge-color the path $u_6u_1u_2u_3$ according to the lists L_1 and L_2. Note that either the colors of u_6u_1 and u_2u_3 are distinct, or u_6u_1 and u_2u_3 are colored with the same color which is not in L_3. In both cases we can color the remaining edges of G according to the list L_3.

Case 2. v_1 is contained in a ring.

By Case 1 we may assume that $\Delta(G) \geq 3$. Let B be the ring of G that contains v_1, let v be the vertex of B of degree at least 3 and let G' be the graph obtained from G by deleting vertices of $B \setminus v$ (and edges incident with these vertices). Note that G' is triangle-free sparse and that v is of degree at least 3, of degree 1 or is free in G'. Furthermore, if v is of degree 1 in G', then let $P = v \dots v'$ be the limb of G' that contains v; otherwise $P = \{v\}$ and $v' = v$. Also, let $V'' = (V(G) \setminus (V(B) \cup V(P))) \cup \{v'\}$ and $G'' = G[V'']$.

In this case we will assume that $k = 3$, that is, if $k = 2$, then we take v_3 to be an arbitrary vertex such that $\{v_1, v_2, v_3\}$ satisfies conditions of this lemma, and $L_3 = \{1, 2, \dots, s\}$ (such v_3 exists: if $v_2 \notin B$, then we may define v_3 to be v or a free vertex of B; if $v_2 \in B$, then we may define v_3 to be a vertex from $V'' \setminus \{v\}$ of degree at least 3 or free). It suffices to consider the following cases.

Case 2.1. $\{v_1, v_2, v_3\} \subseteq V(B)$.

To obtain an s-edge-coloring of G we first edge-color B according to the lists $L_1, \dots, L_k$ (as in Case 1). Then, we s-edge-color G' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \dots, s\}$), and then permute the colors (in this edge-coloring of G') such that all edges incident with v (in G) have different colors.

Case 2.2. $v_2 \in B$ and $v_3 \notin B$.

W.l.o.g. $v_1 \neq v$. First assume that $P = \{v\}$. If v_3 is not adjacent to v, then to obtain a desired edge-coloring of G we first edge-color B according to the lists L_1 and L_2 (as in Case 1). Let L be the set of colors used for coloring edges incident with v in this coloring. Then, to complete s-edge-coloring of G we, by induction, edge-color G' such that the lists L' and L_3 are associated with vertices v and v_3, where $L' = \{1, 2, \dots, s\} \setminus L$ if $v_2 \neq v$, or $L' = L_2 \setminus L$ if $v = v_2$. So, let us assume that v_3 is adjacent to v. Then v_1 and v_2 are not adjacent and not adjacent to v, and they are free or $v_2 = v$. If $v = v_2$, then to obtain a desired edge-coloring of G we first, by induction, edge-color G' such that the lists L_2 and L_3 are associated with vertices v_2 and v_3, and then edge-color B (as in Case 1) such that the lists L_1 and $\{1, 2, \dots, s\} \setminus L'_2$ are associated with v_1 and v_2, where L'_2 is the set of colors used for coloring edges incident with v in edge-coloring of G'. Hence, suppose that $v \neq v_2$. Let $c \in L_3$. Then to obtain a desired edge-coloring of G we first edge-color B such that the lists L_1, L_2 and $\{1, 2, \dots, s\} \setminus \{c\}$, are associated with v_1, v_2 and v. Then, to complete s-edge-coloring of G we, by induction, edge-color G' such that the lists $\{1, 2, \dots, s\} \setminus L$ and L_3 are associated with vertices v and v_3, where L is the set of colors used for coloring edges incident with v in this edge-coloring of B.

Next, assume that $P \neq \{v\}$. If $v_3 \notin V(P)$, then we first edge-color B such that the lists L_1 and L_2 are associated with v_1 and v_2 (as in Case 1). Then we edge-color P such that the edge incident with v is colored with a color not used for coloring edges incident with v in this edge-coloring of B. Finally, we edge color G'' by induction, such that the lists $L'' = \{1, 2, \dots, s\} \setminus \{c\}$ and L_3 are associated with v' and v_3, where c is the color used for coloring edge of P incident with v' (note that $L'' \cap L_3 \neq \emptyset$, since $|L''| = s - 1$). So, suppose that $v_3 \in V(P)$. If v_3 is not adjacent to v, then we first edge-color B such

that the lists L_1 and L_2 are associated with v_1 and v_2 (as in Case 1). Then we greedily edge-color P from v to v', such that the edge incident with v is colored with a color not used for coloring edges incident with v in this edge-coloring of B, and that edges incident with v_3 are colored with colors from L_3. To complete edge-coloring of G, we edge color G'' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors (in this edge-coloring of G''), such that all edges incident with v' (in G) have different colors. Finally, suppose that v and v_3 are adjacent. Then v_3 is of degree 2 and not free, and so v_1 and v_2 are not adjacent, v_1 is free and v_2 is either free or $v_2 = v$. In particular, no vertex of $\{v_1, v_2\}$ is adjacent to v. If $L_2 \cap L_3 \neq \emptyset$ then let $c \in L_2 \cap L_3$, and otherwise let c be any color from L_3. Then we first edge-color B (as in Case 1), such that: if $v \neq v_2$, then lists L_1, L_2 and $\{1, 2, \ldots, s\} \setminus \{c\}$ are associated v_1, v_2 and v; if $v = v_2$, then lists L_1 and $L_2 \setminus \{c\}$ are associated v_1 and v_2. Then we greedily edge-color P such that vv_3 is colored with c and the other edge from P incident with v_3 with a color from $L_3 \setminus \{c\}$. Finally, we edge-color G'' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors (in this edge-coloring of G'') such that all edges incident with v' (in G) have different colors.

Case 2.3. $v_2, v_3 \notin V(B)$.

First, assume that $P = \{v\}$ (i.e. $v = v'$). If $v_1 = v$, then to obtain an s-edge-coloring of G, we first s-edge-color G' by induction, such that the lists L_1, L_2 and L_3 are associated with v_1, v_2 and v_3 (note that $|L_1| \geq 4$, and hence $|L_1 \cup L_2 \cup L_3| \geq 4$). Finally, we edge-color B and then permute the colors in this edge-coloring of B such that all edges incident with v receive different color. So, suppose $v \neq v_1$. If v_1 is not adjacent to v, then we first s-edge-color G' by induction, such that the lists L_2 and L_3 are associated with v_2 and v_3, and then edge-color B (as in Case 1) such that the lists L_1 and $\{1, 2, \ldots, s\} \setminus L$ are associated with v_1 and v, where L is the set of colors used for coloring edges incident with v in this edge-coloring of G'. Finally, suppose that v_1 is adjacent to v. Then v_1 is of degree 2 and not free, so v_2 and v_3 are either of degree at least 3 or free, and hence $\{v_2, v_3\}$ is anticomplete to v. Let c be a color from L_1, and $L' = \{1, 2, \ldots, s\} \setminus \{c\}$. Hence, to obtain an s-edge-coloring of G, we first edge-color G' by induction, such that the lists L', L_2 and L_3 are associated with v, v_2 and v_3 (note that $|L'| = s - 1 \geq 3$, and hence $|L' \cup L_2 \cup L_3| \geq 3$), and then edge-color B (as in Case 1) such that the lists L_1 and $\{1, 2, \ldots, s\} \setminus L''$ are associated with v_1 and v, where L'' is the set of colors used for coloring edges incident with v in this edge-coloring of G'.

Now, suppose that $v \neq v'$. If $v_2, v_3 \in V''$, then we first, by induction, s-edge-color G'' such that the lists L_2 and L_3 are associated with v_2 and v_3. Then we edge-color P greedily from v' to v (note that v and v' are not adjacent), such that the edge incident with v' is colored with a color not used for coloring edges incident with v' in this edge-coloring of G'', and that the edge incident with v is colored with a color from L_1. Let this color be c. To complete edge-coloring of G, we edge-color B such that: if $v_1 = v$, then the list $L_1 \setminus \{c\}$ is associated with v_1; if $v_1 \neq v$, then the lists L_1 and $L = \{1, 2, \ldots, s\} \setminus \{c\}$ are associated

with v_1 and v (note that $L \cap L_1 \neq \emptyset$, since $|L| = s-1$). Next, suppose that $v_2, v_3 \in V(P)$. In this case, we first edge-color P such that the lists L_2, L_3 and possibly L_1 (if $v_1 = v$) are associated with v_2, v_3 and possibly v_1 (if $v_1 = v$). Let c be the color of the edge incident with v in this edge-coloring of P. Then we edge-color B such that: if $v_1 = v$, then the list $L_1 \setminus \{c\}$ is associated with v_1; if $v_1 \neq v$, then the lists L_1 and $L = \{1, 2, \ldots, s\} \setminus \{c\}$ are associated with v_1 and v (note that $L \cap L_1 \neq \emptyset$, since $|L| = s-1$). To complete edge-coloring of G, we s-edge-color G'' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors (in this edge-coloring of G'') such that all edges incident with v' (in G) have different colors. Finally, we may assume that $v_2 \in V(P) \setminus \{v'\}$ and $v_3 \in V'' \setminus \{v'\}$. To obtain s-edge-coloring of G we first s-edge-color P such that the lists L_2 and possibly L_1 (if $v_1 = v$) are associated with v_2 and possibly v_1 (if $v_1 = v$). Let c (resp. c') be the color of the edge incident with v (resp. v') in this edge-coloring of P. Next, by induction, we s-edge-color G'' such that the lists L_3 and $L' = \{1, 2, \ldots, s\} \setminus \{c'\}$ are associated with v_3 and v' (note that $L_3 \cap L' \neq \emptyset$, since $|L'| = s-1$). To complete s-edge-coloring of G we s-edge-color B such that: if $v_1 = v$, then the list $L_1 \setminus \{c\}$ is associated with v_1; if $v_1 \neq v$, then the lists L_1 and $L = \{1, 2, \ldots, s\} \setminus \{c\}$ are associated with v_1 and v (note that $L \cap L_1 \neq \emptyset$, since $|L| = s-1$).

By Case 2, from now on we may assume that no vertex from $\{v_1, \ldots, v_k\}$ is contained in a ring, and by Case 1 we may assume that $\Delta(G) \geq 3$. In particular, since every vertex of G is contained in a ring or a branch, every vertex of $\{v_1, \ldots, v_k\}$ is contained in a branch of G.

Case 3. v_1 is free.

Let B be the branch of G that contains v_1 with endnodes u_1 and u_2, and let G' be the graph obtained from G by deleting internal vertices of B (and edges incident with these vertices). Note that since G is sparse, vertices u_1 and u_2 are free or of degree at least 3 in G', and every neighbor of u_1 and u_2 is of degree 2 in G and G'. In particular, for every $v \in V(G') \setminus \{u_1, u_2\}$, if $\{u_1, u_2, v\}$ is not contained in a ring of length 5 of G', then the set $\{u_1, u_2, v\}$ is good in G'.

Case 3.1. Neither v_2 nor v_3 is adjacent to both u_1 and u_2.

First, let us assume that $V(B) \cap \{v_2, \ldots, v_k\} \neq \emptyset$. If $\{v_1, \ldots, v_k\} \subseteq V(B)$, then to obtain an s-edge-coloring of G we first s-edge-color B according to the lists $L_1, \ldots, L_k$, and then, by induction, s-edge-color G' with the list L'_i associated with u_i, for $i \in \{1, 2\}$. The list L'_i, for $i \in \{1, 2\}$, is defined as follows: $L'_i = \{1, 2, \ldots, s\} \setminus \{c_{u_i}\}$ if $u_i \notin \{v_2, \ldots, v_k\}$ and $L'_i = L_j \setminus \{c_{u_i}\}$ if $u_i = v_j$, for some $2 \leq j \leq k$, where c_{u_1} (resp. c_{u_2}) is the color of the edge incident with u_1 (resp. u_2) in this edge-coloring of B.

Let us now assume that w.l.o.g. $v_2 \in B$, but $v_3 \notin B$ (in this case $k = 3$), and w.l.o.g. let v_2 be in the u_1v_1-subpath of B. If v_2 and v_3 are not adjacent (i.e. $v_2 \neq u_1$, or $v_2 = u_1$ and v_3 is not adjacent to u_1), then to obtain desired s-edge-coloring of G we first s-edge-color B according to the lists L_1 and L_2, and such that the edges incident with u_1 and u_2 receive

different colors (this can be done since the edge incident with u_2 is the last that we color, and we have at least 2 options for coloring it). Then, by induction, we s-edge-color G' such that the lists L'_1, L'_2 and L_3 are associated with vertices u_1, u_2 and v_3 (L'_1 and L'_2 are defined as in the previous part of the proof, and they satisfy $|L'_1 \cup L'_2| \geq 3$, since $L'_1 \neq L'_2$). So, let us assume that v_3 is adjacent to u_1 and $v_2 = u_1$. Let $c \in L_2 \cap L_3$. To obtain desired s-edge-coloring of G, we first greedily s-edge-coloring B starting with the edge incident with u_1 and giving it a color $c' \in L_2 \setminus \{c\}$, and such that the color of the edge incident with u_2 is not c'. Then, by induction, we s-edge-color G' such that the lists $L''_1 = L_2 \setminus \{c'\}$, $L''_2 = \{1, 2, \ldots, s\} \setminus \{c_{u_2}\}$ and L_3 are associated with vertices u_1, u_2 and v_3, where c_{u_2} is the color of the edge incident with u_2 in the edge-coloring of B (note that $|L''_1 \cup L''_2| \geq 3$, since $L''_1 \neq L''_2$).

Finally, let us assume that v_2 and v_3 are not in B. Observe that $\{u_1, v_1, v_2, u_2\}$ cannot induce a path, since otherwise both v_2 and v_3 would be of degree 2 and not free. Therefore, by the case we are in, at most one of the sets $\{v_2, v_3, u_1\}$ and $\{v_2, v_3, u_2\}$ induces a path of length 2. W.l.o.g. assume that $\{v_2, v_3, u_1\}$ does not induce a path of length 2. Furthermore, if u_1 is adjacent to v_2 or v_3, then that vertex is not free or degree at least 3 in G. Also, by the case we are in, $\{v_2, v_3, u_1\}$ cannot be contained in a ring of G' of length 5. Hence the set $\{v_2, v_3, u_1\}$ is good in G'. Now, to obtain desired s-edge-coloring of G we first, by induction, s-edge-color G' such that the lists $\widetilde{L}_1$, L_2 and possibly L_3 (if $k = 3$) are associated with vertices u_1, v_2 and possibly v_3 (if $k = 3$), where $\widetilde{L}_1 = \{1, 2, \ldots, s\} \setminus \{c'_1\}$ ($c'_1 \notin L_1$ if $|L_1| = 2$, or arbitrary otherwise). Then branch B is greedily 3-edge-colored in the following way: we color the edge incident with u_1 with color c'_1, the edge incident with u_2 with a color not used for coloring edges incident with u_2 in G', then color v_1u_2-subpath of B (greedily from u_2 to v_1) and finally color u_1v_1-subpath of B (greedily from v_1 to u_1).

Case 3.2. v_2 is adjacent to both u_1 and u_2.

In this case, v_2 is of degree 2 and not free in G. Also, if $k = 3$, then v_3 must be free or of degree at least 3 in G, and it follows that v_3 is anticomplete to $\{u_1, u_2\}$ and so is free or of degree at least 3 in G'.

First, assume that $k = 2$ or $v_3 \notin B$. Note that in this case, if $k = 3$, then $\{u_1, v_2, v_3\}$ is not contained in a ring of G' of length 5. So, to obtain desired s-edge-coloring of G we do the following. First, by induction, we s-edge-color G' so that the lists $\widetilde{L}_1$, L_2 and possibly L_3 (if $k = 3$) are associated with vertices u_1, v_2 and possibly v_3 (if $k = 3$), where $\widetilde{L}_1 = \{1, 2, \ldots, s\} \setminus \{c'_1\}$ ($c'_1 \notin L_1$ if $|L_1| = 2$, or arbitrary otherwise). Then branch B is greedily 3-edge-colored in the following way: we color the edge incident with u_1 with color c'_1, the edge incident with u_2 with a color not used for coloring edges incident with u_2 in G', then color v_1u_2-subpath of B (greedily from u_2 to v_1) and finally color u_1v_1-subpath of B (greedily from v_1 to u_1).

Next let us assume that v_3 is in B and free. Then v_1 and v_3 are not adjacent, and let us w.l.o.g. assume that v_3 is in the v_1u_2-subpath of B. Then to obtain desired s-edge-coloring of G we first, by induction, s-edge-color G' such that the lists $\widetilde{L}_1$ and L_2 are associated

with vertices u_1 and v_2, where $\widetilde{L}_1 = \{1, 2, \ldots, s\} \setminus \{c'_1\}$ ($c'_1 \notin L_1$ if $|L_1| = 2$, or arbitrary otherwise). Then branch B is greedily 3-edge-colored in the following way: we color the edge incident with u_1 with color c'_1, the edge incident with u_2 with a color not used for coloring edges incident with u_2 in G', then color v_1u_2-subpath of B (greedily from u_2 to v_1) and finally color u_1v_1-subpath of B (greedily from v_1 to u_1).

So, w.l.o.g. let $v_3 = u_1$. Then to obtain desired s-edge-coloring of G we first, by induction, s-edge-color $G'' = G \setminus \{v_2\}$, such that the lists L_1, L''_3 and L'' are associated with vertices v_1, v_3 and u_2, where $L''_3 = L_3 \setminus \{c'\}$ for some $c' \in L_2 \cap L_3$, and $L'' = \{1, 2, \ldots, s\} \setminus \{c''\}$ for some $c'' \in L_2 \setminus \{c'\}$ (note that $\{v_1, v_3, u_2\}$ is not contained in ring of G'' of length 5). Finally, we color the edge u_1v_2 in c' and u_2v_2 in c''.

By Case 3, from now on we may assume that no vertex from $\{v_1, \ldots, v_k\}$ is free. Therefore it suffices to consider the following cases.

Case 4. v_1 and possibly v_3 (if $k = 3$) are of degree at least 3.

If v_2 is also of degree at least 3, then the proof follows from Lemma 5.3 (with S the set of all vertices of degree at least 3, lists L_i, for $i \in \{1, \ldots, k\}$, given to edges incident with v_i, and list $\{1, \ldots, s\}$ given to all other edges). So, suppose that $\deg(v_2) = 2$. Let $B = u_1 \ldots u_2$ be the branch of G that contains v_2, and G' be the graph obtained from G by deleting internal vertices of B (and edges incident with these vertices). Note that since G is sparse $\{v_1, v_3\}$ is anticomplete to $\{u_1, u_2\}$ and each of the vertices u_1 and u_2 is free or of degree at least 3 in G'.

First, let us assume that $\Delta(G) = 4$. If $v_1, v_3 \notin B$, then to obtain s-edge-coloring of G, we first s-edge-color B according to L_2. Let $L' = \{1, 2, \ldots, \Delta(G)\} \setminus \{c_{u_1}\}$ and $L'' = \{1, 2, \ldots, \Delta(G)\} \setminus \{c_{u_2}\}$, where c_{u_1} (resp. c_{u_2}) is the color of the edge incident with u_1 (resp. u_2) in this edge-coloring of B. Finally, we s-edge-color G' using Lemma 5.3, such that the lists L_1, L', L'' and possibly L_3 (if $k = 3$) are associated with edges incident with v_1, u_1, u_2 and possibly v_3 (if $k = 3$), respectively, and the list $\{1, 2, \ldots, \Delta(G)\}$ associated with all other edges. If w.l.o.g. $v_1 = u_1$, then to obtain s-edge-coloring of G, we first s-edge-color B according to the lists L_1, L_2 and possibly L_3 (if $v_3 = u_2$). Next, we associate with v_1 the list $L'_1 = L_1 \setminus \{c_{u_1}\}$, and to u_2 the list $L'' = \{1, 2, \ldots, \Delta(G)\} \setminus \{c_{u_2}\}$ if $v_3 \neq u_2$, or $L'' = L_3 \setminus \{c_{u_2}\}$ if $v_3 = u_2$, where c_{u_1} (resp. c_{u_2}) is the color of the edge incident with u_1 (resp. u_2) in this edge-coloring of B. Then we s-edge-color G', by induction, such that the lists L'_1, L'' and possibly L_3 (if $k = 3$ and $v_3 \neq u_2$) are associated with u_1, u_2 and possibly v_3 (if $k = 3$ and $v_3 \neq u_2$).

Finally, let $\Delta(G) = 3$. In this case $L_1 = L_3 = \{1, 2, 3\}$, and hence any 3-edge-coloring of G respects the lists L_1 and L_3. So, to obtain a desired s-edge-coloring of G we first s-edge-color G using Lemma 5.3 (we give the list $\{1, 2, 3\}$ to all edges) and then permute the colors such that the edges incident with v_2 receive colors from the list L_2.

(3) We prove the claim by induction on $|V(G)|$. By induction, we may assume that G is connected. It suffices to consider the following cases.

Case 1. v_1 and v_2 are of degree at least 3.

The proof in this case follows from Lemma 5.3 (with S the set of all vertices of degree at least 3, lists L_i, for $i \in \{1,2\}$, given to edges incident with v_i, and list $\{1,2,\ldots,s\}$ given to all other edges).

Case 2. v_1 is of degree 2.

If $\Delta(G) = 2$ then G is a hole and it is easy to see how to obtain the desired coloring. So we may assume that $\Delta(G) \geq 3$. We now consider the following cases.

Case 2.1. v_1 is contained in a ring of G.

Let B be that ring, let v be the vertex of degree at least 3 of B, and let G' be the graph obtained from G by deleting degree 2 vertices of B (and edges incident with these vertices). Note that G' is triangle-free sparse and that v is of degree at least 3, of degree 1 or is free in G'. Also, since G is sparse, v is not adjacent to a vertex of degree at least 3. In particular, if v is contained in a small theta of G (or any of its induced subgraphs), then v is not a degree 2 vertex of this small theta. Finally, if v is of degree 1 in G', then let $P = v \ldots v'$ be the limb of G' that contains v; otherwise $P = \{v\}$ and $v = v'$. Also, let $V'' = (V(G) \setminus (V(B) \cup V(P)) \cup \{v'\}$ and $G'' = G[V'']$.

If $\{v_1, v_2\} \in V(B)$, then we proceed as in Case 2.1 of part (2). So, let us assume that $v_2 \notin V(B)$. Our proof in this case is similar to the proof of Case 2.3 of part (2).

First, let $P = \{v\}$ (i.e. $v = v'$). If $v_1 = v$, then to obtain an s-edge-coloring of G, we first s-edge-color G' by induction such that the lists L_1 and L_2 are associated with v_1 and v_2. To complete edge-coloring of G, we edge-color B and then permute the colors in this edge-coloring of B such that all edges incident with v receive different color. So, suppose $v \neq v_1$. If v_1 is not adjacent to v, then we first s-edge-color G' (using part (2)) such that the list L_2 is associated with v_2, and then edge-color B (as in Case 1 of (2)) such that the lists L_1 and $\{1,2,\ldots,s\} \setminus L$ are associated with v_1 and v, where L is the set of colors used for coloring edges incident with v in this edge-coloring of G'. So, suppose that v_1 is adjacent to v. Then we first s-edge-coloring G' by induction such that the lists L_2 and $L' = \{1,2,\ldots,s\} \setminus \{c\}$ are associated with v_2 and v, where $c \in L_1$ is arbitrary (note that $L_2 \cap L' \neq \emptyset$, since $|L'| = s - 1$). To complete edge-coloring of G we edge-color B (as in Case 1 of (2)) such that the lists L_1 and $\widetilde{L} = \{1,2,\ldots,s\} \setminus L''$ are associated with v_1 and v, where L'' is the list of colors used for coloring edges incident with v in this edge-coloring of G' (note that $c \in L_1 \cap \widetilde{L}$).

Suppose now that $v \neq v'$. If $v_2 \in V''$, then we first s-edge-color G' (using part (2)) such that the list L_2 is associated with v_2 . Then we edge-color P greedily from v' to v (note that v and v' are not adjacent), such that the edge incident with v' is colored with a color not used for coloring edges incident with v' in this edge-coloring of G'', and that the edge incident with v is color with a color from L_1. Let this color be c. To complete edge-coloring of G, we edge color B such that: if $v_1 = v$, then the list $L_1 \setminus \{c\}$ is associated with v_1; if $v_1 \neq v$, then the lists L_1 and $L = \{1,2,\ldots,s\} \setminus \{c\}$ are associated with v_1

and v (note that $L \cap L_1 \neq \emptyset$, since $|L| = s - 1$). Next, suppose that $v_2 \in V(P)$. In this case, we first edge-color P such that the lists L_2 and possibly L_1 (if $v_1 = v$) are associated with v_2 and possibly v_1 (if $v_1 = v$). Let c be the color of the edge incident with v in this edge-coloring of P. Then we edge-color B such that: if $v_1 = v$, then the list $L_1 \setminus \{c\}$ is associated with v_1; if $v_1 \neq v$, then the lists L_1 and $L = \{1, 2, \ldots, s\} \setminus \{c\}$ are associated with v_1 and v (note that $L \cap L_1 \neq \emptyset$, since $|L| = s - 1$). To complete edge-coloring of G, we s-edge-color G'' using Lemma 5.3 (such that all edges receive the list $\{1, 2, \ldots, s\}$), and then permute the colors (in this edge-coloring of G'') such that all edges incident with v' (in G) have different colors.

By Case 2.1, from now on we may assume that neither v_1 nor v_2 is contained in a ring of G. Let $B = u_1 \ldots u_2$ be the branch of G that contains v_1, and let G' be the graph obtained from G by deleting internal vertices of B (and edges incident with these vertices). Since G is sparse vertices u_1 and u_2 are free or of degree at least 3 in G', and every neighbor of u_1 and u_2 is of degree 2 in G and G'. In particular, if u_1 (resp. u_2) is contained in a small theta of G (or any of its induced subgraphs), then u_1 (resp. u_2) is not a degree 2 vertex of this small theta.

Case 2.2. v_1 and v_2 are not parallel.

In this case $\{u_1, u_2, v_2\}$ is not contained in a ring of G' of length 5.

If $v_2 \in B$, then to obtain s-edge-coloring of G we first s-edge-color B according to the lists L_1 and L_2, and then by induction s-edge-color G' such that the list L' and L'' are associated with u_1 and u_2. Lists are defined as follows: $L' = \{1, 2, \ldots, s\} \setminus \{c_{u_1}\}$ (resp. $L'' = \{1, 2, \ldots, s\} \setminus \{c_{u_2}\}$) if $u_1 \neq v_2$ (resp. $u_2 \neq v_2$) or $L' = L_2 \setminus \{c_{u_1}\}$ (resp. $L'' = L_2 \setminus \{c_{u_2}\}$) if $u_1 = v_2$ (resp. $u_2 = v_2$), where c_{u_1} (resp. c_{u_2}) is the color of the edge incident with u_1 (resp. u_2) in the edge-coloring of B.

So, let $v_2 \notin B$. Since $\{u_1, u_2, v_2\}$ does not induce a path of length 2 (by the case we are in),and since the set is good in G', to obtain s-edge-coloring of G we first s-edge-color B according to the list L_1, and then by part (2) s-edge-color G' with the lists $L' = \{1, 2, \ldots, s\} \setminus \{c_{u_1}\}$, $L'' = \{1, 2, \ldots, s\} \setminus \{c_{u_2}\}$ and L_2 associated with u_1, u_2 and v_2, where c_{u_1} (resp. c_{u_2}) is the color of the edge incident with u_1 (resp. u_2) in the edge-coloring of B (note that $L_2 \cap L' \neq \emptyset$ and $L_2 \cap L'' \neq \emptyset$).

Case 2.3. v_1 and v_2 are parallel.

If v_1 or v_2 is free, then we can apply part (2). So, suppose that both v_1 and v_2 are not free. Let B' be the branch of G that contains v_2.

Case 2.3.1. At least one of the branches B and B' is of length at least 3.

W.l.o.g. let B' be of length at least 3 and v_2 adjacent to u_1. Now we define colors c_1 and c_2 that are going to be used when edge-coloring G:

- if v_1 is adjacent to both u_1 and u_2, then c_1 and c_2 are distinct colors from L_1;
- if v_1 is adjacent to u_1, but not adjacent to u_2, then c_1 is a color from L_1 and c_2 a color not from $\{c, c_1\}$, where c is a color from L_1 distinct from c_1;

- if v_1 is adjacent to u_2, but not adjacent to u_1, then c_2 is a color from L_1 and c_1 a color not from $\{c, c_2\}$, where c is a color from L_1 distinct from c_2.

Now, we first, by part (2), s-edge-color G' such that the lists $L' = \{1, 2, \ldots, s\} \setminus \{c_1\}$, $L'' = \{1, 2, \ldots, s\} \setminus \{c_2\}$ and L_2 are associated with u_1, u_2 and v_2 (note that $|L' \cup L'' \cup L_2| \geq 3$, since $L' \cup L'' = \{1, 2, \ldots, s\}$). To complete the edge-coloring of G we color the branch B in the following way: we first color the edges incident with u_1 and u_2 with colors c_1 and c_2, respectively, and then greedily edge-color the rest of B starting from v_1 and according to the list L_1.

Case 2.3.2. Branches B and B' are of length 2.

First, let us assume that both u_1 and u_2 are of degree at least 4 in G, and let $G'' = G \setminus \{v_1, v_2\}$. Then G'' is triangle-free sparse and vertices u_1 and u_2 are free or of degree at least 3 in G''. Now we define colors c_1, c_2, c_3, c_4 that are going to be used when edge-coloring G:

- if $|L_1 \cap L_2| \geq 2$ and $c', c'' \in L_1 \cap L_2$, then $c_1 = c_4 = c'$, $c_2 = c_3 = c''$;
- if $|L_1 \cap L_2| = 1$ and $L_1 \cap L_2 = \{c\}$, then $c_1 = c_4 = c$, $c_2 = c'$ and $c_3 = c''$, where $c' \in L_2 \setminus \{c\}$ and $c'' \in L_1 \setminus \{c\}$;
- if $L_1 \cap L_2 = \emptyset$, then $c_1, c_3 \in L_1$, $c_1 \neq c_3$, and $c_2, c_4 \in L_2$, $c_2 \neq c_4$.

Now, by induction, we edge-color G'' such that the lists $\{1, 2, \ldots, s\} \setminus \{c_1, c_2\}$ and $\{1, 2, \ldots, s\} \setminus \{c_3, c_4\}$ are associated with u_1 and u_2, and then color edges u_1v_1, u_1v_2, u_2v_1 and u_2v_2 in colors c_1, c_2, c_3 and c_4, respectively.

So, we may assume that w.l.o.g. $\deg_G(u_1) = 3$. Let $\widetilde{G}$ be the graph obtained from $G \setminus \{v_1, v_2\}$ by adding the edge u_1u_2. Since u_1 is of degree 2 in $\widetilde{G}$, graph $\widetilde{G}$ is sparse, and since $\{v_1, v_2\}$ is not contained in a small theta of G graph $\widetilde{G}$ is triangle-free. Furthermore, since at least one neighbor of u_1 in $\widetilde{G}$ is of degree 2, u_1 is not a degree 2 vertex of some small theta of $\widetilde{G}$.

We define lists of colors $\widetilde{L}_1$ and $\widetilde{L}_2$ that are going to be used for obtaining an edge-coloring of $\widetilde{G}$ (and G):

(i) if $|L_1 \cap L_2| \geq 2$ and $c_1, c_3 \in L_1 \cap L_2$, then $\widetilde{L}_1 = \{c_1, c_2\}$ and $\widetilde{L}_2 = \{1, 2, \ldots, s\} \setminus \{c_3\}$, where $c_2 \notin \{c_1, c_3\}$;

(ii) if $|L_1 \cap L_2| = 1$ and $L_1 \cap L_2 = \{c_1\}$, then $\widetilde{L}_1 = \{c_1, c_2\}$ and $\widetilde{L}_2 = \{1, 2, \ldots, s\} \setminus \{c_2\}$, where $c_2 \in L_1 \setminus \{c_1\}$;

(iii) if $L_1 \cap L_2 = \emptyset$, then $\widetilde{L}_1 = \{c_1, c_2\}$ and $\widetilde{L}_2 = \{1, 2, \ldots, s\} \setminus \{c_2\}$, where $c_1 \in L_1$ and $c_2 \in L_2$.

Now, by induction, we s-edge-color $\widetilde{G}$ such that the lists $\widetilde{L}_1$ and $\widetilde{L}_2$ are associated with u_1 and u_2. Furthermore, we can permute the colors c_1 and c_2 in case (i), such that the edge

u_1u_2 is colored with c_1. Finally, to obtain an edge-coloring of G we extend the obtained edge-coloring of $\widetilde{G} \setminus \{u_1u_2\}$ in the following way. In case (i) we color the edges u_1v_1, u_1v_2, u_2v_1 and u_2v_2 with colors c_1, c_3, c_3 and c_1, respectively; in case (ii) we color the edges u_1v_1, u_1v_2, u_2v_1 and u_2v_2 with colors c_1, c_3, c_2 and c_1, respectively, where $c_3 \in L_2 \setminus \{c_1\}$; in case (iii) we color the edges u_1v_1, u_1v_2, u_2v_1 and u_2v_2 with colors c_3, c_4, c_1 and c_2, respectively, where $c_3 \in L_1 \setminus \{c_1\}$ and $c_4 \in L_2 \setminus \{c_2\}$.

Note that this proof yields an $\mathcal{O}(nm)$-time algorithm that finds described edge-coloring. Indeed, all steps in the proof can be done in linear time, except when Lemma 5.3 is applied (which takes $\mathcal{O}(nm)$), but then the edge-coloring of G can be completed in linear time. □

Lemma 5.5 *Let G be a triangle-free chordless graph, and let v_1 and v_2 be distinct vertices of $V(G)$ both of degree at least 1. Let $\widetilde{G}$ be a graph obtained from G by adding a path $Q = q_1 \dots q_k$, $k \geq 2$, (whose vertices are disjoint from vertices of G) and edges q_1v_1 and q_kv_2 (these are the only edges between G and Q). Assume that $\widetilde{G}$ is also triangle-free chordless. Suppose that we are given two lists of colors $L_1, L_2 \subseteq \{1, 2, \dots, s\}$, where $s = \max\{3, \Delta(G)\}$, such that $|L_1| \geq \deg_G(v_1)$, $|L_2| \geq \deg_G(v_2)$, and if v_1 and v_2 are adjacent, then $L_1 \cap L_2 \neq \emptyset$. Also, suppose that if both v_1 and v_2 are of degree 1 in G and $|L_1| = |L_2| = 1$, then their neighbors in G are distinct. Then there exists an s-edge-coloring of G such that every edge of G incident with v_i is colored with a color from L_i, for $i \in \{1, 2\}$. Furthermore, such an edge coloring can be obtained in $\mathcal{O}(n^3m)$-time.*

PROOF — We prove this lemma by induction on $|V(G)|$. By induction, Theorem 5.2 and Lemma 5.4, we may assume that G is connected.

Case 1. G contains a vertex v of degree 1.

First, suppose that G is a path, i.e. $P = v \dots v'$. If $\{v_1, v_2\} \cap \{v, v'\} = \emptyset$, then we s-edge-color this path according to the lists L_1 and L_2. If $|\{v_1, v_2\} \cap \{v, v'\}| = 1$ and w.l.o.g. $v = v_1$, then we first color the edge incident with v (with a color from L_1 if $v_1v_2 \notin E(G)$, or a color from $L_1 \cap L_2$ is $v_1v_2 \in E(G)$) and then greedily s-edge-color the rest of G (starting from v) according to the list L_2. If w.l.o.g. $v_1 = v$ and $v_2 = v'$, we first color edges incident with v_1 and v_2 (with colors from L_1 and L_2 if $v_1v_2 \notin E(G)$, or a color from $L_1 \cap L_2$ if $v_1v_2 \in E(G)$), and then greedily edge-color the rest of G.

So, suppose that G is not a path. Let $B = v \dots v'$ be the limb of G that contains v and let G' the graph induced by $(V(G) \setminus V(B)) \cup \{v'\}$. If $\{v_1, v_2\} \subseteq V(B)$, then we first s-edge-color B in the following way: if B is not of length 2 or $\{v_1, v_2\} \neq \{v, v'\}$, then we s-edge-color B as in the previous paragraph; if B is of length 2 and w.l.o.g $v_1 = v$ and $v_2 = v'$, then we color the edge incident with v with a color $c \in L_1$ and then color the edge incident with v' with a color from $L_2 \setminus \{c\}$ (note that $|L_2| \geq 3$). To complete edge-coloring of G we s-edge-color G' using Theorem 5.2 and permute the colors in this edge-coloring of G' so that the edges incident with v' (in G) all receive different colors.

If $\{v_1, v_2\} \subseteq V(G')$, then we first, by induction, s-edge-color G' so that the lists L_1 and L_2 are associated with v_1 and v_2 (note that v' is of degree at least 2 in G', so a vertex is of degree 1 in G' iff it is of degree 1 in G). Then, we greedily s-edge-color B (starting from v') such that edges incident with v' all receive different colors.

Finally, suppose w.l.o.g. that $v_1 \in V(B) \setminus \{v'\}$ and $v_2 \in V(G') \setminus \{v'\}$. If both v_1 and v_2 are of degree 1 and adjacent to v', then we first color edges incident with v_1 and v_2 (with colors from L_1 and L_2), then s-edge-color $G \setminus \{v_1, v_2\}$ using Theorem 5.2 and finally permute the colors in this edge-coloring of $G \setminus \{v_1, v_2\}$ such that edges incident with v' all receive different colors. So, suppose that w.l.o.g. v_2 is of degree at least 2 or not adjacent to v'. Then, to obtain an s-edge-coloring of G, we first greedily s-edge-color B such that edges incident with v_1 receive colors from the list L_1. Let $L' = \{1, 2, \ldots, s\} \setminus \{c\}$, where c is the color of the edge incident with v' in this edge-coloring of B. Also, let Q'' be the path induced by $V(Q)$ and vertices of the v_1v'-subpath of B, and let Q' be the path induced by $V(Q'') \setminus \{v'\}$. Then Q' is disjoint from G', its endnodes are adjacent to v' and v_2, and the graph induced by $V(G') \cup V(Q')$ is triangle-free chordless. Hence, to complete s-edge-coloring of G we, by induction, s-edge-color G' so that the lists L_2 and L' are associated with v_2 and v' (note that $\deg_{G'}(v') \geq 2$, and if $v'v_2 \in E(G)$, then $\deg_{G'}(v_2) \geq 2$ and hence $L' \cap L_2 \neq \emptyset$ since $|L'| = s - 1$).

By Case 1, we may assume that $\delta(G) \geq 2$. By Theorem 5.1, it is enough to consider the following cases.

Case 2. G is sparse.

Follows from part (3) of Lemma 5.4. Indeed, in this case v_1 and v_2 are not degree 2 vertices that belong to a small theta H of G, since otherwise $\widetilde{G}[V(H) \cup V(Q)]$ is a cycle with chords.

Case 3. G has a cutvertex.

Let v be a cutvertex of G and let $(X_1, X_2, \{v\})$ be a partition of $V(G)$ such that X_1 is anticomplete to X_2.

First, let us assume that $v_i \in X_i$, for $i \in \{1, 2\}$. Let Q'_i, for $i \in \{1, 2\}$, be a chordless path between v_{3-i} and v in G, and let Q_i be the path induced in $\widetilde{G}$ by $(V(Q) \cup V(Q'_i)) \setminus \{v\}$. Then Q_i is disjoint from $G[X_i \cup \{v\}]$, its endnodes are adjacent to v and v_i respectively, and the graph induced by $X_i \cup \{v\} \cup V(Q_i)$ is triangle-free chordless. So, if v_i is not adjacent to v, for some $i \in \{1, 2\}$, then we obtain an s-edge-coloring of G in the following way. We first s-edge-color $G[X_{3-i} \cup \{v\}]$ using Theorem 5.2 and then permute the colors in this edge-coloring so that edges incident with v_{3-i} receive colors from the list L_{3-i}. Let L' be the set of colors used for coloring edges incident with v in this coloring, and let $L = \{1, 2, \ldots, s\} \setminus L'$. Then, by induction, we s-edge-color $G[X_i \cup \{v\}]$ so that edges incident with v_i receive colors from the list L_i and edges (from $G[X_i \cup \{v_i\}]$) incident with v receive colors from the list L. Merging these colorings we obtain an s-coloring of G. So, let us assume that v_i is adjacent to v, for $i \in \{1, 2\}$. Let $c_i \in L_i$, for $i \in \{1, 2\}$, be distinct colors. To obtain an s-edge-coloring of G we first, by induction, s-edge-color $G[X_1 \cup \{v\}]$,

such that edges incident with v_1 receive colors from the list L_1 and edges incident with v (in $G[X_1 \cup \{v\}]$) colors from the list $\{1, 2, \ldots, s\} \setminus \{c_2\}$. Let L be the set of colors used for coloring edges incident with v in this coloring, and let $L' = \{1, 2, \ldots, s\} \setminus L$. Then, to obtain a desired edge-coloring of G we, by induction, s-edge-color $G[X_2 \cup \{v\}]$, so that edges incident with v_2 receive colors from the list L_2 and edges incident with v (in $G[X_2 \cup \{v\}]$) colors from the list L' (note that $c_2 \in L' \cap L_2$).

So, we may assume w.l.o.g. that $v_1, v_2 \in X_1 \cup \{v\}$. To obtain an s-edge-coloring of G, we first, by induction, s-edge-color $G[X_1 \cup \{v\}]$ so that the lists L_1 and L_2 are associated with v_1 and v_2. Then we s-edge-color $G[X_2 \cup \{v\}]$ using Theorem 5.2 and permute the colors in this edge-coloring so that edges incident with v (in G) all receive different colors.

Case 4. G has a proper 2-cutset $\{a, b\}$.

By Case 2 we may assume that G is 2-connected. Let a proper 2-cutset $\{a, b\}$ of G, with the split $(X_1, X_2, \{a, b\})$, be chosen so that the side X_1 is minimum among all such splits, that is, by Theorem 5.1, such that the block of decomposition G_{X_1} is sparse and a and b each have at least two neighbors in X_1. Then, $G[X_1 \cup \{a, b\}]$ is also triangle-free sparse, and since G_{X_1} is sparse, each of the vertices a and b is of degree at least 3 or free in $G[X_1 \cup \{a, b\}]$.

Case 4.1. $v_i \in X_i$, for $i \in \{1, 2\}$.

First, let us assume that v_1 is adjacent to both a and b. Since G_{X_1} is sparse, v_1 is of degree 2. By the minimality of X_1, this implies that $G[(X_1 \setminus \{v_1\}) \cup \{a, b\}]$ is a chordless path. Let Q' be a chordless path in G whose one endnode is v_2, the other is a vertex of $\{a, b\}$, and no interior vertex is in $\{a, b\}$. Then $V(Q) \cup V(Q') \cup X_1 \cup \{a, b\}$ induces in $\widetilde{G}$ a cycle with a chord (av_1 or bv_1), a contradiction. Therefore v_1 cannot be adjacent to both a and b. Similarly, $G[X_1 \cup \{a, b\}]$ is not a hole of length 5. Indeed, if we suppose the opposite, then v_1 is adjacent to a or b, w.l.o.g. to a. Now, if Q' is a path from v_2 to a in $G[X_2 \cup \{a, b\}]$ whose interior does not go through b, then $V(Q) \cup (Q') \cup X_1 \cup \{a, b\}$ induces a cycle with chord av_1, a contradiction. Finally, note that $\{a, b\}$ is not contained in a ring of length 5 of $G[X_1 \cup \{a, b\}]$. Indeed, if we suppose the opposite, then, since $G[X_1 \cup \{a, b\}]$ is not a hole of length 5, the degree at least 3 vertex of this ring is a cutvertex of G.

By the previous paragraph w.l.o.g. v_1 is not adjacent to b. Next, suppose that v_1a is also not an edge. Then, to obtain an s-edge-coloring of G we first s-edge-color $G[X_2 \cup \{a, b\}]$ using Theorem 5.2, and then permute the colors so that edges incident with v_2 receive colors from the list L_2. Let L'_a (resp. L'_b) be the set of colors used for coloring edges incident with a (resp. b) in this coloring, and let $L_a = \{1, 2, \ldots, s\} \setminus L'_a$ (resp. $L_b = \{1, 2, \ldots, s\} \setminus L'_b$). We complete s-edge-coloring of G using part (2) of Lemma 5.4, that is, we s-edge-color $G[X_1 \cup \{a, b\}]$ so that edges incident with v_1, a and b receive colors from the lists L_1, L_a and L_b.

Hence, we may assume that v_1 is adjacent to a (but not to b). Let $c_1 \in L_1$, and Q' be the path induced by $V(Q) \cup \{v_1\}$. Then Q' is disjoint from $G[X_2 \cup \{a, b\}]$, its endnodes are adjacent to a and v_2, and $\widetilde{G}[X_2 \cup \{a, b\} \cup V(Q')]$ is triangle-free chordless. So, to obtain

an s-edge-coloring of G we first, by induction, s-edge-color $G[X_2 \cup \{a,b\}]$, so that edges incident with v_2 receive colors from the list L_2 and edges incident with a (in $G[X_2 \cup \{a,b\}]$) colors from the list $\{1,2,\ldots,s\} \setminus \{c_1\}$. Let L'_a (resp. L'_b) be the set of colors used for coloring edges incident with a (resp. b) in this coloring, and let $L_a = \{1,2,\ldots,s\} \setminus L'_a$ (resp. $L_b = \{1,2,\ldots,s\} \setminus L'_b$). To complete s-edge-coloring of G we s-edge-color $G[X_1 \cup \{a,b\}]$ using part (2) of Lemma 5.4, so that edges incident with v_1, a and b receive colors from the lists L_1, L_a and L_b, respectively (note that $c_1 \in L_1 \cap L_a$).

Case 4.2. $v_1, v_2 \in X_i \cup \{a,b\}$, for some $i \in \{1,2\}$.

Note that if a and b are of degree 1 in $G[X_j \cup \{a,b\}]$, for some $j \in \{1,2\}$, then their neighbors in X_j are distinct. Indeed, if we suppose the opposite, then their common neighbor is a cutvertex of G.

Now, to obtain an s-edge-coloring of G we first, by induction, s-edge-color $G[X_i \cup \{a,b\}]$, so that edges incident with v_j, for $j \in \{1,2\}$, receive colors from the list L_j. Let L_a (resp. L_b) be the set of colors used for coloring edges incident with a (resp. b) in this coloring, and let $L'_a = \{1,2,\ldots,s\} \setminus L_a$ (resp. $L'_b = \{1,2,\ldots,s\} \setminus L_b$). Let Q'_{3-i} be a chordless path from a to b in $G[X_i \cup \{a,b\}]$, and Q_{3-i} be the path induced by $V(Q'_{3-i}) \setminus \{a,b\}$. Then Q_{3-i} is disjoint from $G[X_{3-i} \cup \{a,b\}]$, its endnodes are adjacent to a and b, and $G[X_{3-i} \cup \{a,b\} \cup V(Q_{3-i})]$ is triangle-free chordless. So, to complete s-edge-coloring of G we, by induction, s-edge-color $G[X_{3-i} \cup \{a,b\}]$, so that edges incident with a and b receive colors from the lists L'_a and L'_b.

Finally, let us explain how this proof yields an $\mathcal{O}(n^3m)$-time algorithm. By Case 1, in linear time we can reduce the problem to the one where $\delta(G) \geq 2$. By Lemma 5.4, Case 2 can be done in $\mathcal{O}(nm)$ time. In Case 3, either we use induction and apply Lemma 5.4, or we use Theorem 5.2 to color the entire side. In each step of Case 4 we first find a desired 2-cutset, which can be done in $\mathcal{O}(n^2m)$ time (see [9]), and then edge-color "the basic" side, which can be done in $\mathcal{O}(nm)$ time (by Lemma 5.4). Since there is at most $\mathcal{O}(n)$ steps and each time we use Theorem 5.2 the entire side is colored, the running time of the algorithm is $\mathcal{O}(n \cdot (n^2m + nm) + n^3m) = \mathcal{O}(n^3m)$, as claimed. □

Lemma 5.6 *Let $G \in \mathcal{D}$ and let $(X_1, X_2, A_1, A_2, B_1, B_2)$ be a split of a minimally-sided 2-join of G, with X_1 being its minimal side, and let G_1 and G_2 be the corresponding blocks of decomposition. Let $s = \max\{3, \omega(G)\}$, and assume that we are given an s-coloring c of $G[X_2]$. We can extend c to an s-coloring of G in $\mathcal{O}(n^3m)$-time. Furthermore, if G is a basic graph then it can be $\max\{3, \omega(G)\}$-colored in $\mathcal{O}(n^3m)$-time.*

PROOF — By Lemma 2.11, $|A_1|, |B_1| \geq 2$, and by Lemma 2.8, (X_1, X_2) is a consistent 2-join. Hence A_2 and B_2 are cliques. Also, by Lemma 2.10, $G_1 \in \mathcal{D}$, and by Lemma 2.11, G_1 does not have a 2-join. So, by Theorem 2.4, G_1 is basic.

Let L_a (resp. L_b) be the set of colors that c assigns to vertices of A_2 (resp. B_2). Let $L'_a = \{1,\ldots,s\} \setminus L_a$ and $L'_b = \{1,\ldots,s\} \setminus L_b$. We want an s-coloring of $G[X_1]$ in which the

vertices of A_1 are colored with colors from L'_a and vertices of B_1 are colored with colors from L'_b.

First suppose that G_1 is a line graph of a triangle-free chordless graph. Then G_1 is claw-free and hence (since $|A_1|, |B_1| \geq 2$) A_1 and B_1 are both cliques. Let R be the triangle-free chordless graph such that $L(R) = G[X_1]$. Since R is triangle-free, A_1 (resp. B_1) corresponds to the set of edges incident to vertex v_1 (resp. v_2) of R that is of degree at least 1 in R. Note that v_1 and v_2 are not adjacent since $A_1 \cap B_1 = \emptyset$. Since A_1, A_2, B_1, B_2 are all cliques, $\deg_R(v_1) \leq |L'_a|$ and $\deg_R(v_2) \leq |L'_b|$. We associate lists L'_a and L'_b to vertices v_1 and v_2 respectively, and the result follows from Lemma 5.5.

Now suppose that G_1 is a P-graph with special clique K and skeleton R. Let K' be the vertices of K that are centers of claws. Note that all centers of claws of G_1 are in K'. For $u \in K'$, by (viii) of the definition of the skeleton of a P-graph, all pendant vertices of $L(R)$ that are adjacent to u are of degree 2 in G_1. Let H be the graph obtained from $G[X_1]$ by removing degree 2 vertices of G_1 that are adjacent to a vertex of K'. Then H is claw-free, and hence by Lemma 2.2 and Lemma 2.3, H is the line graph of a triangle-free chordless graph, say R_H.

If A_1 and B_1 are both cliques, then (since $|A_1|, |B_1| \geq 2$) $A_1 \cup B_1 \subseteq V(H)$, and we s-color H, similarly to the case when G_1 was the line graph of triangle-free chordless graph, so that vertices of A_1 (resp. B_1) are colored with colors from L'_a (resp. L'_b). This coloring easily extends to an s-coloring of $G[X_1]$ since $s \geq 3$.

So we may assume that A_1 is not a clique. Since $G \in \mathcal{D}$, by Lemma 2.6, G is diamond-free, and hence (since $|A_1| \geq 2$) it follows that $|A_2| = 1$. Therefore $|L'_a| \geq 2$. Let a_2 be the vertex of the marker path of G_1 that is complete to A_1. Since A_1 is not a clique, a_2 is center of a claw and hence $a_2 \in K'$. It follows that $K \cap X_1 \subseteq A_1$, and so B_1 is a clique. Let $A'_1 = A_1 \cap V(H)$ and $A''_1 = A_1 \setminus A'_1$. So vertices of A''_1 are all of degree 2 in G_1, and hence of degree 1 in H. Also, $A'_1 \subseteq K$ (since the vertices that are adjacent to centers of claws of G_1, and in particular to a_2, must be either in K or of degree 2 in G_1), and hence A'_1 is a (possibly empty) clique.

We first s-color H so that the vertices of A'_1 (resp. B_1) receive the colors from L'_a (resp. L'_b), and then we extend this coloring to the desired coloring of $G[X_1]$. Clique B_1 of G_1 corresponds to edges incident to a vertex v_2 of R_H. We assign list L'_b to v_2. Since B_1 and B_2 are cliques, $\deg_{R_H}(v_2) \leq |L'_b|$. If $A'_1 = \emptyset$ then we s-color H by Theorem 5.2 (in $\mathcal{O}(n^3m)$-time) and then permute colors so that the vertices of B_1 are colored with colors from L'_b. So let us assume that $A'_1 \neq \emptyset$, and let v_1 be the vertex of R_H whose incident edges correspond to vertices of A'_1. We assign list L'_a to v_1. Since A_2 and A'_1 are cliques, $\deg_{R_H}(v_1) \leq |L'_a|$. Note that v_1 and v_2 are not adjacent in R_H since $A_1 \cap B_1 = \emptyset$. It now follows from Lemma 5.5 that we can obtain the desired s-coloring of H in $\mathcal{O}(n^3m)$-time. So, we may assume that we have an s-coloring of H in which vertices of A'_1 (resp. B_1) are colored with colors from L'_a (resp. L'_b). We now extend that to the desired s-coloring of $G[X_1]$. Since $s \geq 3$ and vertices of $X_1 \setminus H$ all have degree 2, we can greedily extend the coloring of H to vertices of $X_1 \setminus (H \cup A''_1)$. Since $|A_2| = 1$ and $s \geq 3$, it follows that

$|L'_a| \geq 2$. Since vertices of A''_1 are of degree 1 in H, we can clearly extend the coloring to them as well, so that they receive a color from L'_a.

Therefore, s-coloring of $G[X_2]$ can be extended to an s-coloring of G in $\mathcal{O}(n^3m)$-time. Observe that this proof also shows that any basic graph can be colored in $\mathcal{O}(n^3m)$-time.□

Theorem 5.7 *There is an algorithm with the following specifications:*

Input: *A graph $G \in \mathcal{C}$.*

Output: *A $\chi(G)$-coloring of G.*

Running time: $\mathcal{O}(n^5m)$.

Furthermore, if $G \in \mathcal{C}$ then $\chi(G) \leq \max\{3, \omega(G)\}$.

PROOF — We can decide in linear time if G is 2-colorable, and if it is 2-colorable we can 2-color it (also in linear time). So it is enough to give an algorithm that outputs a $\max\{3, \omega(G)\}$-coloring of G.

Claim: *Every $G \in \mathcal{D}$ can be $\max\{3, \omega(G)\}$-colored in $\mathcal{O}(n^4m)$-time.*

Proof of Claim: Let $G \in \mathcal{D}$ and let $s = \max\{3, \omega(G)\}$. We s-color G as follows. First check whether G contains a 2-join (this can be done in $\mathcal{O}(n^2m)$-time by the algorithm in [1]). If it does not, then by Theorem 2.4 G is basic, and hence it can be s-colored in $\mathcal{O}(n^3m)$-time by Lemma 5.6. Otherwise, by Lemma 2.13, we construct a 2-join decomposition tree T_G (of depths $1 \leq p \leq n$) using marker paths of length 3, in $\mathcal{O}(n^4m)$-time. By Lemma 2.14 all graphs $G^1_B, \ldots, G^p_B, G^p$ that correspond to the leaves of T_G are in $\mathcal{D}_{\text{BASIC}}$.

All the 2-joins used in the construction of T_G are extreme 2-joins. For our purpose here we want them to be minimally-sided 2-joins. Note that by Lemma 2.11, every minimally-sided 2-join is an extreme 2-join, but not every extreme 2-join is a minimally-sided one. The way T_G is constructed in [14] first a minimally-sided 2-join is found and then in order to achieve $\mathcal{M}$-independence, it is possibly pulled in the direction of minimal side to obtain another extreme 2-join that is then used in the construction of T_G. If we do not care about $\mathcal{M}$-independence (as we do not here), we can have the algorithm that constructs T_G just use the minimally-sided 2-join that is first found. This way we obtain T_G with all the other properties, except $\mathcal{M}$-independence, in which every 2-join used is minimally-sided (which is what we need here).

To obtain the desired coloring of G, we process vertices of T_G from bottom up. We start with G^p. As G^p is basic, we color it in $\mathcal{O}(n^3m)$-time by Lemma 5.6. Since G^p and G^p_B are blocks of decomposition w.r.t. a minimally-sided 2-join of G^{p-1}, with G^p_B being a block that corresponds to a minimal side, by Lemma 5.6 we extend the coloring of G^p to G^{p-1} in $\mathcal{O}(n^3m)$-time. We proceed like this up the tree, all the way to the root of T_G, namely $G^0 = G$. As the depth of T_G is at most n, it follows that G can be s-colored in $\mathcal{O}(n^4m)$-time. This completes the proof of the Claim.

We now consider $G \in \mathcal{C}$. By Theorem 2.5 we construct the clique-cutset decomposition tree T of G in $\mathcal{O}(nm)$-time. So all the leaves of T are graphs from $\mathcal{D}$, and there are at most n of them. So to s-color all the leaves, by the Claim, it takes time $\mathcal{O}(n \cdot n^4 m) = \mathcal{O}(n^5 m)$. Finally, process the tree from bottom up, permuting colors of the blocks of decomposition so they agree on the clique cutset and paste the colorings of the blocks together. Going this way all the way up to the root of T, we obtain the desired coloring of G in $\mathcal{O}(n^5 m)$-time.□

6 A note on clique-width

In this section we show that the class $\mathcal{D}_{\text{BASIC}}$ has unbounded clique-width (and hence unbounded rank-width [10]). So the class of (theta,wheel)-free graphs with no clique cutset has unbounded clique-width.

For $k \geq 3$, let C_k be a chordless cycle of length k. For $k \geq 1$, let H_k be the graph on vertex set $\{w_1, \ldots, w_{k+1}, u', u'', v', v''\}$, such that $\{w_1, \ldots, w_{k+1}\}$ induces a chordless path of length k, and the only other edges of H_k are $u'w_1$, $u''w_1$, $v'w_{k+1}$ and $v''w_{k+1}$.

Let Φ_k be the class of planar bipartite $(C_3, \ldots, C_k, H_1, \ldots, H_k)$-free graphs of vertex degree at most 3.

Lemma 6.1 ([8]) *For any positive integer k, the tree- and clique-width of graphs in Φ_k is unbounded.*

Note that every (H_1, C_3, C_4)-free graph is chordless and triangle-free, so the class of triangle-free chordless graphs is the superclass of Φ_4, which, by previous lemma, has unbounded clique-width. Furthermore, if Φ_4' is the class of 2-connected graphs from Φ_4, then Φ_4' also has unbounded clique-width (see, for example, [5]). Moreover, the following holds.

Lemma 6.2 ([5]) *If $\mathcal{G}$ is a class of graph that has unbounded clique-width, then the class $L(\mathcal{G}) := \{L(G) : G \in \mathcal{G}\}$ also has unbounded clique-width.*

This lemma, together with our previous observations, implies that the class $L(\Phi_4')$ has unbounded clique-width. Since $L(\Phi_4') \subseteq \mathcal{D}_{\text{BASIC}}$, we conclude that the class $\mathcal{D}_{\text{BASIC}}$ has unbounded clique-width.

Interestingly, N.K. Le [6] proved that the class of (theta, wheel, prism)-free graphs that do not have a clique cutset has bounded clique-width (using the decomposition theorem for this class from [3]). This means that one can use the machinery of [2] and [13] to obtain faster polynomial-time algorithms for coloring and stable set problem for (theta, wheel, prism)-free graphs.

References

[1] P. Charbit, M. Habib, N. Trotignon, K. Vušković. Detecting 2-joins faster. *Journal of Discrete Algorithms*, 17: 60-66, 2012.

[2] B. Courcelle, J.A. Makowsky, U. Rotics, Linear time solvable optimization problems on graphs on bounded clique width. *Theory of Computing Systems*, 33 (2): 125-150, 2000.

[3] E. Diot, M. Radovanović, N. Trotignon, K. Vušković. The (theta,wheel)-free graphs Part I: ony-prism and only pyramid graphs. arXiv:1504.01862

[4] J. Edmonds. Paths, trees and flowers. *Canad. J. Math.*, 17: 449-467, 1965.

[5] M. Kamiński, V. Lozin, M. Milanič. Recent developments on graphs of bounded clique width. *Discrete Applied Mathematics*, 157: 2747–2761, 2009.

[6] N.K. Le, private communication.

[7] B. Lévêque, F. Maffray, N. Trotignon. On graphs with no induced subdivision of K_4. *Journal of Combinatorial Theory, Series B*, 102: 924–947, 2010.

[8] V. Lozin, D. Rauthenbach. The tree- and clique-width of bipartite graphs in special classes. *Australasian Journal of Combinatorics*, 34: 57-67, 2006.

[9] R.C.S. Machado, C.M.H. de Figueiredo, N. Trotignon. Edge-colouring and total-colouring chordless graphs. *Discrete Mathematics*, 313 (4): 1547-1552, 2010.

[10] S. Oum, P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96 (4): 514-528, 2006.

[11] M. Radovanović, N. Trotignon, K. Vušković. The (theta,wheel)-free graphs Part II: structure theorem. arXiv:1703.08675

[12] M. Radovanović, N. Trotignon, K. Vušković. The (theta,wheel)-free graphs Part IV: induced cycles and paths.

[13] R.E. Tarjan. Decomposition by clique separators. *Discrete Mathematics*, 55: 221-232, 1985.

[14] N. Trotignon, K. Vušković. Combinatorial optimization with 2-joins. *Journal of Combinatorial Theory, Series B*, 102:153-185, 2012.