Towards an Isomorphism Dichotomy for Hereditary Graph Classes

Pascal Schweitzer

Published online: 21 June 2017
© Springer Science+Business Media New York 2017

Abstract In this paper we resolve the complexity of the isomorphism problem on all but finitely many of the graph classes characterized by two forbidden induced subgraphs. To this end we develop new techniques applicable for the structural and algorithmic analysis of graphs. First, we develop a methodology to show isomorphism completeness of the isomorphism problem on graph classes by providing a general framework unifying various reduction techniques. Second, we generalize the concept of the modular decomposition to colored graphs, allowing for non-standard decompositions. We show that, given a suitable decomposition functor, the graph isomorphism problem reduces to checking isomorphism of colored prime graphs. Third, we extend the techniques of bounded color valence and hypergraph isomorphism on hypergraphs of bounded color class size as follows. We say a colored graph has generalized color valence at most $k$ if, after removing all vertices in color classes of size at most $k$, for each color class $C$ every vertex has at most $k$ neighbors in $C$ or at most $k$ non-neighbors in $C$. We show that isomorphism of graphs of bounded generalized color valence can be solved in polynomial time.

Keywords Graph isomorphism · Modular decomposition · Bounded color valence · Reductions · Forbidden induced subgraphs

An extended abstract of this paper was presented at STACS 2015 [35]

Pascal Schweitzer
schweitzer@informatik.rwth-aachen.de

RWTH Aachen University, Aachen, Germany
1 Introduction

Given two graphs $G_1$ and $G_2$, the graph isomorphism problem asks whether there exists a bijection from the vertices of $G_1$ to the vertices of $G_2$ that preserves adjacency and non-adjacency. In this paper we continue the systematic investigation of the complexity of graph isomorphism on hereditary graph classes with a focus on classes characterized by finitely many forbidden induced subgraphs as initiated in [26, 27].

Given a set of finite graphs $H_1, \ldots, H_t$ we define $(H_1, \ldots, H_t)$-free to be the class of all graphs that do not contain any $H_i$ as an induced subgraph. In the light of the unknown complexity status of the graph isomorphism problem, the goal in this context is typically to classify the complexity for the various graph classes into being polynomial time solvable or isomorphism complete (i.e., polynomially equivalent to graph isomorphism). Recently, in [32] it is shown that, assuming that graph isomorphism is not polynomial time solvable in general, there exist graph classes closed under taking (not necessarily induced) subgraphs which are of intermediate complexity. Trivially, this implies the same conditional existence of hereditary graph classes (i.e., graph classes characterized by forbidden subgraphs) of intermediate complexity. However, the construction in [32] intrinsically requires the use of infinitely many forbidden subgraphs. In contrast to this, in [32] it is also shown that there are no intermediate graph classes characterized by finitely many forbidden subgraphs. However, this statement does not carry over to forbidden induced subgraphs and the question of the existence of intermediate graph classes characterized by finitely many forbidden induced subgraphs remains open. A more precise statement of the dichotomy result in [32] is that a graph class characterized by finitely many forbidden subgraphs has a polynomial time solvable graph isomorphism problem if one of the forbidden graphs is a union of subdivided stars, and is graph isomorphism complete otherwise. These graphs, the forests of subdivided stars, also play a central role in the complexity of hereditary graph classes.

With respect to classes defined by forbidden induced subgraphs, there is a dichotomy for the isomorphism problem on $H_1$-free graphs into polynomially solvable and isomorphism complete cases. In [26] the complexity of the isomorphism problem on $(H_1, H_2)$-free graphs is determined for various pairs $(H_1, H_2)$ and the results also follow the “polynomially solvable versus isomorphism complete” dichotomy. The crucial cases that were not resolved are those where either $H_1$ or $H_2$ is a complete graph. More specifically, except for finitely many cases, the unresolved cases were shown to be polynomially equivalent to a class where one of the graphs is a complete graph. In the light of all of these results we conjecture the following:

**Conjecture 1** If $C = (H_1, \ldots, H_t)$-free is a graph class defined by the finite set of forbidden induced subgraphs $H_1, \ldots, H_t$ then graph isomorphism of graphs in $C$ is polynomial time solvable or isomorphism complete.

In this paper we continue the investigation of the complexity of the isomorphism problem on the graph classes focusing on the case of two forbidden subgraphs, where one of the graphs is complete. As mentioned above these are the crucial cases that
were not resolved. For the resolution of those classes, techniques beyond those that were developed in [26] are required. These techniques are presented in this paper leading to the following theorem:

**Theorem 2** On all but finitely many classes of the form \((H_1, H_2)\)-free the graph isomorphism problem is polynomial time solvable or isomorphism complete.

**Contribution** In order to prove the theorem, three new techniques for the structural and algorithmic analysis of graphs are developed in this paper.

Firstly, we develop a methodology to show isomorphism completeness of the isomorphism problem on graph classes by providing a unifying framework for various reductions typically used for that purpose. The advantage of this framework is that it allows a streamlined abstract way to argue why some class is isomorphism complete, which boils down to an algorithmically checkable argument.

Secondly, we generalize the modular decomposition to colored graphs and define the concept of a colored modular decomposition with respect to a decomposition functor. This not only allows us to show that the graph isomorphism problem reduces to colored isomorphism of prime graphs but it also allows us to decompose graphs that are prime with respect to the classic modular decomposition. To show this reduction we also describe how to remodel an algorithm that has access to an oracle producing a complete invariant for a graph class into an algorithm that only has access to an isomorphism test of the graph class.

Thirdly, we extend the techniques of bounded color valence and hypergraph isomorphism on hypergraphs of bounded color class size as follows. We say a colored graph has generalized color valence at most \(k\) if, after removing all vertices in color classes of size at most \(k\), for each color class \(C\) every vertex has at most \(k\) neighbors in \(C\) or at most \(k\) non-neighbors in \(C\). We show that isomorphism of graphs of bounded generalized color valence can be solved in polynomial time. This generalization allows us to perform isomorphism tests for graphs whose automorphism group cannot be forced into having bounded size composition factors even when using finitely many individualization steps. Since for such graphs alternating groups of unbounded size can appear among the composition factors, it seems that the standard group theoretic techniques cannot be directly applied. This shows that indeed new techniques were required to solve the particular cases.

We apply the three mentioned techniques to resolve the complexity of isomorphism on all but finitely many of the graph classes characterized by two forbidden induced subgraphs. For the resolved classes we either provide a reduction from the general problem or a polynomial time algorithm. The applications of the techniques include for example showing that bipartite graphs that are free from a fixed forbidden double star refine into graphs of bounded generalized color valence. This can be used to show that isomorphism of graphs of bounded clique number with a fixed forbidden double star can be solved in polynomial time. We in turn apply this in conjunction with the colored modular decomposition technique to solve isomorphism for graphs of bounded clique number which do not contain \(P_5\) (a path of length 4) as an induced subgraph. We apply the general reductions to show that classes of graphs without cliques of size 4 and certain unions of paths are extensive enough to
have an isomorphism problem that is isomorphism complete. Furthermore, we apply
the modular decomposition techniques in conjunction with the bounded generalized
color valence to analyze the structure of various graph classes of bounded clique
number with certain forbidden forests of subdivided stars.

Related Work  We refer the reader to [3, 23, 34] for an introduction to the diverse
complexity-theoretic results related to the isomorphism problem. There are numerous
results known on the complexity of graph isomorphism of hereditary graph classes.
A collection showing the problem for many classes to be equivalent to the gen-
eral problem is given by Booth and Colbourn [8]. In that paper, the complexity of
classes characterized by one forbidden subgraph \( H \) is shown to depend on whether
the graph \( H \) is an induced subgraph of \( P_4 \) (the path on 4 vertices). The systematic
study of classes characterized by two forbidden subgraphs was initiated in [26].

With regard to algorithms, a recent very general result, implying many results
devised earlier on more special graph classes, is a theorem of Grohe and Marx [17]
that shows that isomorphism of graph classes defined by a forbidden topological
minor can be solved in polynomial time.

There are several applications of modules (sometimes called homogeneous sets)
or some form of modular decomposition in the context of graph isomorphism. For
example, Goldberg’s plain exponential algorithm [15] uses the concept of sections,
which can be seen as colored modules. Furthermore, Junttila and Kaski [22] define
non-uniform components within the individualization-refinement approach that also
constitute colored modules. As described below, Rao [33] also exploits the classic
modular decomposition to devise an isomorphism algorithm for gem and co-gem free
graphs (i.e., \( (P_4 \cup K_1, P_4 \cup K_1) \)-free graphs). His technique can be seen as a special
case of the techniques using colored modular decompositions described in this paper.
Other hereditary graph classes, for which graph isomorphism algorithms make use of
modular decomposition, are for example subclasses of circular-arc graphs (see [11]).

In the early stages of the discovery of the group theoretic technique for the isomor-
phism problem, Luks [29] applied the technique to show that isomorphism of graphs
of bounded maximum degree can be solved in polynomial time. Babai [2] applied
the notion of bounded color valence in his moderately exponential algorithm for the
general isomorphism problem and recently, with additional more sophisticated com-
brinatorial arguments, achieved the breakthrough result of a quasi-polynomial time
algorithm [4]. Miller (see [30]) applied this technique in a series of papers to perform
isomorphism tests of \( k \)-separable and \( k \)-contactable graphs as well as isomorphism
of hypergraphs of bounded color class size (see also [1], [5] and [36]). In our gener-
alization in this paper, only the subgraph induced by the color classes that are not of
bounded size is required to exhibit bounded color valence.

There have been several studies aiming at dichotomy results for computational
problems on graph classes characterized by forbidden induced subgraphs. For exam-
ple, this has been done for the computation of the chromatic number [25], dominating
sets [28], coloring [13], and list coloring [16] for two forbidden subgraphs.

Moreover there are numerous results analyzing whether the clique width of a graph
class characterized by forbidden subgraphs is bounded (see [12] for an extensive list
of references). A variation of techniques presented here was subsequently used to
show that graph isomorphism of graphs of bounded clique width can be solved in polynomial time [18].

Structure of the Paper We mainly apply the three techniques developed in this paper to classes characterized by two forbidden induced subgraphs. However, the intention behind their presentation is to allow them to be applicable in a broader sense to practical and theoretical algorithms for isomorphism of general graphs. The paper first presents the techniques and the second part explains how to apply them to various classes of two forbidden subgraphs.

In the first part of the paper we provide preliminaries such as introducing notation and recalling basic tools (Section 2). We then devise a methodology to prove isomorphism completeness results (Section 3) and briefly describe how to simulate a complete invariant given only an isomorphism algorithm (Section 5). After this we turn to techniques for isomorphism testing using modular decompositions (Section 6) and devise a polynomial time algorithm for graphs of bounded generalized color valence (Section 7).

The reduction techniques are applied to show the isomorphism completeness of various graph classes characterized by forbidden induced subgraphs (Section 4). The algorithmic techniques are then applied to graph classes with forbidden double stars (Section 8) and graph classes of graphs without induced paths of length 4 (Section 9). The techniques are also applied to analyze specific triangle-free graphs (Section 10) and specific graphs of bounded clique number (Section 11). We conclude by showing that together with the theorems in [26] this resolves the complexity of all but finitely many graph classes defined by two forbidden induced subgraphs (Section 12).

2 Preliminaries

In this paper all graphs are finite, simple, undirected graphs. For a graph $G$, by $V(G)$ and $E(G)$ we denote the vertex set and the edge set, respectively. By $N_G(S) = N(S)$ we denote the neighborhood of a set $S$, i.e., the vertices in $V(G) \setminus S$ adjacent to some vertex in $S$. By $id$ we always denote the identity map. The bipartite complement of a bipartite graph $G$ with bipartition classes $A$ and $B$ is obtained by replacing $E(G)$ with $\{(a,b) \mid a \in A, b \in B\} \setminus E(G)$.

We write $H \preceq G$ if the graph $G$ contains a graph $H$ as an induced subgraph. A graph $G$ is $H$-free if $H \not\subseteq G$. It is $(H_1, \ldots, H_k)$-free, if it is $H_i$-free for all $i$. A graph class $\mathcal{C}$ is $H$-free (respectively $(H_1, \ldots, H_k)$-free) if this is true for all $G \in \mathcal{C}$. A graph class $\mathcal{C}$ is hereditary if it is closed under taking induced subgraphs. The class $(H_1, \ldots, H_k)$-free is the class of all $(H_1, \ldots, H_k)$-free graphs. Note that each class $(H_1, \ldots, H_k)$-free is hereditary. We say a graph $G$ contains a graph $H$ (as an induced subgraph) if an induced subgraph of $G$ is isomorphic to $H$.

By $I_t$, $K_t$, $P_t$, and $C_t$ we denote the independent set, the clique, the path, and the cycle on $t$ vertices, respectively. The clique number of a graph $G$ is the largest integer $t$ such that $G$ contains $K_t$. By $H \dot{\cup} H'$ we denote the disjoint union of $H$ and $H'$; we use $tH$ for the disjoint union of $t$ copies of the graph $H$. By $\overline{G}$ we denote
the (edge) complement of $G$. The graph $K_2 \cup I_2$, i.e., the graph obtained from $K_4$ by deleting an edge, is called the diamond.

A *star* is a graph isomorphic to the complete bipartite graph $K_{1,t}$ for some positive integer $t$. The *subdivision* of an edge is the replacement of the edge with a path of length two. A *subdivided star* is a possibly repeated subdivision of a star. Since any subdivided star is a tree, we call the disjoint union of subdivided stars a *forest of subdivided stars*. If a subdivided star has a vertex of degree at least 3 then this vertex is unique and called the *center*. For non-negative integers $a_0, \ldots, a_t$ with $a_t > 0$, we define the graph $H(a_t, \ldots, a_1, a_0)$ to be the disjoint union of an independent set of size $a_0$ with the following subdivided star $H$. The star $H$ is the subdivided star that for $i \in \{1, \ldots, t\}$ has exactly $a_i$ leaves at distance $i$ from the center and no other leaves. (If $\sum_{i=1}^t (a_i) < 3$ the center of $H$ is defined so that the graph $H$ is a path whose two leaves have suitable distances from that center.) Examples of such graphs are depicted in Fig. 1.

In [26] it is shown that isomorphism of $(H_1, \ldots, H_t)$-free graphs is isomorphism complete unless one of the forbidden graphs is a forest of subdivided stars.

In this paper a *colored graph* is a vertex colored graph whose coloring does not need to be proper (i.e., adjacent vertices can have the same color). Isomorphisms between colored graphs are required to respect the colors, that is, they must map vertices to vertices of the same color. A *singleton* is a vertex with a unique color. The *naive vertex refinement* algorithm, or 1-dimensional Weisfeiler-Lehman algorithm, is a standard technique of repeatedly recoloring the vertices, refining the partition induced by the colors, by using the multiplicity of colors appearing among the neighbors of a vertex (see for example [34]). It has the property that after the refinement, the number of neighbors a vertex $v$ has in a color class $C$ only depends on the colors of $v$ and $C$. Such a coloring is called *stable*. A graph has *color valence at most* $k$ if for every vertex $v$ and every color class $C$ there are at most $k$ neighbors of $v$ in $C$ or there are at most $k$ non-neighbors of $v$ in $C$.

### 3 Reductions

In this section we develop a systematic approach to proving isomorphism invariant reductions. This provides general means to construct isomorphism complete graph

![Examples of subdivided stars with possibly added isolated vertices](Fig. 1)

(a) The graph $H(1, b, 0)$. It plays a central role in Section 11.

(b) The graph $H(1, 0, b, 1)$. It plays a central role in Section 10.
classes. Standard reductions like subdividing, taking complements, and adding isolated vertices fall into this framework. Likewise, most reductions performed in [26] and also various reductions in [8] fall into this framework.

**Definition 1** Let $J$ be a finite set and $L: J \times J \rightarrow \{A, N\}$ be a labeling assigning every ordered pair of vertices the label $A$ for adjacent or $N$ for non-adjacent. Moreover let $L_N: J \times J \rightarrow \mathbb{N} \cup \{\infty\}$ be a labeling assigning every ordered pair of vertices a non-negative integer or infinity.

A graph $G$ belongs to the class encoded by the labeled graph $(J, L, L_N)$ if there exists a map $\phi: V(G) \rightarrow J$ such that the following hold:

1. If $v \in V(G)$ and $j \in J$ such that $L(\phi(v), j) = A$ and $L_N(\phi(v), j) \neq \infty$ then there are at most $L_N(\phi(v), j)$ vertices $v' \in V(G) \setminus \{v\}$ that are non-adjacent to $v$ such that $\phi(v') = j$.
2. If $v \in V(G)$ and $j \in J$ such that $L(\phi(v), j) = N$ and $L_N(\phi(v), j) \neq \infty$ then there are at most $L_N(\phi(v), j)$ vertices $v' \in V(G) \setminus \{v\}$ that are adjacent to $v$ such that $\phi(v') = j$.

In this definition, the triple $(J, L, L_N)$ should be thought of as a generalized graph.

The graph class encoded by $(J, L, L_N)$ then contains graphs that can be obtained by replacing the elements of $J$ with sets of vertices. In some contexts, this is referred to as blowing-up the elements of $J$. Adjacency of the new vertices is essentially governed by the adjacency of the original graph. However, the values of $L_N$ control the number of exceptions to this rule that are allowed per vertex.

The definition captures various constructions that are used in graph theory. A first class of examples is formed by the complete multipartite graphs, which turn out to be graphs modeled with the function $L$ satisfying $L(x, y) = N$ if and only if $x = y$ and the function $L_N$ being the constant function evaluating to $0$.

A second, more general class of examples modeled by the definition is the graphs of bounded color valence, which are graphs that frequently appear in the context of graph isomorphism. Such graphs are obtained whenever $L_N$ is a bounded function.

A third example is the class of graphs that map homomorphically onto a finite graph $H$. This class is obtained by letting $(J, L)$ model the graph $H$ (i.e., $V(H) = J$ and $L(x, y) = A$ if and only if $\{x, y\}$ is an edge of $H$) and setting $L_N(x, y) = 0$ if $L(x, y) = N$ and $L_N(x, y) = \infty$ otherwise.

Of interest to us is the complexity of the isomorphism problem of graph classes encoded by a triple $(J, L, L_N)$. It turns out that when $L_N$ is a bounded function then isomorphism can be solved in polynomial time.

**Theorem 3** Let $(J, L, L_N)$ be a triple that encodes a graph class. If all values of $L_N$ are finite then isomorphism of graphs encoded by $(J, L, L_N)$ can be solved in polynomial time.

**Proof** We show the statement by induction on $|J|$. However, we show a statement that is slightly stronger. Namely, we show that there exists an algorithm that solves
the isomorphism problem for input pairs where only one of the graphs is required to be encoded by \((J, L, L_N)\). Moreover we require that the algorithm does not produce any incorrect answers when queried with a pair of graphs both of which are not encoded by \((J, L, L_N)\). That is, when both graphs are not encoded by \((J, L, L_N)\) the algorithm either correctly determines whether the graphs are isomorphic or states that the answer was not determined.

For \(|J| = 1\) the encoded graphs have bounded degree or bounded co-degree, so isomorphism can be solved in polynomial time [29]. Let \((J, L, L_N)\) be an encoding with \(|J| > 1\). We first describe a reduction rule. Suppose there are \(j_1, j_2 \in J\) such that for all \(j \in J\) we have \(L(j_1, j) = L(j_2, j)\) and \(L(j, j_1) = L(j, j_2)\). We now argue that in this case we can replace \(J\) by a smaller set at the expense of increasing the values in \(L_N\). To this end, let \(\pi\) be the projection that maps \(j_2\) to \(j_1\), i.e., the map that satisfies \(\pi(j) = j\) if \(j \neq j_2\) and \(\pi(j_2) = j_1\). We claim that every graph encoded by \((J, L, L_N)\) is also encoded by \((J', L', L'_N)\) where \(J' = J \setminus \{j_2\}\), \(L|_{J'\setminus\{j_2\}}\) is the restriction of \(L\) to \(J'\), and \(L'(j, j') = 2 \cdot \max\{L(\ell, \ell') | \ell \in \pi^{-1}(j), \ell' \in \pi^{-1}(j')\}\). Indeed, if for a graph \(G\) a map \(\phi: V(G) \to J\) is an encoding by \((J, L, L_N)\) then the map \(\pi \circ \phi\) is an encoding by \((J', L', L'_N)\). By induction we can solve isomorphism for graphs encoded by \((J', L', L'_N)\) in polynomial time.

Suppose now that the above situation does not arise. We can assume that for every map that encodes one of the input graphs by \((J, L, L_N)\), the map to \(J\) is surjective. Otherwise we can solve the isomorphism problem on a smaller encoding by induction.

We will now guess an encoding \(\phi_1\) of the input graph \(G_1\). For each \(j\) in \(J\) we guess a vertex \(v \in V(G_1)\) with \(\phi_1(v) = j\). Let \(M\) be the set of guessed vertices. We now guess all vertices that form an exception with a vertex in \(M\). That is, we guess all vertices \(u\) for which there exists a \(v \in M\) such that \(L(\phi_1(v), \phi_1(u)) = A\) but \(u\) and \(v\) are not adjacent or such that \(L(\phi_1(v), \phi_1(u)) = N\) but \(u\) and \(v\) are adjacent. Since \(L_N\) is bounded, the possible number of vertices that form an exception with a vertex in \(M\) is bounded. For all these vertices we guess their image under \(\phi_1\). By the reduction described above, for all non-exceptional vertices not in \(M\) there is only one choice for their image under \(\phi_1\) not violating the encoding or making them an exception. Thus \(\phi_1\) is completely determined by the guesses we have performed. The process of guessing an image under \(\phi_1\) is performed only for a bounded number of vertices. Thus, there are only polynomially many options.

Similarly for \(G_2\) there are only polynomially many options for suitable encodings. We can also check validity of an encoding in polynomial time. Once an encoding is determined, the graphs have bounded color valence and isomorphism can be checked in polynomial time.

The theorem implicitly shows that isomorphism of graphs which have bounded color valence for some coloring that uses a bounded number of color classes can be solved in polynomial time, even if the color classes are not given.

**Corollary 1** For every positive integer \(c\), graph isomorphism of graphs whose vertices can be partitioned into \(c\) color classes such that the graph has color valence at most \(c\) can be solved in polynomial time.
Proof Every graph that can be partitioned into $c$ color classes such that the graph has color valence at most $c$ can be encoded by a triple $(J, L, L_N)$ with $|J| \leq c$ and a function $L_N$ bounded by $c$. There are only finitely many encodings of this type. For each of these we can apply the algorithm from the proof of the theorem. The input graphs are isomorphic if and only if at least one of the calls to the algorithm claims them to be isomorphic. 

While encodings can be used to show polynomial time solvability of certain graph classes, they can also be used to show hardness results as follows.

**Definition 2** An encoding $(J, L, L_N)$ is a simple path encoding in case $L$ is symmetric (i.e., if $L(j, j') = L(j', j)$ holds) and there is a sequence of vertices $(p_1, \ldots, p_t)$ of length at least 2 in $J$ such that we have $L_N(p_1, p_2) = \infty$, $L_N(p_t, p_{t-1}) \geq 2$ and for all $k \in \{1, \ldots, t-1\}$ we have $L_N(p_k, p_{k+1}) \geq 1$ and $L_N(p_{t-1}, p_k) \geq 1$.

Intuitively, a simple path encoding allows enough freedom to encode bipartite graphs with one bipartition class having vertices of degree two. We can formally prove this statement in the form of a reduction.

**Theorem 4** The class of graphs encoded by a simple path encoding is graph isomorphism complete.

**Proof** First we describe a reduction and then show it is isomorphism invariant. Let $(J, L, L_N)$ be a simple path encoding with the path $P = (p_1, \ldots, p_t)$. Let $G = (V, E)$ be a graph. We now describe a reduction that transforms $G$ into a graph $G'$ that can be encoded by $(J, L, L_N)$. The vertex set of $G'$ is $V(G') = V(G) \cup E(G) \cup \{(v, \{v,v'\}, k) \mid \{v,v'\} \in E(G) \text{ and } k \in \{2, \ldots, t-1\}\}$. We define a map $\phi : V(G') \rightarrow J$ such that for all $v \in V(G)$ we have $\phi(v) = p_1$ and for all $e \in E(G)$ and $i \in \{2, \ldots, t-1\}$ we have $\phi(e) = p_i$ and $\phi(\{e,i\}) = p_1$. The edge set $E(G')$ will be formed as a symmetric difference as follows:

- Let $E_1$ be the set of pairs $\{u, u'\} \subset V(G')$ for which $L(\phi(u), \phi(u')) = A$.
- Let $E_2$ be the set of pairs
  $$\{(v, \{v,v'\}, i), (v, \{v,v'\}, i+1) \mid \{v,v'\} \in E(G), i \in \{2, \ldots, t-2\}\}.$$  
- In the case $t > 2$ we define $E_3$ to be the set of pairs $\{v, \{v,v'\}, 2\} \mid \{v,v'\} \in E(G)$ and $E_4$ to be the set of pairs $\{v, v', (v, \{v,v'\}, t-1) \mid \{v,v'\} \in E(G)\}$.
- If $t = 2$ we set $E_3$ as $\{v, \{v,v'\}\}$ and $E_4$ is the empty set.

The edge set $E(G')$ is $E_1 \triangle (E_2 \cup E_3 \cup E_4)$, where $\triangle$ denotes the symmetric difference.

By following the construction, given a graph $G$, the graph $G'$ can be constructed in polynomial time. The map $\phi$ is an encoding and shows that the graph $G$ can be encoded by the triple $(J, L, L_N)$. Since all steps of the construction are isomorphism invariant, given two isomorphic graphs $G_1$ and $G_2$ the respective graphs produced by the construction to each of them are isomorphic as well.
To finish the proof we need to show the following: there is a graph isomorphism complete class of graphs \( C \) such that for any two non-isomorphic graphs in \( C \) the construction produces two non-isomorphic graphs.

For this let \( C \) be the class of graphs that have the following properties: the number of edges is at least \( 5n \). The minimum degree and the maximum co-degree are at least 4. This class can be seen to be isomorphism complete by using the operation that adds 6 non-adjacent vertices to a graph and making them adjacent to every other vertex. There are various other reductions showing that this is an isomorphism complete class, for example the class of self-complementary graphs fulfills these properties, which is an isomorphism complete class \([10]\).

To show that the construction described above maps non-isomorphic graphs from \( C \) to non-isomorphic graphs it suffices to reconstruct the labeling. To show that the labeling is reconstructible, we use a degree argument. The degree of a vertex with label \( p_1 \) is \( d + k_1n - \varepsilon + k_2m \) where \( d \) is the degree of the vertex in \( G \), the variables \( k_1, \varepsilon \in \{0, 1\} \) depend on the value of \( L(p_1, p_1) \), and \( k_2 \in \{0, \ldots, |J|\} \) is a constant that depends on the values of \( L(p_1, p_1) \).

The degree of a vertex with a label \( p_i \) different from \( p_1 \) is \( k_1n + k_2m + k_3 \), where again \( k_1, k_2 \) and \( k_3 \) depend on the labeling \( L \). More precisely, \( k_1 \in \{0, 1\} \) depends on \( L(p_i, p_1) \), \( k_2 \) depends on the values of \( L(p_i, p_i) \) and \( k_3 \) is an integer with an absolute value that is bounded above by 2 arising from exceptions allowed by \( L_N \).

Since \( m \geq 5n \) the degree of vertices with label \( p_1 \) cannot be expressed as a sum with constraints of the latter kind (i.e., as \( k_1n + k_2m + k_3 \)). Likewise the degree of vertices with a label different from \( p_1 \) cannot be expressed with constraints of the former kind (i.e., as \( d + k_1n - \varepsilon + k_2m \)). This implies that the set of vertices with label \( p_1 \) is reconstructible. Since for \( i > 1 \) the set of vertices with label \( p_i \) contains exactly the vertices that have a neighbor and a non-neighbor among the vertices with label \( p_{i-1} \), we can reconstruct \( \phi \) and thus the partition. Once the partition is reconstructed, the edge sets \( E_i \) can be reconstructed, and thus the original graph can be reconstructed. This shows that the reduction preserves isomorphism for graphs in \( C \) and finishes the proof.

In the next section, we are going to apply the theorem to show that various classes (namely the classes \((2K_2 \cup K_1, K_4)\)-free, \((P_6, P_4 \cup P_2, K_4)\)-free, \((H(1, 0, 3, 0), K_4)\)-free, bipartite \((2P_3 \cup K_1)\)-free, \((H(1, 0, 2, 0), K_5)\)-free) are isomorphism complete. This in turn implies that various classes \((H_1, H_2)\)-free that are superclasses of one of these are isomorphism complete (such as \((K_3, 2P_3 \cup K_1)\)-free). The graph classes characterized by encodings as described in Section 4 contain many classes useful to show isomorphism completeness of various hereditary graph classes. These results are sufficient for the purpose of this paper, proving Theorem 2. However, we remark that there are several completeness results for classes of the form \((H_1, H_2)\)-free where further restrictions on the encoded graphs become necessary. For example when excluding the diamond, it is beneficial to require that there is only one encoded path between vertices mapped to \( p_1 \) and \( p_f \). Furthermore, there are some reductions that do not seem to fit into the reduction scheme of this section, even with further restrictions. Among those are in particular the line graph reductions (see [26]).
4 Isomorphism Complete Classes

The reduction techniques of the previous section provide a systematic approach to show isomorphism completeness results for graphs characterized by finitely many forbidden induced subgraphs. The following examples show how to apply the techniques in a mechanical way to show isomorphism completeness of graph classes characterized by forbidden induced subgraphs.

**Theorem 5** The isomorphism problem on the class of $(2K_2 \cup K_1, K_4)$-free graphs, of $(P_6, P_4 \cup P_2, K_4)$-free graphs, and of $(H(1, 0, 3, 0), K_4)$-free graphs is graph isomorphism complete.

**Proof** We first show the statement for the class of $(2K_2 \cup K_1, K_4)$-free graphs. For this, let $((p_1, p_2, p_3), L, L_N)$ be the encoding given by setting $L(v, v') = N$ if and only if $\{v, v'\} \neq \{p_1, p_3\}$ and

$$L_N(v, v') = \begin{cases} \infty & \text{if } (v, v') = (p_1, p_2) \\ 1 & \text{if } (v, v') = (p_2, p_1) \\ 1 & \text{if } (v, v') = (p_2, p_3) \\ 2 & \text{if } (v, v') = (p_3, p_2) \\ 0 & \text{otherwise}. \end{cases}$$

This encoding is depicted in Fig. 2a. By Theorem 4 it suffices to show that graphs encoded by $((p_1, p_2, p_3), L, L_N)$ are $(2K_2 \cup K_1)$-free and $(K_4)$-free. This is equivalent to showing that neither $2K_2 \cup K_1$ nor $K_4$ can be encoded. For $K_4$ this is obvious since all encoded graphs are tripartite. For $2K_2 \cup K_1$ we conduct a case distinction. First note that, under a tentative encoding, it cannot be the case that

![Fig. 2](https://example.com/fig2.png)

(a) An encoding that is suitable for $(2K_2 \cup K_1, K_4)$-free graphs.

(b) An encoding that is suitable for $(P_6, P_4 \cup P_2, K_4)$-free graphs.

(c) An encoding that is suitable for $(H(1, 0, 3, 0), K_4)$-free graphs.

Fig. 2 The figures show two encodings $(J, L, L_N)$ with symmetric function $L$. 
all elements of \( \{p_1, p_2, p_3\} \) are in the image of the encoding, as otherwise two non-adjacent vertices would be mapped to \( p_1 \) and \( p_3 \) each. If the isolated vertex is mapped to \( p_1 \) (respectively \( p_3 \)) then all other vertices must be mapped to \( p_2 \) or \( p_3 \) (respectively \( p_2 \) or \( p_3 \)). Since end points of edges cannot be mapped to the same element of \( \{p_1, p_2, p_3\} \), two vertices are mapped to \( p_2 \). This implies that some vertex mapped to \( p_2 \) has two non-neighbors mapped to \( p_1 \) (respectively \( p_3 \)) contradicting that \( L_N(p_2, p_1) = 1 \) (respectively \( L_N(p_2, p_3) = 1 \)). In the case where the isolated vertex is mapped to \( p_2 \) there are two vertices mapped to \( p_1 \) or two vertices mapped to \( p_3 \) which similarly leads to a contradiction.

To show the second part of the theorem for \((P_6, P_4 \cup P_2, K_4)\)-free graphs we use the encoding \( \{(p_1, p_2, p_3), L', L_N\} \) with \( L'(v, v') = N \) if and only if \( v = v' \) (and the same function \( L_N \) as in the first part of the proof). This encoding is depicted in Fig. 2b. Analogously to the first part, it suffices for us to show that the graphs \( P_6, P_4 \cup P_2 \) and \( K_4 \) cannot be encoded. Again, the encoded graphs are tripartite so \( K_4 \) cannot be encoded.

We first note that for any encoding of \( P_4 \), some vertex is mapped to \( p_1 \) and some vertex is mapped to \( p_3 \). Indeed, otherwise, some vertex is mapped to \( p_2 \), two neighbors are mapped to \( p_1 \), or two neighbors mapped to \( p_3 \).

Suppose there is an encoding of \( P_6 \). Since \( P_4 \leq P_6 \), some vertex is mapped to \( p_1 \) and some vertex is mapped to \( p_3 \). Since every non-empty complete bipartite graph on 4 vertices contains a vertex of degree 3 or a cycle, this implies that in any encoding at most 3 vertices can be mapped to \( p_1 \) or \( p_3 \). Thus 3 vertices must be mapped to \( p_2 \). The path \( P_6 \) only has two independent sets of size 3. The vertices not mapped to \( p_2 \) also form an independent set, so they must be all mapped to \( p_1 \) or all mapped to \( p_3 \) which contradicts the observation about encodings of \( P_4 \).

Suppose now that there is an encoding of \( P_4 \cup P_2 \). We already know that some vertex of \( P_4 \) is mapped to \( p_1 \) and some vertex is mapped to \( p_3 \). However, some vertex of \( P_2 \) must also be mapped to \( p_1 \) or \( p_3 \) and this vertex would thus have to be adjacent to a vertex of \( P_4 \).

To show the third part of the theorem for \((H(1, 0, 3, 0), K_4)\)-free graphs we use the encoding \( \{(p_1, p_2, p_3), L', L_N\} \) with \( L'(v, v') = A \) if and only if \( \{v, v'\} = \{p_2, p_3\} \) or \( \{v, v'\} = \{p_1, p_3\} \) (and the same function \( L_N \) as above). This encoding is depicted in Fig. 2c. By the same arguments as before \( K_4 \) cannot be encoded, so it suffices to show that \( H(1, 0, 3, 0) \) cannot be encoded. Let \( h \) be the vertex of degree 4 in the graph \( H(1, 0, 3, 0) \) and let \( x_1, x_2 \) and \( x_3 \) be the vertices on the path from \( h \) to the leaf of distance 3 from the center. Suppose \( h \) is mapped to \( p_1 \). In this case \( x_1 \) could not be mapped to \( p_2 \) since \( x_2 \) would have to be mapped to \( p_3 \), which contradicts that \( x_2 \) is not adjacent to \( h \). Thus \( x_1 \) would have to be mapped to \( p_3 \). Then \( x_2 \) is mapped to \( p_1 \), since otherwise \( x_3 \) cannot be mapped anywhere. This implies that all leaves are mapped to \( p_2 \). However, then \( x_1 \), which is mapped to \( p_3 \), has 3 non-neighbors which are mapped to \( p_2 \) contradicting the requirements of the encoding. Supposing next that \( h \) is mapped to \( p_2 \), then at least one, and thus all of its neighbors are mapped to \( p_3 \). Since \( x_2 \) must then be mapped to \( p_2 \) this would mean that \( x_2 \) has 2 non-neighbors that are mapped to \( p_3 \) contradicting the encoding. Finally suppose \( h \) is mapped to \( p_3 \). This implies that one of the vertices \( x_2 \) and \( x_3 \) is mapped to \( p_2 \) while the other is mapped to \( p_3 \). This in turn implies that all leaf
vertices adjacent to $h$ are mapped to $p_2$ which shows that some vertex in $\{x_2, x_3\}$ is mapped to $p_3$ but has 3 non-neighbors that are mapped to $p_2$ which yields a contradiction. □

As can been seen in the proof of the previous theorem, given an encoding, checking whether no $H$-free graph can be encoded amounts to checking whether $H$ itself can be encoded, which is a finite computation that can in fact be fully automated. We consider another example concerned with bipartite graphs.

**Theorem 6** The isomorphism problem for bipartite $\left(2P_3 \cup K_1\right)$-free graphs is graph isomorphism complete.

**Proof** Let $((p_1, p_2, p_3, p_4), L, L_N)$ be the encoding being depicted in Fig. 3 with $L(v, v') = N$ if and only if $\{v, v'\} \neq \{p_1, p_4\}$ and

$$L_N(v, v') = \begin{cases} 
\infty & \text{if } (v, v') = (p_1, p_2) \\
1 & \text{if } (v, v') = (p_2, p_1) \\
1 & \text{if } (v, v') = (p_2, p_3) \\
1 & \text{if } (v, v') = (p_3, p_2) \\
1 & \text{if } (v, v') = (p_3, p_4) \\
2 & \text{if } (v, v') = (p_4, p_3) \\
0 & \text{otherwise.}
\end{cases}$$

For every encoding of a graph the pre-images of $\{p_1, p_3\}$ and $\{p_2, p_4\}$ must be independent sets and thus the encoded graphs are bipartite. Let $\pi$ be an encoding of $2P_3 \cup K_1$ and consider the bipartition induced by the pre-images just considered. We first observe that no independent set of $2P_3 \cup K_1$ can be partitioned such that there are 2 vertices in each bipartition class. Indeed, if this were the case, then two vertices are mapped to $p_2$ or one pair of endpoints is mapped to $p_3$. Due to the requirements of the encoding this implies that the other pair is mapped to $p_1$ or $p_4$, which again gives a contradiction to the requirements of the encoding.

Since $P_3$ is connected, the endpoints of each $P_3$ are contained in the same bipartition class. Thus by our observation about independent sets of size 4, all end points of the paths are mapped to the same bipartition class. Furthermore, by the same reason, the isolated vertex must be mapped to the same bipartition class. This means the other two vertices have 3 non-neighbors in the other bipartition class. Thus they are mapped to $p_1$. This implies that all other vertices are mapped to $p_2$ giving a contradiction to $L(p_2, p_1)$ for the isolated vertex.

Fig. 3 An encoding that is suitable for bipartite $(2P_3 \cup K_1)$-free graphs

© Springer
On a more intuitive level, one can also interpret the proof as an argument showing that $2P_3 \cup K_1$ cannot be partitioned into two parts such that its bipartite complement is a forest of subdivided stars.

Finally, we conclude with one more example-application of the reduction technique.

**Theorem 7** The graph isomorphism problem on $(H(1, 0, 2, 0), K_5)$-free graphs is graph isomorphism complete.

**Proof** Let $((p_1, p_2, p_3, p_4), L, L_N)$ be the encoding being depicted in Fig. 4 with $L(v, v') = N$ if and only if $\{v, v'\} = \{p_1, p_2\}, \{v, v'\} = \{p_3, p_4\}$ or $v = v'$, and with $L_N$ defined as in the proof of Theorem 6. Since graphs encoded by this encoding are 4-partite, it suffices to show that $H(1, 0, 2, 0)$ cannot be encoded. Let $h$ be the vertex of degree 3 in $H(1, 0, 2, 0)$ and let $x_1, x_2$ and $x_3$ be the vertices on the path of length 4 starting in $h$. In any encoding, the vertices $x_3$ and $x_2$ are mapped to different elements of $\{p_1, p_2, p_3, p_4\}$. Since there is an edge in $H(1, 0, 2, 0)$ whose endpoints are non-adjacent to the endpoints of the edge $\{x_2, x_3\}$ the possible images of $\{x_2, x_3\}$ under a tentative encoding are $\{p_1, p_2\}$, $\{p_2, p_3\}$ and $\{p_3, p_4\}$. If the set $\{x_2, x_3\}$ is mapped to $\{p_2, p_3\}$ then the 3 vertices that are neither adjacent to $x_2$ nor $x_3$ must all be mapped to $p_2$ or $p_3$. In any case, two of these vertices are mapped to the same vertex in $\{p_2, p_3\}$ and there is a vertex in $\{x_2, x_3\}$ non-adjacent to both of them and not mapped to the same vertex in $\{p_2, p_3\}$ contradicting the encoding. We now assume that $\{x_2, x_3\}$ is mapped to $\{p_1, p_2\}$. The case where the set is mapped to $\{x_3, x_4\}$ is symmetric. This implies that all vertices except possibly $x_1$ are mapped to $p_1$ or $p_2$. Since vertices that are mapped to $p_2$ can have at most one neighbor mapped to $p_1$ we conclude that $h$ is mapped to $p_1$ and the leaves adjacent to $h$ are mapped to $p_2$. Since vertices mapped to $p_3$ cannot have two non-neighbors mapped to $p_2$ we conclude that $x_1$ is mapped to $p_2$. This implies that $x_2$ is mapped to $p_1$ which is a contradiction since $x_1$ cannot have two neighbors mapped to $p_1$. \(\square\)

## 5 Isomorphism, Invariants and Canonical Labeling

We recall the concept of a complete invariant and the related concept of a canonical labeling. A *complete graph invariant* for a graph class $\mathcal{C}$ is a map $\text{Inv}: \mathcal{C} \to \mathcal{D}$ into some class $\mathcal{D}$ such that for graphs $G_1$ and $G_2$ in $\mathcal{C}$ we have $\text{Inv}(G_1) = \text{Inv}(G_2)$ if and only if $G_1$ and $G_2$ are isomorphic. A *canonical labeling* is a map that assigns...

\[ N \quad (\infty, N, 1) \quad (1, A, 1) \quad (1, N, 2) \quad N \]

\[ p_1 \quad p_2 \quad p_3 \quad p_4 \]

*Fig. 4* An encoding that is suitable for bipartite $(H(1, 0, 2, 0), K_5)$-free graphs.
to every graph $G$ a graph $C(G)$ with vertex set $V(C(G)) = \{1, \ldots, |G|\}$ and an isomorphism $\phi: G \rightarrow C(G)$ such that the map assigning $C(G)$ to $G$ is a complete invariant.

There are relatively general techniques with which one can turn a complete invariant into a canonical labeling algorithm [19, 20, 24]. These techniques are typically only applicable when dealing with invariants for graph classes that have additional properties, for example the ability to attach gadgets to vertices in order to encode information or having balanced separators of bounded size. However, in our situation we can employ colors to achieve this.

Intuitively the technique finds the first vertex of the canonical labeling by individualizing each vertex one by one and choosing the vertex which yields lexicographically the smallest invariant. By repeating the process, having already individualized the first vertex, then individualizing each possible second vertex and applying the invariant, the second vertex is found. Iterating this procedure yields the canonical labeling.

In this paper, we are mainly interested in isomorphism algorithms, as opposed to canonical labeling algorithms or complete invariants. We will therefore require a tool to simulate an invariant within one execution of our algorithm, given only an algorithm that performs isomorphism checks. The key difference between having an invariant and an isomorphism algorithm is that the invariant must be consistent across different executions of an algorithm. Our simulated invariant will not be consistent across different calls of the same algorithm.

**Theorem 8** Let $A$ be a polynomial time algorithm with access to a complete invariant $O$ for a graph class $C$ given as oracle. Suppose the outputs of $A$ are independent of the choice of the invariant $O$. If isomorphism of graphs in $C$ can be solved in polynomial time then there is a polynomial - time algorithm $B$ whose outputs coincide with those of $A$, which does not require access to an oracle.

**Proof** Let Iso be a polynomial time isomorphism algorithm for graphs in $C$. We simulate an algorithm that has access to a complete invariant as follows. We maintain a list of graphs that contains the first graph of each isomorphism class which has been used as input to the oracle so far. Whenever a new oracle call is made with a graph $G$, we check, using Iso, whether $G$ is isomorphic to a graph that was already used as input to the oracle. If so, the returned invariant is the graph in the database isomorphic to $G$. Otherwise $G$ is added to the list and returned as invariant. Concerning the running time, checking whether a graph already has an isomorphic copy in the list requires a number of calls to Iso equal to the number of graphs already added to the list. The total number of calls to Iso is thus at most quadratic in the number of oracle calls.

We will use the theorem to replace an invariant with an isomorphism algorithm in the next section when dealing with modular decompositions (more precisely to prove Theorem 9).

The reason why the technique described in the proof of Theorem 8 is not viable in practical applications of isomorphism algorithms is two-fold. Firstly, the algorithm
suffers from a polynomial increase in particular in the space required and secondly, the technique makes the algorithm inherently sequential, whereas the advantage of canonization is its parallelizability. Another way of stating this is that the values of the invariant are not compatible across different executions of the algorithm.

6 Colored Modular Decomposition

In this section we are concerned with modular decompositions and their application to the isomorphism problem. We will work with colored graphs since this is convenient in the graph isomorphism context. However, we will also generalize the concept of a module to that of a colored module, since this is required by our later applications. Since we do not require previous knowledge about the uncolored modular decomposition, we will not define it. We refer the reader to the survey by Habib and Paul [21] for more information on the uncolored decomposition and its algorithmic applications. For the colors, we will assume that there is a linear order on the colors. In algorithmic applications such a linear order can always be obtained by comparing the bit-strings corresponding to the colors lexicographically.

Definition 3 Let $G$ be a colored graph. A colored module is a subset $M$ of $V(G)$ such that for all $v \in V(G) \setminus M$, if $x, x' \in M$ are of the same color then either $v$ is adjacent to both vertices $x$ and $x'$ or to neither $x$ nor $x'$.

A module $M$ is non-trivial if it contains at least two vertices that cannot be distinguished by vertices outside of the module. That is, $M$ is non-trivial if there are $x, y \in M$ such that for all $v \in V(G) \setminus M$ the vertex $v$ is adjacent to $x$ if and only if it is adjacent to $y$. Note that every module that contains two vertices of the same color is non-trivial.

Definition 4 A map assigning to every graph $G$ a subset of the vertices $M(G)$ is said to be isomorphism invariant if for every isomorphism $\phi: G \rightarrow G'$ we have $\phi(M(G)) = M(G')$.

A map that assigns to every graph $G$ a partition of a subset of the vertices $M(G) = \{M_1, \ldots, M_k\}$ is said to be isomorphism invariant if for every isomorphism $\phi: G \rightarrow G'$ we have $M(G') = \{\phi(M_1), \ldots, \phi(M_k)\}$.

Definition 5 A decomposition functor is a map assigning to every graph $G$ a partition of a subset of the vertices into modules that is isomorphism invariant.

We elaborate briefly on the choice of terminology “decomposition functor”. A decomposition functor can be seen as a covariant functor in the sense that every homomorphism between graphs induces a homomorphism between the subgraphs induced by the modules. Moreover it induces a homomorphism between the quotient graphs defined below. Similarly, color refinements can be seen as functors (see [34, Page 22] for more information on this). Note that a decomposition functor assigns to every graph a decomposition. It is thus not simply a decomposition of just a single
graph. This is what the word functor expresses, rather than talking about just having a decomposition for graphs. The crux is that these decompositions must be consistent across isomorphism classes.

Technically a decomposition functor is not a set-theoretic map since the class of all graphs is not a set. However, these category theoretic hurdles will not be important in the rest of this paper. In any case it does not seem that the word “invariant decomposition map” would express the concept really well.

A graph \( G \) is prime with respect to a decomposition functor \( \text{Mod} \) if \( \text{Mod}(G) \) does not contain non-trivial modules. While there is a standard decomposition functor for the uncolored case, for the colored case it is in general not clear whether we can find a useful functor to decompose the graphs, and we have to find such a functor for a given graph class first, in order to decompose the graphs.

We illustrate that colored modules allow us to decompose graphs that are prime as uncolored graphs. Consider a connected, bipartite graph with bipartition classes \( A \) and \( B \) of different colors. Let \( A' \subset A \) and \( B' \subset B \) be sets such that \( A' \cup B' \) induces a matching between the vertices in \( A' \) and the vertices in \( B' \) and such that all vertices of \( A' \) are adjacent to all vertices of \( B \setminus B' \) and all vertices of \( B' \) are adjacent to all vertices of \( A \setminus A' \). Then \( A' \cup B' \) forms a colored module, but no vertex of \( A' \cup B' \) is contained in a non-trivial classic (uncolored) module.

In the remainder of this section, we argue for certain decomposition functors that graph isomorphism for a hereditary graph class can be solved in polynomial time if the isomorphism problem for colored prime graphs in the class can be solved in polynomial time. To facilitate the proof we can assume that we are given a complete invariant for the prime graphs in the hereditary graph class and then apply Theorem 8.

Our next goal is to define the concept of the quotient graph. In the uncolored case, the quotient graph is obtained by replacing each module with a single vertex whose adjacency to the rest of the graph is the same as that of every vertex of the module. However, for the colored case, the adjacency to the rest of the graph depends on the color of the vertex in the module. This means that for every adjacency type we need to retain a vertex that has the same adjacency type with respect to vertices outside the module. (For a vertex \( v \), the adjacency type with respect to vertices outside the module is the set of vertices outside the module adjacent to \( v \).)

A replacement operator is an isomorphism invariant map that assigns every non-trivial module \( M \) in a decomposition of a graph \( G \) an induced subgraph of \( M \) in which the vertices are possibly recolored. We require that for every adjacency type of vertices in \( M \) at least one vertex of \( M \) is maintained. Let \( \text{Inv}(M) \) be a complete graph invariant. Given a family of modules \( \{M_1, \ldots, M_k\} \) that partitions the graph, the quotient graph is obtained by simultaneously replacing all modules using the replacement operator (i.e., removing from the module \( M_i \) all vertices not in the induced subgraph assigned to \( M_i \) by the replacement operator) and then recoloring every vertex \( v \) as the triple \( (\chi(v), L, \text{Inv}(M_v)) \), where \( \chi(v) \) is the color of \( v \) after the replacement, \( L \) is a list of the colors of vertices with the same adjacency type as \( v \), and \( \text{Inv}(M_v) \) is the invariant of the module containing \( v \). Intuitively the goal of this coloring is to provide information how to reconstruct the original graph. However, it is not necessarily possible to retrace which vertices belong together, i.e., which vertices originated from the same module. This is resolved in the lemma below.
We say that the decomposition functor is *simple* with respect to a replacement operator if for every complete invariant every quotient graph is prime. Intuitively, this means that the decomposition functor provides us with maximal modules.

**Definition 6** Given a decomposition functor, we say a replacement operator is *reversible* if the following holds: two graphs $G_1$ and $G_2$ are isomorphic if and only if their colored quotient graphs with respect to the decomposition functor and the replacement operator are isomorphic.

Note that reversibility does not depend on the complete invariant that is used for the recoloring of the quotient graph.

We remark that for uncolored graphs the definitions of module, primality and the quotient graph coincide with the usual definition from the literature (see [21]). In that context, the decomposition functor is typically chosen to partition the graph into components, components of the complement graph, or maximal modules. The replacement operator simply replaces the entire module by one vertex. However, for the applications we have in mind, we require the more general concept of colored modules. Certain conditions immediately imply that a replacement operator is reversible.

**Lemma 1** A replacement operator is reversible (with respect to some graph class $C$) if one of the following situations happens (for all graphs in $C$).

1. There is only one trivial module,
2. all replacements contain only one vertex, or
3. non-trivial modules induce connected graphs, but the non-trivial modules are pairwise non-adjacent.

**Proof** To reconstruct the original graph we need to replace each set $M'$ that is a replacement for module $M$ by the original module $M$. We also need to determine the adjacency of the vertices in $M$ to vertices outside of $M$. This adjacency information is encoded in the coloring of the vertices of $M'$ by definition. However, it is not possible for all replacements to determine the sets $M'$ that are replacements of modules.

However, in each of the three cases in the lemma, it is possible to reconstruct the sets of vertices $M'$ that originated from the same module. This is obvious for the first two cases. In the third case the sets $M'$ originating form non-trivial modules are the maximal connected subgraphs consisting of vertices whose third entry in the color, which is $\text{Inv}(M_v)$ by definition of the quotient graph, is the same.

In other words, the reason for the reversibility is that in each of these cases it is possible to reconstruct which replacement vertices originate from the same module.

A reversible simple decomposition functor can be used to test isomorphism by considering only isomorphisms between quotient graphs and modules. Iterating this, we will now develop an algorithm that performs an isomorphism test for decomposable graphs if one has access to a complete invariant for prime graphs.
Algorithm 1 Inv: a complete invariant for a hereditary class \( C \) with decomposition functor

**Input:** \((G, \text{Mod}, R, \text{Inv}_p)\): A graph \( G \) in \( C \), a simple decomposition functor \( \text{Mod} \), a reversible replacement operator \( R \), and an algorithm \( \text{Inv}_p \) that computes a complete invariant for graphs in \( C \) which are colored and prime with respect to \( \text{Mod} \).

**Output:** \( \text{Inv}(G) \): The value of \( G \) with respect to a complete invariant \( \text{Inv} \) for \( C \).

1: \( Q \leftarrow G \)
2: if \( Q \) is not prime then
3: \hspace{1em} Compute \( \text{Mod}(Q) \).
4: \hspace{1em} for all non-trivial \( M \in \text{Mod}(G) \) do
5: \hspace{2em} if \( M \) is prime then
6: \hspace{3em} Compute \( \text{Inv}_p(M) \).
7: \hspace{2em} else
8: \hspace{3em} Recursively compute \( \text{Inv}(M, \text{Mod}, \text{Inv}_p) \).
9: \hspace{1em} end if
10: end for
11: Compute the quotient graph \( Q' \) of \( G \) with respect to \( \text{Mod}, R \), and the computed invariants \( \text{Inv}_p(M) \).
12: \( Q \leftarrow Q' \)
13: end if
14: return \( \text{Inv}_p(Q) \)

*(Description of Algorithm 1)* Given a reversible, simple decomposition functor \( \text{Mod} \) and an algorithm \( \text{Inv}_p \) that computes a complete invariant for graphs in a graph class \( C \) that are prime with respect to \( \text{Mod} \), Algorithm 1 constitutes a complete invariant for graphs in \( C \). The algorithm applies the functor \( \text{Mod} \). If this application yields non-trivial modules, the algorithm recursively computes the invariant of the modules and forms the quotient graph. The invariant of the input is the invariant of this prime graph.

**Lemma 2** Let \( C \) be a hereditary graph class and \( \text{Mod} \) a simple polynomial-time computable decomposition functor for \( C \) with polynomial-time computable reversible replacement operator \( R \). Given a complete invariant \( \text{Inv}_p \) for colored prime graphs in \( C \), Algorithm 1 constitutes a complete invariant for all graphs in \( C \). If the total running time for query calls to \( \text{Inv}_p \) is polynomially bounded, then Algorithm 1 runs in polynomial time.

**Proof (Correctness)** First note that whenever a recursive call is performed in Line 8, the module \( M \), which is the new input graph, has less vertices than the original graph. Thus the algorithm terminates. Suppose \( G_1 \) and \( G_2 \) are isomorphic graphs to which the algorithm is applied. By definition, the decomposition functor is invariant, and thus, by induction, the recursive calls compute the same invariant for corresponding modules. Since decomposition functors are isomorphism invariant, the quotient graphs are isomorphic. These observations show that the invariants computed for \( G_1 \)
and $G_2$ are the same. Suppose now that two non-isomorphic graphs $G_1$ and $G_2$ are given to Algorithm 1. By induction we can assume that the invariants for the modules form a complete invariant for the modules. Therefore, the correctness of the algorithm follows from the assumption that $R$ is reversible.

(Running time) We show that, given a complete invariant for prime graphs that runs in polynomial time, Algorithm 1 also runs in polynomial time. Consider the recursion tree of a call to Algorithm 1 for a graph $G$ of size $n$. Since all steps can be performed in polynomial time, it suffices to polynomially bound the number of nodes in the tree. The number of vertices for each node is smaller than the number of vertices of the parent. This bounds the height of the recursion tree, and thus it suffices for us to bound the number of leaves of the tree. With the possible exception of the root, for each recursive call corresponding to a leaf, the input graph is a non-trivial module $M$ of some coloring of $G$ and thus in particular contains two indistinguishable vertices. Since at least one of these vertices is not present in any quotient graph that has a module that contains $M$, this pair of vertices cannot be a pair of indistinguishable vertices in any other call corresponding to a leaf. The number of leaves is thus bounded by the number of pairs of vertices.

Using Theorem 8 we can replace the requirement for a complete invariant by an isomorphism algorithm.

**Theorem 9** Let $C$ be a hereditary graph class and Mod a simple polynomial time computable decomposition functor with polynomial-time computable, reversible replacement operator $R$ for colored graphs in $C$. If the isomorphism problem for colored prime graphs in $C$ can be solved in polynomial time then the isomorphism problem of all graphs in $C$ can be solved in polynomial time.

**Proof** The theorem follows by combining Lemma 2 and Theorem 8. However, there is an issue with the length of the encoding of colors. In particular, if the calculated invariants are repeatedly applied to recolor the graphs then, even with a polynomial number of iterations, the encoding of the colors may become super-polynomial. To avoid this, we apply the standard technique of hashing colors. More precisely, we maintain a numbered list of colors. Every time a new color is encountered, the next unused number is assigned to this color. When an oracle call is performed by Algorithm 1, and consequentially according to Theorem 8 the algorithm for prime graphs is invoked to simulate this oracle call, then the graph will be given in encoded form with short descriptions for the colors. Since only polynomially many colors can appear during the execution of the algorithm, this shows that the entire algorithm runs in polynomial time.

The theorem in particular applies to the standard uncolored modular decomposition. Indeed, this decomposition is associated with a simple decomposition functor with polynomial-time computable, reversible replacement. In his diploma thesis, Fuhlbrück [14] also describes a reduction for the standard uncolored decomposition functor. Already when trying to solve the isomorphism problem for uncolored graphs in a graph class $C$ with the described method, the fact that the quotient graph needs to be colored implies that we require an isomorphism algorithm for colored prime
graphs. In [33], Rao describes a special case of the theorem for the uncolored modular decomposition, essentially considering hereditary graph classes in which every prime graph is of bounded size. In principle the algorithm described in Rao’s paper is a special case of the algorithm described here, but this is disguised slightly by the fact that in Rao’s description the algorithm simultaneously performs the computation of the (uncolored) modular decomposition, the invariant for the possible prime graphs, and the hashing of colors. Moreover, in the isomorphism context, the bi-join decomposition, also described in [33], can also be treated by using colored modules. In different terminology, the graph classes considered by Rao are sometimes called graphs of bounded composition-width or modular width. However, we caution the reader that there are two unrelated definitions of composition width for groups and graphs, respectively, both appearing in the context of isomorphism (see [5]).

Together with the observation that vertices of degree 1 can be removed in an isomorphism invariant fashion by recoloring their neighbors, the previous theorem also provides means of solving isomorphism of distance hereditary graphs. Indeed, by a theorem of Bandelt and Mulder [7] every prime distance hereditary graph has a vertex of degree at most 1. However, there is a faster, tailored algorithm for distance hereditary graphs running in linear time [31].

In our applications of Theorem 9 we also use another unrelated technique of dealing with small color classes that we describe next.

7 Bounded Generalized Color Valence

In this section we show that isomorphism of graphs of bounded generalized color valence can be solved in polynomial time. However, the proofs of the lemmas and theorems within the section require familiarity with the computational group theoretic methods that have been developed within the context of the isomorphism problem. For example, when computing automorphism groups, cosets, and sets of isomorphisms between two combinatorial objects, we use a succinct representation by generators and a representative. We also use the set stabilizer theorem [29] for groups with composition factors of bounded size (see also [1, 5, 30]), which implies that for a permutation group with composition factors of bounded size acting faithfully on a set, we can compute the stabilizer of any given subset in polynomial time. For a good overview over the various computational group theoretic techniques, we refer the reader to [36].

Before we solve the isomorphism problem of graphs of bounded generalized color valence, let us consider some examples. Hypergraphs on sets of bounded color class size can directly be encoded as graphs by adding a vertex for every hyperedge that is adjacent to the elements of the hyperedge. Thus, a polynomial time algorithm for bounded generalized color valence also gives rise to a polynomial time algorithm for hypergraphs of bounded color class size. However, there are examples that are not captured by this. Consider a graph of maximum degree at most $c$ consisting of a large number of components such that there is only a small number of isomorphism types among these components. Now add an arbitrary finite number of new
vertices colored with new colors such that the added vertices form color classes of size at most $c$. The new vertices are connected via edges in an arbitrary way to the original vertices. The resulting graph has bounded generalized color valence. However, its automorphism group may contain composition factors of arbitrarily large size. The isomorphism algorithm for graphs of bounded degree exploits the fact that at least for components it is possible, by individualizing one vertex, to obtain a group with bounded composition factor. However in the example graphs just described, it is not clear how the standard group theoretic arguments can be applied.

Nevertheless, our goal is to prove that graph isomorphism of graphs with bounded generalized color valence can be solved in polynomial time. Note that the classes of bounded generalized color valence are closed under refinement operations such as individualization and naive vertex refinement.

To solve the isomorphism problem, we need to deal with the small color classes. If there are only a bounded number of them, we can apply individualization to all vertices in small color classes. However, the number of small color classes can grow linearly in the number of vertices. To remedy this problem, we exploit the existence of certain colored modules and use group theoretic techniques.

We first define a decomposition functor that works well with graphs of bounded generalized color valence that have been refined with naive vertex refinement. Given a subset $S$ of vertices, we say a vertex $v' \notin S$ contained in the color class $C'$ is of degree dependence at most $d$ with respect to $S$ if there is a vertex $v \in S$ such that $v' \in N(v) \land |N(v) \cap C'| \leq d$ or $v' \notin N(v) \land |C' \setminus (N(v))| \leq d$. Intuitively, this definition says that there is a vertex $v \in S$ such that individualization of this vertex followed by refinement with respect to adjacency towards $v$ produces a set of size at most $d$ within which $v'$ can be found.

**Definition 7** A non-empty subset $S$ is a $d$-degree dependence module, if no vertex outside $S$ has degree dependence at most $d$ with respect to $S$.

**Lemma 3** Let $G$ be a graph of color valence at most $c$ which is stable under naive vertex refinement. If $G$ does not contain color classes of size at most $2c$ then the minimal $c$-degree dependence modules partition $G$. Moreover, the map assigning every such graph the family of minimal $c$-degree dependence modules is a polynomial-time computable decomposition functor.

**Proof** To show the first part of the lemma it suffices to show that the following holds in $G$: if $v$ and $v'$ are vertices in the color classes $C$ and $C'$ respectively then $|(N(v) \cap C')| \leq c$ if and only if $|(N(v') \cap C)| \leq c$ and also $|C' \setminus (N(v))| \leq c$ if and only if $|C \setminus (N(v'))| \leq c$. We only show the first equivalence, the second one follows by considering the complement graph. For the first equivalence, by symmetry, it suffices to show one direction. Thus we assume $|(N(v) \cap C')| \leq c$. If it is not the case that $|(N(v') \cap C)| \leq c$ then $|C \setminus (N(v'))| \leq c$ since the graph has color valence at most $c$. We count the number of edges and non-edges between $C$ and $C'$. There are at most $|C|c$ edges and at most $|C'|c$ non-edges. This implies that $|C||C'| \leq |C|c + |C'|c$ which shows that one of the color classes has size at most $2c$. 

\[ \square \]
The fact that the minimal $c$-degree dependence modules are isomorphism invariant follows from the fact that the definition of the modules is isomorphism invariant. They form colored modules since the color valence of the graph is assumed to be at most $c$. To compute the modules in polynomial time one can, for every vertex, compute its module by repeatedly adding vertices that have color dependence at most $c$ with respect to the set created so far.

In Lemma 3, using naive vertex refinement is essential. Indeed, consider a wheel, i.e., a cycle with an added center adjacent to every other vertex. In this graph, there is only one non-trivial 3-degree dependence module, namely the set that contains only the center of the wheel. Thus the set of 3-degree dependence modules does not partition the graph.

We show now that the set of isomorphisms between subgraphs induced by certain $c$-degree dependence modules can be computed in polynomial time.

**Lemma 4** Let $c$ be a positive integer. Let $G_1$ and $G_2$ be colored graphs with vertices $v_1$ and $v_2$. For $i \in \{1, 2\}$ let $M_i$ be a $c$-degree dependence module that is minimal among the $c$-degree dependence modules containing $v_i$. Given $M_1$ and $M_2$, we can in polynomial time compute all isomorphisms from $G_1[M_1]$ to $G_2[M_2]$ (as a coset of a group).

**Proof** For $i \in \{1, 2\}$, by iterating over all vertices in $M_i$, we can find all vertices $v_i$ such that $M_i$ is minimal among all $c$-degree dependence modules containing $v_i$. Since there are only quadratically many pairs of such vertices, it thus suffices to be able to compute all isomorphisms from $M_1$ to $M_2$ mapping $v_1$ to $v_2$ assuming that $v_1$ and $v_2$ are given. We individualize $v_i$ in each graph (with the same color in both graphs) and apply naive vertex refinement. We now consider the graphs $G_1[M_1]$ and $G_2[M_2]$. If the color classes that appear are not the same then there is no isomorphism from $G_1[M_1]$ to $G_2[M_2]$, so we assume otherwise. Note that we can order the color classes in a canonical way to $C_1, \ldots, C_t$ such that $C_1 = \{v_1, v_2\}$ and such that for $i \in \{2, \ldots, t\}$ and for vertices in $C_i$ there is a color class $C_j$ with $j < i$ such that each vertex in $C_j$ has at most $c$ neighbors or at most $c$ non-neighbors in $C_i$. This enables us to compute the set of isomorphisms between the graphs induced by the first $i$ color classes on the basis of the isomorphisms between the graphs induced by the first $i - 1$ color classes. Indeed, this shows that the graphs have automorphism groups that have composition factors that are subgroups of $S_c$, the symmetric group of degree $c$. Therefore we can apply the known techniques based on group theoretic isomorphism algorithms. For example we can apply techniques by Babai and Luks [5] or Miller [30]. We refer the reader to [5, Section 4.2] for an explanation of all these options.

We next provide a corollary of Lemma 4 that will facilitate its application in our main theorem.

**Corollary 2** Let $G_1$ and $G_2$ be graphs of generalized color valence at most $c$. Let $V_1$ and $V_2$ be the set of vertices contained in a color class of size at most $2c$ in $G_1$ and $G_2$.
respectively. Let $M_1$ and $M'_1$ be minimal $c$-degree dependence modules of $G_1 \setminus V_1$ and let $M_2$ be a minimal $c$-degree dependence module of $G_2 \setminus V_2$. We can compute in polynomial time the automorphism group of $G[M_1 \cup M'_1 \cup V]$ and the automorphism group of $G_1[M_1 \cup V_1] \cup G_2[M_2 \cup V_2]$ preserving $G_1[M_1 \cup V_1]$ and $G_2[M_2 \cup V_2]$ as blocks.

Proof The corollary follows from Lemma 4 by observing that $M_1 \cup V_1$, $M'_1 \cup V_1$ and $M_2 \cup V_2$ are contained in minimal $2c$-degree dependence modules of their respective graphs.

In the following we will be working on two graphs simultaneously. Note that when performing naive vertex refinement on two graphs separately, it may be the case that the color degrees differ in the two graphs even within the same color class. To avoid this, we perform a joint refinement, by performing the refinement on the disjoint union of the two graphs.

Lemma 5 Let $G_1$ and $G_2$ be colored graphs, stable under the joint vertex refinement, which are of color valence at most $c$ and which do not contain any color class of size at most $2c$. If $\phi$ is a map from $V(G_1)$ to $V(G_2)$ that maps the $c$-degree dependence modules of $G_1$ isomorphically to the $c$-degree dependence modules of $G_2$ then $\phi$ is an isomorphism.

Proof It suffices to show the following: if two vertices $v$ and $v'$ of $G_1$ which are not contained in the same $c$-degree dependence module are adjacent (non-adjacent) then $\phi(v)$ and $\phi(v')$ are adjacent (non-adjacent). Let $C$ be the color class of $v'$. Since $G_1$ has color valence at most $c$ the vertex $v$ has at most $c$ neighbors in $C$ or at most $c$ non-neighbors in $C$. In the former case all neighbors of $v$ of color $C$ are in the same $c$-degree dependence module as $v$. In the latter case all non-neighbors of $v$ are in the same $c$-degree dependence module as $v$. This implies that all vertices of $C$ that are not in the same $c$-degree dependence module have the same adjacency type towards $v$. This is also the case in $G_2$ and this adjacency type is the same in the graphs $G_1$ and $G_2$, since they have been jointly refined. This shows that $\phi$ is an isomorphism.

The previous proof implicitly shows that for graphs of color valence at most $c$ that are stable under naive vertex refinement which do not contain color classes of size at most $2c$ the $c$-degree dependence modules are in fact colored modules in the sense of the previous section, also providing the motivation for the terminology.

A frequent observation in the graph isomorphism context is that the isomorphism problem between two graphs reduces to computation of the automorphism group of a combination of the two graphs. We will also take this route, since it drastically streamlines our proof. However, this comes at the expense of increased running time. Note that in our case, for the reduction, we cannot simply take the usual path of forming the disjoint union of two graphs and compute its automorphism group, since this union might not have bounded generalized color valence as can be seen when forming the disjoint union of two large cliques.
Lemma 6  The isomorphism problem of graphs of generalized bounded color valence at most \( c \) polynomial time reduces to the computation of the automorphism group of a graph of generalized color valence at most \( 2c \).

Proof  Let \( G_1 \) and \( G_2 \) be colored graphs of generalized color valence at most \( c \). To each graph \( G_i \), we add a vertex \( v_i \) adjacent to all vertices of this graph. The two added vertices obtain a color that neither appears in \( G_1 \) nor on \( G_2 \). Next, we run the joint naive vertex refinement on each of the augmented graphs to obtain the graphs \( G'_1 \) and \( G'_2 \). If the obtained graphs do not contain the same color classes with the same multiplicities we reject the graphs as non-isomorphic. We form their disjoint union. Let \( C \) and \( C' \) be color classes each of multiplicity more than \( 2c \) in each of the graphs \( G'_1 \) and \( G'_2 \). We now determine whether we should add all edges between vertices of these colors which are in different graphs or none. If within the graphs \( G'_1 \) and \( G'_2 \) vertices in \( C \) have more than \( c \) neighbors in \( C' \) then we add all edges between vertices of colors \( C \) and \( C' \) if they lie in different graphs. Note that in this case, by arguments similar to those performed in the proof of Lemma 3, vertices in \( C' \) symmetrically then also have more than \( c \) neighbors in \( C \). If this operation is performed across all pairs of color classes which are larger than \( 2c \), we obtain a graph \( G \) of bounded color valence at most \( 2c \). Since we added the universal vertices \( v_1 \) and \( v_2 \) to each of the graphs, any automorphism of the final graph must respect \( V(G_1) \) and \( V(G_2) \) as blocks. Thus, \( G \) contains an automorphism that swaps \( v_1 \) and \( v_2 \) if and only if \( G_1 \) and \( G_2 \) are isomorphic.  

The lemma allows us to focus on computing the automorphism group of a single graph.

Theorem 10  For all positive integers \( c \), the automorphism group of a graph of generalized color valence at most \( c \) can be computed in polynomial time.

Proof  Let \( G \) be a colored graph of generalized color valence at most \( c \). Since refinement does not increase generalized color valence, we can assume that the coloring of the graph is stable under naive vertex refinement. Let \( U \) be the set of vertices contained in a color class of size at most \( 2c \). Let \( V \) be the set of minimal \( c \)-degree dependence modules of \( G - U \).

Suppose \( S \) and \( S' \) are subsets of \( U \). We say two \( c \)-degree dependence modules \( M \) and \( M' \) in \( V \) are equivalent relative to \( (S, S') \) if there is an isomorphism from \( G[M \cup S \cup S'] \) to \( G[M' \cup S \cup S'] \) that fixes all vertices of \( S \). We write \( M \sim_{(S, S')} M' \) if \( M \) and \( M' \) are equivalent in this way and denote by \( [M]_{\sim_{(S, S')}} \) the equivalence class of \( M \). Equivalence of modules relative to \( \sim_{(S, S')} \) and thus computation of the equivalence class can be performed in polynomial time by Corollary 2.

Let \( S \) be a union of color classes in \( U \). By \( \text{Aut}_S \) we denote the following automorphism group: we consider all permutations of \( S \). Such a permutation \( \phi \) is in \( \text{Aut}_S \) if there exists a permutation \( \pi \) of the \( c \)-degree dependence modules of \( G - U \) such that the following holds for all \( M \in V \): if \( \pi(M) = M' \) then there exists an isomorphism from \( G[M \cup U] \) to \( G[M' \cup U] \) which, when restricted to \( S \), agrees with \( \phi \).

We will first show how to compute \( \text{Aut}_U \) and then explain how to compute the automorphism group of \( G \) from \( \text{Aut}_U \). Note that \( \text{Aut}_\emptyset \) is the trivial group.
Consequently, to compute $\text{Aut}_U$, it suffices for us to show the following claim for an arbitrary color class $C$ in $U$ not contained in $S$.

**Claim.** Given $\text{Aut}_S$ we can in polynomial time compute $\text{Aut}_{S\cup C}$.

Every element $\psi' \in \text{Aut}_S$ gives rise to exactly one permutation $\overline{\psi'}$ of the equivalence classes induced by $\sim_{(S, U \setminus S)}$. This gives rise to an action of $\text{Aut}_S$ on the equivalence classes. We first compute a subgroup $\text{Aut}_S'$ of $\text{Aut}_S$ that respects a certain coloring of these equivalence classes. For this consider the equivalence classes induced by $\sim_{(S, U \setminus (S \cup C))}$. They form a refinement of the classes induced by $\sim_{(S, U \setminus S)}$. We label each equivalence class $M$ under $\sim_{(S, U \setminus S)}$ by the multiset of block-sizes that arise when $M$ is partitioned according to $\sim_{(S \cup C, U \setminus (S \cup C))}$.

We now define $\text{Aut}_S'$ to be the subgroup of $\text{Aut}_S$ that stabilizes the labels. This subgroup can be computed in polynomial time since the involved groups have composition factors of bounded size (see [5]).

Our strategy now for computing $\text{Aut}_{S \cup C}$ on the basis of $\text{Aut}_S'$ is to find a set on which a supergroup of $\text{Aut}_S'$ acts such that $\text{Aut}_{S \cup C}$ can be observed to be a stabilizer of a computable subset. Let $\mathcal{M}$ be the sets of equivalence classes in $V$ with respect to $\sim_{(S \cup C, U \setminus (S \cup C))}$. For each $[M]_{\sim_{(S \cup C, U \setminus (S \cup C))}}$ in $\mathcal{M}$ we define the set of pairs $([\phi, [M]]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ where $\phi$ is a permutation of $C$. We say two pairs $([\phi, [M]]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ and $([\phi', [M']]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ are congruent if $|[M]_{\sim_{(S \cup C, U \setminus (S \cup C))}}| = |[M']_{\sim_{(S \cup C, U \setminus (S \cup C))}}|$ and there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ which fixes $S$ and is equal to $\phi'^{-1} \phi$ on $C$. This definition is independent of the chosen representatives $M$ and $M'$ and thus congruence is well defined. Moreover, congruence is an equivalence relation. The classes of $\mathcal{M}$ embed into the congruence classes of pairs by taking $\phi$ to be the identity.

We define an action of the group $\text{Sym}(C) \times \text{Aut}_S'$ on these equivalence classes of pairs as follows. The element $(\psi, \psi')$ maps the congruence class of $([\phi, [M]]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ to the congruence class of $(\psi \phi f, [M']_{\sim_{(S \cup C, U \setminus (S \cup C))}})$, where the representative $M'$ is chosen such that $\overline{\psi'}([M]_{\sim_{(S, U \setminus S)}}) = [M']_{\sim_{(S, U \setminus S)}}$ and $|[M]_{\sim_{(S \cup C, U \setminus (S \cup C))}}| = |[M']_{\sim_{(S \cup C, U \setminus (S \cup C))}}|$ and $f$ is chosen as a permutation of $C$ such that there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ which is equal to $f'^{-1}$ and $\psi'$ on $C$ and $S$, respectively. We need to argue that this action is well defined. First, observe that since $\text{Aut}_S'$ is obtained by stabilizing the labeling as defined above, an $M'$ fulfilling the requirements exists. Suppose now that $M''$ and $f''$ also satisfy the required properties. Then there exists an isomorphism from $G[M' \cup U]$ to $G[M'' \cup U]$ that is equal to $\psi'(\psi'^{-1}) = id$ on $S$ and equal to $f''^{-1} f = (\psi \phi f')^{-1}(\psi \phi)$ on $C$. Thus the pairs $(\pi' \psi \phi, [M']_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ and $(\pi' \psi \phi, [M'']_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ are congruent. We also need to show that we have defined an action. Suppose the pair $(\overline{\psi}, \overline{\psi'})$ maps the congruence class of $(\psi \phi f, [M']_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ to the congruence class of $(\overline{\psi} \psi \phi f^{\overline{f}}, [\overline{M}]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$ with $\overline{M}$ and $\overline{f}$ satisfying the requirements as above. Since there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ which is equal to $\pi \psi$ and $\psi'$ on $C$ and $S$ respectively and there is an isomorphism from $G[M' \cup U]$ to $G[M' \cup U]$ which is equal to $\overline{f} \overline{\psi}$ and $\overline{\psi'}$ on $C$ and $S$ respectively, there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ which is equal to $\overline{\pi \psi} \overline{\psi'}$ on $C$ and $S$ respectively. It also follows from the definition that the neutral element of $\text{Sym}(C) \times \text{Aut}_S'$ fixes all pairs $([\phi, [M]]_{\sim_{(S \cup C, U \setminus (S \cup C))}})$. Thus, we have defined an action.
Abusing notation, we can consider the group of color preserving permutations of $\text{Sym}(S \cup C)$ as a subgroup of $\text{Sym}(S) \times \text{Sym}(C)$. In this sense, we claim that the group $\text{Aut}_{S \cup C}$ is the subgroup of $\text{Sym}(C) \times \text{Aut}'_S$ that stabilizes the image of $\mathcal{M}$ under the above-mentioned embedding $[M] \sim_{(S \cup C \cup \{S \cup C\})} \mapsto (\text{id}, [M] \sim_{(S \cup C \cup \{S \cup C\})})$. Suppose first that $(\psi, \psi')$ is an element of this stabilizer subgroup. We use the action on the congruence classes to show that $(\psi, \psi')$ is an isomorphism from $G$ to $\text{Aut}_{S \cup C}$. We define a bijection on the $c$-degree dependence modules contained in $[M] \sim_{(S \cup C \cup \{S \cup C\})}$ by mapping the elements of every class $[M] \sim_{(S \cup C \cup \{S \cup C\})}$ with an arbitrary bijection to the elements of $[M'] \sim_{(S \cup C \cup \{S \cup C\})}$, where the class $[M'] \sim_{(S \cup C \cup \{S \cup C\})}$ is chosen such that the pair $(\text{id}, [M'] \sim_{(S \cup C \cup \{S \cup C\})})$ is the image of $(\text{id}, [M] \sim_{(S \cup C \cup \{S \cup C\})})$ under $(\psi, \psi')$. Since $M$ and $M'$ are arbitrary, it suffices now to observe that there is an isomorphism from $M$ to $M'$ which is equal to $\psi$ and $\psi'$ on $C$ and $S$ respectively. Conversely, if $(\psi, \psi')$ is an element of $\text{Aut}_{S \cup C}$ then $\psi'$ is an element of $\text{Aut}_S$ by definition of the groups $\text{Aut}_S$ and $\text{Aut}_{S \cup C}$. Since it must respect multiplicities it is thus also an element of $\text{Aut}'_S$. Let $M$ be an arbitrary $c$-degree dependence module in $\mathcal{V}$. By definition of $\text{Aut}_{S \cup C}$ there is an $M' \in \mathcal{V}$ and an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ that agrees with $(\psi, \psi')$ on $S \cup C$. This implies that under the action of $\text{Sym}(C) \times \text{Aut}'_S$ the pair $(\text{id}, [M] \sim_{(S \cup C \cup \{S \cup C\})})$ is mapped to $(\psi \text{id}\psi'^{-1}, [M'] \sim_{(S \cup C \cup \{S \cup C\})}) = (\text{id}, [M'] \sim_{(S \cup C \cup \{S \cup C\})})$. This shows that elements of $\text{Aut}_{S \cup C}$ stabilize the image of $\mathcal{M}$ under the embedding described above. We want to compute this stabilizer. To achieve that the action is faithful, we additionally let the group act on the set $A \cup C$. Since the group has bounded composition factors, the stabilizer can be computed in polynomial time. This proves the claim.

It remains to explain how to compute the automorphism group of $G$ from the group $\text{Aut}_U$. Consider the homomorphism $\pi: \text{Aut}(G) \rightarrow \text{Sym}(U)$ that is obtained by restricting the permutations in $\text{Aut}(G)$ to the set $U$. We first argue that $\pi(\text{Aut}(G)) \leq \text{Aut}_U$. To see this consider an automorphism $\alpha$ of $G$. Since the $c$-degree dependence modules of $G - U$ are isomorphism invariant, the automorphism $\alpha$ maps every such module $M$ to a module $M'$ such that there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ that is equal to $\pi(\alpha)$ when restricted to $U$, showing $\pi(\text{Aut}(G)) \leq \text{Aut}_U$. We next argue that $\pi$ is surjective. Let $\psi$ be an element of $\text{Aut}_U$. By definition of $\text{Aut}_U$ there is a bijection from $\mathcal{V}$ to $\mathcal{V}$ such that if $M$ is mapped to $M'$ then there is an isomorphism from $G[M \cup U]$ to $G[M' \cup U]$ that is equal to $\psi$ when restricted to $U$. By Lemma 5, combining all these mappings gives an automorphism of $G$ that is equal to $\psi$ when restricted to $U$. This shows that $\pi$ is surjective. It also shows how to compute a lift of $\psi$ to $\text{Aut}(G)$ such that this lift projects to $\psi$ under $\text{Aut}(G)$. Using Corollary 2 such a lift can be computed in polynomial time. By the first isomorphism theorem, it now suffices to compute a lift for each generator in a generating set of $\text{Aut}_U$ and to compute the kernel of the map $\pi$. The kernel of $\pi$ is by definition the point-wise stabilizer of $U$. This is equal to the automorphism group of a graph obtained from $G$ by individualizing all elements of $S$. Refining such a graph with naive vertex refinement yields a graph of bounded color valence, for which we already know how to compute the automorphism group in polynomial time.

By defining a faithful group action of the automorphism group of a graph on the $c$-degree dependence modules it is possible to gradually compute the automorphism group, yielding also an isomorphism test for such graphs.
Theorem 11 Graph isomorphism for colored graphs of generalized color valence at most \( c \) can be solved in polynomial time.

Proof This follows directly from the previous theorem and Lemma 6.

In Sections 8–11 which follow next, we apply Theorem 11 together with Theorem 9 in order to show that the classes \((K_{1,s} \cup K_{1,s}, K_t)\)-free, \((P_5, K_t)\)-free, \((H(1, 0, b, 1), K_3)\)-free, and \((H(1, b, 0), K_t)\)-free have a polynomial-time solvable isomorphism problem. These classes include various subcases for which the complexity of the isomorphism problem was also previously unresolved.

8 Color Refinement and Forbidden Double Stars

In this section we are concerned with graphs which do not contain double stars. More precisely we consider classes of graphs that, for some positive integer \( s \), do not contain the disjoint union of two stars \( K_{1,s} \cup K_{1,s} \) as induced subgraphs. Our first lemma shows that bipartite graphs without double stars refine to graphs of bounded color valence under the naive vertex refinement.

Theorem 12 In a bipartite \((K_{1,s} \cup K_{1,s})\)-free graph in which the vertices are colored according to a bipartition, the naive vertex refinement produces a graph of bounded color valence. Moreover, the conclusion remains true if we only forbid double stars whose centers are in the same bipartition class.

Proof Let \( G \) be a bipartite graph without a \( K_{1,s} \cup K_{1,s} \) subgraph that has the centers in the same bipartition class. We apply the naive vertex refinement to the graph \( G \). We now show that if \( C \) and \( C' \) are two color classes then the color valence of every vertex \( v \) in \( C \) towards vertices in \( C' \) is less than \( R = 2s \). Suppose otherwise. Let \( v \) be a vertex in \( C \) and let \( N \) be the neighbors of \( v \) in \( C' \). By our assumption \( |N| \geq R \) and \( |C'| - |N| \geq R \). Since the original graph was colored according to a bipartition and the new coloring is a refinement, the vertices in \( C \) are not adjacent to \( v \). Since \( G \) is \((K_{1,s} \cup K_{1,s})\)-free, the neighborhood of a vertex in \( C \) differs from the neighborhood of \( v \) by \( s \) or more vertices. Thus, there are more than \(|C|(R - s)\) edges between \( C \) and \( N \). This implies that every vertex of \( N \) is incident with more than \(|C|/2\) edges. However, since neighborhoods cannot differ by more than \( s \) vertices, there are at most \(|C|s\) edges between \( C \) and \( C' \setminus N \). This implies that a vertex in \( C' \setminus N \) is incident with at most \(|C|s/(|C'| - |N|) \leq |C|s/R \leq |C|/2 \) edges, which gives a contradiction.

In later sections we repeatedly apply the theorem to show that isomorphism of graphs in various hereditary graph classes, among them for example \((P_5, K_t)\)-free graphs, can be solved in polynomial time. However, for some classes we also require a slight variant of Theorem 12 that also holds when the parts of the graphs are of bounded degree, as opposed to being independent sets.
Theorem 13  For every pair of positive integers $s, t$ there exists integers $c, d \in \mathbb{N}$ such that the following holds. In every $(K_1, s \cup K_1, s, K_t)$-free graph, there is a set $S$ of size at most $d$, individualization of which, followed by naive vertex refinement, produces a graph of generalized color valence at most $c$.

Proof  If the graph $G$ does not contain a star of large size, then by an easy Ramsey argument, the graph has bounded degree implying the claim of the lemma (see the introduction of [26] for details). Otherwise, we color the vertices of $G$ according to adjacency type to some large star in $G$. Note that each color class is either of bounded degree or $(K_{t-1})$-free. Since the number of color classes is bounded, by induction on $t$ we thus see that there is a set $S'$ of bounded size, such that coloring the vertices by adjacency type to $S'$ produces color classes of bounded degree. We apply the naive vertex refinement to the graph colored in this way.

We now show that if $C$ and $C'$ are two sufficiently large color classes then the color valence of every vertex $v$ in $C$ towards vertices in $C'$ is bounded by some constant $R$.

Suppose otherwise. Let $v$ be a vertex in $C$ and let $N$ be the neighbors of $c$ in $C'$. By our assumption $|N| > R$. There are $(|C| - d - 1)$ vertices in $C$ that are not adjacent to $v$. Since $G$ is $(K_1, s \cup K_1, s)$-free, there are at least $(|C| - d)(R - s)$ edges between $C$ and $N$. If we assume that $R$ is larger than $ks$ this implies that every vertex of $N$ is incident with at least $((|C| - d)(k - 1))/k$ edges with an endpoint in $C$. However, there are at most $(|C| - d)s + dR$ edges between $C$ and $C' \setminus N$. This implies that a vertex in $C' \setminus N$ is incident with at most $((|C| - d)s + d(|C' - |N|))/(|C' - |N|)) \leq (|C| - d)s/R + d \leq (|C| - d)/k + d$ edges. If $k$ and $|C|$ are sufficiently large, this gives a contradiction.

Note that the theorem and its proof are similar to Theorem 12 and its proof. However, since we are not be able to decompose our graph into color classes that are independent sets, we can only reason about color valence between large color classes.

Corollary 3  The isomorphism problem for $(K_1, s \cup K_1, s, K_t)$-free graphs can be solved in polynomial time.

Proof  The corollary follows by combining Theorem 13 and Corollary 11.

9  $P_5$-Free Graphs of Bounded Clique Number

In this section we analyze the structure of $(P_5, K_t)$-free graphs developing means to solve the isomorphism problem for such a class. We will need a theorem by Bacsó and Tuza [6] that says that every connected $(P_5)$-free graph has a dominating $P_3$ or a dominating clique. Recall that a dominating set is a set of vertices $S$ such that every vertex has a neighbor in $S$.

An instructive class of examples to have in mind are graphs obtained from disjoint unions of bipartite graphs to which a $(P_5, K_{t-3})$-free graph is joined. In this class it is easy to see that even when allowing individualization of a bounded number
of vertices, the naive vertex refinement does not produce graphs of bounded color valence. In the following we employ modules to circumvent this problem.

If \( G \) is a colored graph and \( S \) a set of vertices then a refinement of \( G \) by adjacency towards \( S \) is a coloring obtained by refining the coloring of \( G \) such that the vertices of \( S \) are singleton color classes and two vertices \( v \) and \( v' \) outside of \( S \) are in the same color class if and only if they have the same color in the original coloring of \( G \) and for every vertex of \( s \in S \) either both \( v \) and \( v' \) are adjacent to \( s \) or both are non-adjacent to \( s \).

Lemma 7 For all positive integers \( t \) and \( d \), there exists a positive integer \( \ell \) such that refining an uncolored, connected \((P_5,K_t)\)-free graph \( G \) by adjacency towards a dominating set of size \( d \) yields color classes for which the following holds: for each color \( C \), among the components in the graph induced by the color class \( C \), there are at most \( \ell \) components that do not form a colored module in the entire graph.

Proof Let \( S \) be a dominating set in an uncolored, connected graph \( G \) which is \((P_5,K_t)\)-free and let \( C \) be a color class in the graph obtained from \( G \) by refining towards adjacency. Let \( C \) be a component of the graph induced by vertices of color \( C \). If \( C \) is not a module then there exists a vertex \( v_C \notin C \) and vertices \( u_C,u'_C \in C \) such that \( v_C \) is adjacent to \( u_C \) but not adjacent to \( u'_C \). Since \( C \) is connected we can choose \( u_C \) and \( u'_C \) to be adjacent. We call the vertex \( v_C \) a distinguishing vertex for \( C \).

We first claim that a vertex \( v \notin C \) cannot be a distinguishing vertex for two distinct components \( C \) and \( C' \) at the same time (i.e., if \( C \neq C' \) then \( v_C \neq v_{C'} \)). Indeed, otherwise the vertices \( u_C,u'_C,v_C,u_{C'},v_{C'} \) would form a \( P_5 \). Since there is only a bounded number of color classes (at most \( 2^d+d \)), it suffices to show for each color class \( D \) that the number of components \( C \) in a color class \( C \) that have a distinguishing vertex of color \( D \) is bounded. Since \( G \) is \( K_t \)-free, it suffices to show that for distinct components \( C \) and \( C' \) within the graph induced by \( C \), the vertices \( v_C \) and \( v_{C'} \) are adjacent if they have the same color. Suppose this is not the case. Since \( v \) and \( v' \) have the same color, \( D \) say, and the coloring is by adjacency towards \( S \), there is a vertex \( s \) in \( S \) that distinguishes vertices in \( D \) from those in \( C \). If \( v_C \) is not adjacent to \( u_{C'} \) and \( v_{C'} \) is not adjacent to \( u_C \) then \( v_C,u_{C'},v_{C'},u_C \) and \( s \) induce a \( P_5 \), which is a contradiction. Without loss of generality we can thus assume that \( v_C \) is adjacent to \( u_{C'} \). Since \( v_C \) cannot distinguish vertices in two different components, \( v_C \) is also adjacent to \( u'_{C'} \). Since the set \( \{v_{C'},u_{C'},v_C,u_C,u'_C\} \) cannot induce a \( P_5 \), \( v_{C'} \) must be adjacent to a vertex in \( C \) and thus all vertices in \( C \). However, in this case \( v_C,u'_{C'},v_{C'},u'_{C} \) and \( s \) would induce a \( P_5 \), which shows that our assumption was false and thus \( v_C \) and \( v_{C'} \) are adjacent.

We strengthen the lemma to show that we can choose the dominating set so that the clique number in the non-module components becomes arbitrarily small.

Lemma 8 For \( n \in \mathbb{N} \setminus \{0\} \) and \( i \in \{1, \ldots, t\} \) there exists an integer \( d \) such that in every connected \((P_5,K_t)\)-free graph there is a dominating set \( S \) of size at most \( d \) such that refining by adjacency towards \( S \) yields color classes for which the following holds: for each color \( C \), every component in the graph induced by the color class \( C \) is a module or \( K_t \)-free.
Proof Suppose \( n \in \mathbb{N} \setminus \{0\}, \ i \in \{1, \ldots, t\} \) and let \( G \) be a connected graph that is \((P_5, K_n)\)-free. We show the statement by downward induction on \( i \). For \( i = t \) there is nothing to show, since the graph \( G \) is \( K_t \)-free by assumption. Suppose now that \( i < t \). By induction we may assume that for \( i + 1 \) we have already shown the statement.

Suppose \( S \) is a dominating set such that refining by adjacency towards \( S \) produces colors within which the components are modules or \( K_{t+1} \)-free. By the induction hypothesis we can choose this set to be of bounded size. By Lemma 7 there is only a bounded number of components in the graphs induced by the color classes that do not form modules. For each such component we pick a dominating set of smallest size. By the theorem of Bacsó and Tuza [6] this dominating set has bounded size. Let \( S' \) be the union of \( S \) and all dominating sets picked for the components in the color classes. The size of \( S' \) is bounded by a function of \( t \) and \( i \). We claim that refining by adjacency produces color classes which are \( K_i \)-free or modules. To see this, note that in general when refining by adjacency towards vertices outside of a module \( M \), the set \( M \) remains a module. For all components \( C \) that were not a module, the set \( S' \) used to refine by adjacency includes a dominating set within the graph induced by \( C \). Thus, for all vertices in the new non-singleton color classes which \( C \) splits into, there is a common neighbor in the dominating set \( S' \). Thus, since the component \( C \) was previously \( K_{i+1} \)-free, the new color classes are now \( K_i \)-free.

The lemma shows in fact that there is a set of bounded size such that the resulting components are all modules.

**Corollary 4** For every positive integer \( t \) there exists an integer \( d \) such that in every connected \((P_5, K_1)\)-free graph there is a dominating set \( S \) of size at most \( d \) such that refining by adjacency towards \( S \) yields color classes for which the following holds: for each color \( C \), all components in the graph induced by the color class \( C \) form a colored module in the entire graph.

**Proof** Since every \( K_1 \)-free graph is empty, the corollary is a reformulation of the previous lemma for the case \( i = 1 \).

The corollary provides us with a mechanism to decompose a \( P_5 \)-free graph into modules which have smaller clique number.

**Theorem 14** For every positive integer \( t \), graph isomorphism of \((P_5, K_1)\)-free graphs can be solved in polynomial time.

**Proof** We show the statement by induction on \( t \). Let \( G_1 \) and \( G_2 \) be graphs that are \((P_5, K_1)\)-free. By the previous corollary, we can guess in both graphs corresponding dominating sets \( S \) of bounded size such that refinement by adjacency produces color classes in which the components are modules. These modules are \( K_{t-1} \)-free since \( S \) is dominating. By induction we can compute the isomorphism type of each such module. Since the classical replacement operation is reversible, we can replace each module by a suitably colored single vertex and it suffices to solve isomorphism
of the two quotient graphs that are obtained. Observe that the color classes in the quotient graph are independent sets. We now argue that if \( C \) and \( C' \) are color classes in the quotient then there is no \( 2K_1 \) in \( C \cup C' \). Supposing otherwise, for this to happen neither \( C \) nor \( C' \) can be contained in \( S \) since vertices in \( S \) are singletons. However, since colors are obtained by refining by adjacency towards \( S \), there is a vertex \( s \in S \) that is adjacent to all vertices in one of the color classes and adjacent to no vertex of the other color class. This vertex together with the alleged copy of \( 2K_1 \) would form a \( P_5 \) yielding a contradiction. Consequently, by Theorem 12, the naive vertex refinement refines the graphs \( G_1 \) and \( G_2 \) to bounded color valence.

At this point we would like to caution the reader that for each particular graph class treated in the theorem, the modular decomposition is formed only a bounded number of times. The fact that vertices are individualized before the modules are formed prevents us from applying the Theorem 9 directly.

10 Specific Classes of Triangle-Free Graphs

We now analyze certain classes of triangle-free graphs and in doing so apply the techniques developed in the previous sections of this paper. Our ultimate goal is to show that isomorphism of triangle-free \( H(1, 0, b, 0) \)-free graphs (see Fig. 1b) can be solved in polynomial time. However, only little is known about isomorphism of these classes and even their structurally simpler subclasses. This forces us to use a form of bootstrapping by showing various structural properties and developing algorithms step by step.

Lemma 9 For every integer \( b \), the isomorphism problem of graphs that are colored, bipartite \( H(1, 0, b, 0) \)-free can be solved in polynomial time.

Proof We analyze the structure of a connected, bipartite, \( H(1, 0, b, 0) \)-free, colored graph \( G \). Let \( A \) and \( B \) be the bipartition classes of \( G \). We apply the naive vertex refinement to \( G \). By possibly having individualized one vertex before the refinement, we can assume that vertices in different bipartition classes have different colors. If \( G \) does not contain \( 2K_{1,2b-1} \) with centers in the same bipartition class, then by Theorem 12 the naive vertex refinement produces a graph of bounded color valence. Thus for such graphs, we can solve isomorphism in polynomial time. We assume therefore that in \( G \) there is an induced subgraph isomorphic to \( 2K_{1,2b-1} \) with centers in \( A \) say.

For a vertex \( v \) in \( A \) we define the set \( N_c[v] \) as the set of vertices of \( A \) obtained by exhaustively applying the following operation: we start by setting \( S = \{v\} \). If there is a vertex \( v' \) in \( A \) such that the neighborhoods of \( S \) and \( v' \) in \( B \) intersect, and the set of neighbors of \( S \) in \( B \) is not properly contained in the set of non-neighbors of \( v' \) in \( B \), then we add \( v' \) to \( S \). In this process the final set obtained is independent of the order in which the vertices are added.

This way, every vertex of \( A \) is assigned a set \( N_c[v] \). We next prove several properties that the family \( \mathcal{F} = \{N_c[v] \mid v \in A\} \) of these sets fulfills. To do so, we first observe that, by induction, for any vertex \( x \) in \( N_c[v] \) there is a sequence \( v = \ldots = x, y, z \in \mathcal{F} \) is a sequence
v_0, v_1, \ldots, v_t = x$ with $t \in \mathbb{N}$ of vertices in $N_c[v]$ such that $v_i$ and $v_{i+1}$ have a common neighbor in $B$.

We claim that the sets in the family $\mathcal{F}$ are nested, i.e., that two such sets are disjoint or one is contained in the other. Suppose $N_c[v]$ and $N_c[v']$ intersect but neither contains the other. Let $v = v_0, v_1, \ldots, v_t = v'$ be a sequence in $N_c[v] \cup N_c[v']$ such that consecutive vertices have a common neighbor in $B$. Such a sequence must exist since the sets intersect. This implies that there is an index $j$ such that $v_j$ is in $N_c[v]$ and $v_{j+1}$ is not in $N_c[v]$. Since $v_j$ and $v_{j+1}$ have a common neighbor it must be the case that the neighborhood of $v_{j+1}$ includes all neighbors of vertices in $N_c[v]$. Therefore $N_c[v]$ is a subset of $N_c[v']$, showing that $\mathcal{F}$ is nested.

For a subset $S$ of $A$, as usual, we let $N(S)$ be the set of vertices of $B$ which are adjacent to some vertex of $N$. We now argue that $N(N_c[v])$ and $N(N_c[v'])$ are disjoint, whenever $N_c[v]$ and $N_c[v']$ are disjoint. Supposing otherwise, there exists a vertex $b \in N(N_c[v]) \cap N(N_c[v'])$. This implies there are vertices $v_1 \in N_c[v]$ and $v_2 \in N_c[v']$ both adjacent to $b$. If the neighborhood of $v_1$ does not properly include all neighbors of vertices in $N_c[v']$ then $v_1$ should have been added to $N_c[v']$. Otherwise all vertices of $N_c[v']$ should have been added to $N_c[v]$. This gives a contradiction.

We also claim that if $v$ and $v'$ are the centers in $A$ in an induced subgraph isomorphic to $2K_{1,2b-1}$ then the sets $N_c[v]$ and $N_c[v']$ are disjoint. If these sets are not disjoint, there exists a sequence $v = v_0, v_1, \ldots, v_t = v'$ in $N_c[v]$ such that the neighborhoods of $v_i$ and $v_{i+1}$ intersect. Observe that, due to the absence of the forbidden subgraph $H(1, 0, b, 0)$, the neighborhoods of two vertices in $A$ that have a common neighbor differ by at most $2(b-1)$ vertices, i.e., the symmetric difference of the neighborhoods contains at most $2(b-1)$ vertices. We show by induction that $v$ and $v_i$ have a common neighbor. For $i = 0$ we have $v = v_0$ and there is nothing to show. Suppose we have shown the statement for $i$. Then all but at most $(b-1)$ neighbors of $v$ are also neighbors of $v_i$ and all but at most $(b-1)$ neighbors $v_i$ are neighbors of $v_{i+1}$. Since $v$ has more than $2(b-1)$ neighbors, this shows that $v_{i+1}$ and $v$ have a common neighbor. We conclude that $v$ and $v'$ have a common neighbor and thus their neighborhoods differ by at most $2(b-1)$. This contradicts the fact that they are centers of a double star $2K_{1,2b-1}$. And thus shows that $N_c[v]$ and $N_c[v']$ are disjoint.

Let $M$ be the maximal sets $N_c[v]$ different from $A$ for which $N(N_c[v])$ is of size at least $2b - 1$. We claim that the sets $N_c[v] \cup N(N_c[v])$ with $N_c[v] \in M$ form modules in the graph that has been color-refined by degree. Let $N_c[v]$ be a set in $M$. We first note that, by definition of $N_c[v]$, every vertex of $A$ not in $N_c[v]$ is adjacent to all vertices or to no vertex of $N(N_c[v])$, since otherwise the vertex would be part of $N_c[v]$. Suppose there is a vertex $y$ in $A$ not in $N_c[v]$ that is adjacent to all vertices of $N(N_c[v])$. This implies $N_c[v] \subseteq N_c[y]$. We claim that every vertex $y'$ of $A$ not in $N_c[v]$ with the same degree as $y$ is also adjacent to all vertices of $N(N_c[v])$. Indeed, otherwise they would form a double star $2K_{1,2b-1}$. By our observation above $N_c[y]$ and $N_c[y']$ would be disjoint, which would show that $N_c[v]$ was not maximal among the sets different from $A$. By definition every vertex of $B$ that is not in $N(N_c[v])$ is not adjacent to any vertex of $N_c[v]$. This shows that the maximal non-trivial sets are modules.

The map that associates $M$ with $G$ is a polynomial time computable decomposition functor. By our observations above, the prime graphs do not contain $2K_{1,2b-1}$
with centers in \( A \) and for these we already described at the beginning of the proof how to solve isomorphism in polynomial time. The functor has a reversible replacement operator, namely maintaining two adjacent vertices of a module, and since the modules are maximal sets, the decomposition is simple. Thus, by Theorem 9 we can solve isomorphism of bipartite, colored \( H(1, 0, b, 0) \)-free graphs in polynomial time. \( \Box \)

We can use the algorithm for the bipartite case to develop an algorithm for all graphs that are \( H(1, 0, b, 0) \)-free and do not contain triangles.

**Lemma 10** Isomorphism of \( (H(1, 0, b, 0), K_3) \)-free graphs can be solved in polynomial time.

**Proof** Let \( G \) be a colored, connected \( (H(1, 0, b, 0), K_3) \)-free graph. If the maximum degree of \( G \) is at most \( 6b \) we can solve isomorphism in polynomial time. Thus, let \( v \) be a vertex of \( G \) of degree at least \( d > 6b \) whose neighborhood is maximal, i.e., the neighborhood of no other vertex properly contains the neighborhood of \( v \). Let \( N_0 \) be the set of vertices whose neighborhood is the same as the neighborhood of \( v \). Let \( N_i \) be the set of vertices that have distance \( i \) from \( N_0 \). Note that \( N_1 \) forms an independent set since \( G \) is triangle free. We partition \( N_2 \) into three groups depending on the number of neighbors in \( N_1 \) being high, low or medium as follows: let \( N_2^H \), and \( N_2^L \) be the sets of vertices in \( N_2 \) which have at most \( c_1 = 3b \) non-neighbors and at most \( c_2 = b \) neighbors in \( N_1 \) respectively. The set \( N_2^M \) is the set of vertices in \( N_2 \) neither in \( N_2^H \) nor \( N_2^L \). Since \( c_1 + c_2 \leq d \) these three sets partition \( N_2 \).

We first argue that the set \( N_2^H \) forms an independent set. Indeed, two vertices in \( N_2^H \) have a common neighbor in \( N_1 \), since each such vertex has at most \( c_1 \) non-neighbors in \( N_1 \), but there are at least \( d > 2c_1 \) vertices in \( N_1 \).

**Claim.** We claim that if a vertex \( u_1 \) in \( N_2 \) has a neighbor \( u_2 \) that is not in \( N_1 \), then the set of neighbors of \( u_1 \) in \( N_1 \) and the set of neighbors of \( u_2 \) in \( N_1 \) form disjoint sets which contain all but at most \( b - 1 \) vertices of \( N_1 \).

The fact that the neighbor sets in \( N_1 \) are disjoint follows from the triangle-freeness of the graph. If there were \( b \) or more vertices in \( N_1 \) not adjacent to \( u_1 \) and not adjacent to \( u_2 \), we could construct the forbidden subgraph \( H(1, 0, b, 0) \) by choosing the two vertices \( u_1 \) and \( u_2 \), a vertex in \( N_1 \) adjacent to \( u_2 \) (consequently not adjacent to \( u_1 \)), one vertex in \( N_0 \) and \( b \) vertices in \( N_1 \) neither adjacent to \( u_1 \) nor to \( u_2 \). This proves the claim.

From the claim it follows that no vertex \( u_1 \) in \( N_3 \cup N_2^L \) has a neighbor \( u_2 \) in \( N_2^L \cup N_2^M \). Indeed since \( b + (c_1 - c_2) \leq d \) there are at least \( b \) vertices in \( N_1 \) not contained in the union of the neighborhoods of \( u_1 \) and \( u_2 \).

We next argue that the bipartite graph with partition classes \( N_2^H \) and \( N_2^L \) does not contain the double star \( 2K_{1,(c_1 + 1)b} \) with the centers in the same class. The centers of such an alleged double star \( J \) cannot be in \( N_2^H \) since such vertices have a common neighbor in \( N_1 \), which, together with a subgraph of \( J \) isomorphic to \( K_{1,b} \cup K_2 \), would result in the forbidden subgraph. We now assume that the centers of \( J \) are in \( N_2^L \). Symmetrically to the other case, the centers cannot have a common neighbor in \( N_1 \). Suppose \( w \) is one of the centers and there is a vertex \( u \) in \( N_1 \) adjacent to \( w \) such that at least one vertex \( y \) of \( J \) in \( N_2^L \) is adjacent to \( u \). Then we find a forbidden subgraph...
with a vertex set consisting of \( y, u, \) the two centers and \( b \) vertices of \( J \) in \( N^h_1 \) adjacent to \( w \). Supposing otherwise, let \( u \) be a neighbor of \( w \) in \( N_1 \) which is not adjacent to any vertex of \( J \) in \( N^h_1 \). Note that among any \((c_1 + 1)b\) vertices in \( N^h_2 \), there is a set \( T \) of \( b \) vertices that has a common neighbor in \( N_1 \). Thus in \( J \) we can choose a set of \( T \) vertices adjacent to a common vertex \( x \) in \( N_1 \) but such that all vertices in \( T \) are not adjacent to \( u \). The vertices in \( T \) together with \( x, v, u \) and \( w \) induce a forbidden subgraph. By Theorem 12, the absence of double stars that we just argued implies that the naive vertex refinement produces bounded color valence in the graph induced by \( N^h_2 \cup N^l_2 \).

Suppose \( N^m_2 \) is non-empty. We distinguish two cases according to the existence of an edge running within \( N^m_2 \cup N^h_2 \). Suppose first there exists an edge. Let \( N^h_2 \) be the set of vertices in \( N^h_2 \) that have a neighbor in \( N^m_2 \). We now show that \( N^m_2 \cup N^h_2 \) is a complete bipartite graph. Let \( x_1 \) and \( x_2 \) be adjacent vertices of \( N^m_2 \cup N^h_2 \). Consider a third vertex \( x_3 \) in \( N^m_2 \). It suffices to show that \( x_3 \) is adjacent to \( x_1 \) or \( x_2 \). Supposing otherwise, using the claim, we see that there is a \( c_1 \) such that \( x_3 \) is not adjacent to \( x_1 \) nor to \( x_2 \). Since \( x_3 \in N^m_2 \) is adjacent to at least \( c_2 = b \) vertices in \( N_1 \), this shows that \( x_3 \) has a common neighbor in \( N_1 \) with \( x_1 \) or \( x_2 \). Due to the forbidden subgraph, this implies that for \( i \in \{1, 2\} \) the number of neighbors of \( x_i \) which are not neighbors of \( x_3 \) is less than \( b \). This implies that the number of vertices in \( N_1 \) to which \( x_3 \) is not adjacent is less than \( c_1 = 3b \), which contradicts \( x_3 \in N^m_2 \).

By the claim, the \( N_1 \)-neighborhoods of adjacent vertices in \( N_2 \) are complementary with respect to \( N_1 \), with the exception of less than \( b \) vertices. Thus in the case that there is an edge \( x_1, x_2 \) in \( N^m_2 \) then individualizing \( x_1 \) and \( x_2 \) and refining by adjacency yields a graph for which every vertex has bounded color valence to color classes contained in \( N_1 \).

Suppose now that there is no edge within \( N^m_2 \cup N^h_2 \). Due to the forbidden subgraph, there cannot be a double star isomorphic to \( K^1_{1,b} \cup K^2_1 \) in the graph induced by \( N_1 \cup N_2 \) with centers in \( N_1 \). We show that if there is a double star \( J \) isomorphic to \( 2K^1_{1,b} \) between \( N^m_2 \) and \( N_1 \) with centers in \( N^m_2 \) then the graph is not prime with respect to a certain polynomial time computable decomposition functor. Let \( x_1 \) and \( x_2 \) be the centers in \( N^m_2 \). Suppose a vertex \( x \) in \( N_2 \) has a common neighbor with both \( x_1 \) and \( x_2 \). Note that this vertex is neither adjacent to \( x_1 \) nor to \( x_2 \). This implies it must be adjacent to all but \( b - 1 \) neighbors of \( x_1 \) and all but \( b - 1 \) neighbors of \( x_2 \). This in turn implies that it must be adjacent to all neighbors of \( x_1 \) and all neighbors of \( x_2 \) since otherwise the forbidden subgraph is formed by \( x_1, x_2, x, \) and suitable neighbors of \( x_1 \) and \( x_2 \). Due to the forbidden subgraph, if a vertex from \( N^h_2 \) has a common neighbor with \( x_1 \) then its neighborhood in \( N_1 \) is a subset of the neighborhood of \( x_1 \) in \( N_1 \). Moreover it cannot be adjacent to a vertex in \( N^h_2 \). The argument shows that the set of vertices which have a common neighbor with \( x_1 \) but not with \( x_2 \), together with their neighbors in \( N_1 \), form a colored module. The union of all such modules is also a module containing only vertices from \( N_1, N^m_2 \) and \( N^h_2 \). This module is a bipartite graph which is isomorphism invariant for the graph in which \( v \) has been individualized and can be computed in polynomial time. By Lemma 9 we can determine the isomorphism type of the module. Replacing it by an edge is reversible. Since it contains all double stars of the described type, the associated decomposition functor is
simple. For the entire graph $G$, note that, when all vertices of this module except the endpoints of one edge are removed, then the graph remains connected.

We next show for every $i > 2$ that for every vertex $v \in N_i$ the number of neighbors of $v$ in $N_i \cup N_{i+1}$ is at most $b - 1$. Indeed, suppose $z_0$ is a vertex in $N_i$ which has a set $T$ of $b$ neighbors in $N_i \cup N_{i+1}$. Using $T$ and $z_0$ we can construct the forbidden subgraph $H(1, 0, b, 0)$ by starting with $z_0$ and repeatedly choosing a neighbor in the level that is one closer to the root. More precisely, let $z_1$ be a neighbor of $z_0$ in $N_{i-1}$, let $z_2$ be a neighbor of $z_1$ in $N_{i-2}$ and let $z_3$ be a neighbor of $z_2$ in $N_{i-3}$. The set $T \cup \{z_0, z_1, z_2, z_3\}$ induces the forbidden subgraph.

Using a very similar argument, we also show that for $i = 2$ the analogous statement holds, i.e., for every vertex $v \in N_2$ the number of neighbors of $v$ in $N_2 \cup N_3$ is bounded: let $z_0$ be a vertex in $N_2$ which has a set $T$ of $b$ neighbors in $N_2 \cup N_3$. Since $N_2$ is an independent set, $T$ is in fact contained in $N_3$. We construct the forbidden subgraph by adding a vertex $z_1$ from $N_1$ adjacent to $z_0$, a vertex $z_2$ from $N_0$ and a vertex $z_3$ from $N_1$ not adjacent to $z_0$. This vertex exists since, due to the neighborhood of $v \in N_0$ being maximal, the neighborhood of $z_0$ cannot contain all of $N_1$.

We summarize what we have achieved up to this point: by choosing an initial vertex $v$ with maximal neighborhood we partition the graph into level sets $N_i$. By either removing a bipartite module in $N_1 \cup N_2^m \cup N_2^l$ or individualizing an edge in $N_2^m$ we obtain a colored, connected graph for which the naive vertex refinement produces a coloring in which every vertex has bounded color valence to every color class that is not contained in $N_2^h$.

However, between the vertices in $N_2^h$ and $N_3$ the same situation could arise as between $N_1$ and $N_2$. We cannot easily apply the previous technique to separate $N_3$ into $N_3^h$, $N_3^m$ and $N_3^l$ since the adjacency of $N_2^h$ towards $N_1$ is not uniform in the same way in which the adjacency of $N_1$ is uniform to $N_0$.

We remedy this by reapplying the entire procedure for a bounded number of iterations using different root sets $N_0$. Intuitively, in each iteration we force some of the vertices previously in $N_2^h$ to be somewhere outside of what is going to be the new level set of distance 2 from the root. For this, first note that in $N_1$ for every set $S'$ of size $c_1 + 1$, every vertex in $N_2^h$ has a neighbor in $S'$. For every vertex $w \in S'$ we can choose some maximal vertex $w'$ whose neighborhood contains the entire neighborhood of $w$. Choosing $w'$ and its twins to become the level set $N_0$ implies that vertices adjacent to $w$ will be in the new level set $N_1$ and in particular not in the new $N_2^h$. Repeating this for all vertices in $S'$ ensures that for each vertex there is at least one execution during which the vertex is not contained in $N_2^h$. If the degree of $w'$ is not sufficiently large, i.e., it is not larger than $6b$, we cannot apply the arguments given above. However, in this case individualizing all neighbors of $w'$ and refining by adjacency already achieves that adjacency for all vertices towards neighbors of $w$ is of bounded color valence. This shows that the graph we obtain has bounded color valence. In total only a bounded number of vertices is being individualized.

By applying the colored module isomorphism theorem (Theorem 9) this proves the lemma, since the modules are bipartite graphs (for which isomorphism can be solved by the previous lemma).

The previous theorems are concerned with forbidding the graph $H(1, 0, b, 0)$ which is obtained from $H(1, 0, b, 1)$ by deleting the isolated vertex. We now study
the classes obtained by deleting a different vertex from $H(1, 0, b, 1)$ and forbidding the resulting graph.

**Lemma 11** Let $s$ be a non-negative integer. Graph isomorphism of colored, bipartite graphs which do not contain an induced subgraph isomorphic to $K_{1,s} \cup K_2 \cup K_1$ such that the center of $K_{1,s}$ and the isolated vertex are in different bipartition classes can be solved in polynomial time.

**Proof** If $G$ is a connected bipartite graph then $G$ does not contain $K_{1,s} \cup K_2 \cup K_1$ with the center and the isolated vertex in different bipartition classes if and only if the bipartite complement of $G$ is $H(1, 0, s, 0)$-free. Isomorphism of such graphs can thus by solved in polynomial time due to Lemma 9 by taking the bipartite complement.

**Corollary 5** Isomorphism of connected, bipartite $H(1, 0, b, 1)$-free graphs for which every pair of vertices in the same bipartition class has a common neighbor can be solved in polynomial time.

**Proof** If a connected, bipartite graph is $H(1, 0, b, 1)$-free and every pair of vertices in the same bipartition class has a common neighbor then the graph does not contain an induced subgraph isomorphic to $K_{1,b} \cup K_2 \cup K_1$ such that the center of $K_{1,b}$ and the isolated vertex are in different bipartition classes. We can thus apply the previous lemma.

Despite being able to solve isomorphism of bipartite $H(1, 0, b, 1)$-free graphs with the corollary, we still require more detailed structural knowledge on these graph classes for our overall goal.

**Lemma 12** Let $G$ be a bipartite graph colored according to a bipartition which does not contain an induced subgraph isomorphic to $K_{1,s} \cup K_2 \cup K_1$ such that the center of $K_{1,s}$ and the isolated vertex are in different bipartition classes. If $G$ does not contain the bipartite complement of $3K_{1,s}$ with centers in the same bipartition class, then by individualizing a set of bounded size and applying the naive vertex refinement, the graph can be turned into a graph of bounded color valence.

**Proof** If the graph has no $2K_{1,s}$ with centers on the same side, then by Theorem 12 the naive vertex refinement produces a graph of bounded color valence. Otherwise, we individualize the vertices of a copy $J_1$ of $2K_{1,s}$ with centers on the same side. If in the new color classes there is no $2K_{1,s}$ with centers on the same side, we can apply Theorem 12 again. Otherwise, we individualize one new copy $J_2$ of $2K_{1,s}$ such that prior to individualization vertices in $J_2$ that lie in the same bipartition class had the same color. Note that the vertex sets of $J_1$ and $J_2$ are disjoint. By possibly repeating the argument, we conclude that we can choose $J_1$ and $J_2$ such that all centers are in the same bipartition class. Since $J_2$ was chosen after the vertices were refined by adjacency to $J_1$, to every vertex of $J_1$ the centers of $J_2$ are either both adjacent or neither of them is adjacent. This implies that both centers of $J_2$ must be adjacent to the non-centers of $J_1$, otherwise we find the forbidden subgraph since the centers
of $J_2$ would have a common non-neighbor in the other bipartition class. Similarly, there cannot be a non-center of $J_2$ to which both centers of $J_1$ are non-adjacent. This implies that one of the centers of $J_1$ is adjacent to all non-centers of $J_2$. This center and its non-adjacent non-center vertices in $J_1$ together with the vertices of $J_2$ form the bipartite complement of $3K_{1,s}$.

We assemble the structural analysis of the subcases performed in this section to develop an algorithm for isomorphism of triangle-free $H(1, 0, b, 1)$-free graphs.

**Theorem 15** The isomorphism problem for $(H(1, 0, b, 1), K_3)$-free graphs can be solved in polynomial time.

*Proof* Without loss of generality we assume $b \geq 3$. If one of the input graphs to a given instance of the graph isomorphism problem is $(H(1, 0, b, 0), K_3)$-free, then we can solve isomorphism using Lemma 10.

We thus discuss the structure of a $(H(1, 0, b, 1), K_3)$-free graph $G$ that contains an induced subgraph $K$ isomorphic to $H(1, 0, b, 0)$. Note that every vertex of $G$ is adjacent to at least one vertex of $K$, i.e., $K$ is dominating. For $i \in \{0, \ldots, 4\}$ we define the sets $N_i$ to be set of vertices of distance $i$ from the leaf in $K$ that is adjacent to a vertex of degree $2$. By individualizing the vertices in $K$ and recoloring the vertices of $G$ according to adjacency towards $K$ we obtain a coloring of the graph into independent sets.

Let $C$ and $C'$ be two color classes which contain adjacent vertices. Let $T$ be the set of neighbors in $K$ of a vertex (and thus any vertex) in $C$ and let $T'$ be the set of neighbors in $K$ of a vertex in $C'$. This implies that $T$ and $T'$ form disjoint non-empty independent sets of $K$. We want to achieve that the graph induced by two color classes $C$ and $C'$ has bounded color valence when refined with the naive vertex refinement. Note that the subgraph induced by $C$ and $C'$ does not contain $K_{1,2b+1} \cup K_2 \cup K_1$ such that the center of $K_{1,2b+1}$ and the isolated vertex are in different bipartition classes: Otherwise, by adding an appropriate vertex of $K$, we could construct the forbidden subgraph $H(1, 0, b, 1)$. Thus, if the graph induced by the two colors does not contain the bipartite of complement $3K_{1,2b+1}$ with centers in the same bipartition class, we can, by Lemma 12, individualize a set of bounded size such that the refinement yields bounded color valence. Since the bipartite complement of $3K_{1,2b+1}$ in particular contains the graph $H(1, 0, b, 0)$, the graph induced by $C$ and $C'$ can only contain the bipartite complement of $3K_{1,2b+1}$ if $T \cup T'$ contains every vertex of $K$.

In the following we thus assume that one of the sets $T$ and $T'$ is $N_0 \cup N_2 \cup N_4$ while the other one is $N_1 \cup N_3$. We also assume that the graph induced by $C$ and $C'$ contains the bipartite of complement $3K_{1,2b+1}$ with centers in the same bipartition class. Let $J$ be an isomorphic copy of this graph. Without loss of generality we assume that the centers are in the color class $C$.

We claim that $G$ is a bipartite graph such that each pair of vertices in one of the bipartition classes has a common neighbor. To show this, we first show that for each vertex $v$ not in $C$, for two of the three stars in $J$, $v$ is adjacent to more than half the vertices of color $C$ or to more than half the vertices of color $C'$. Thus let $v$ be a vertex that is not of color $C$, its color being $C''$ say. Let $T_v$ be the vertices in $K$ to which $v$ is
adjacent. Recall that $T$ is a maximal independent set of $K$. Thus, since $T_v$ is different from $T$, there is a vertex $x$ in $K$ such that $x$ is adjacent to the vertices in $C$ but neither adjacent to the vertices in $C'$ nor the vertices in $C''$. This implies, due to forbidden subgraph $H(1, 0, b, 1)$, that $v$ is adjacent to at least two centers of $J$ or for two of the stars in $J$, $v$ is adjacent to more than half of the vertices in $C'$. Since $T_v$ is non-empty, there is no edge between $v$ and $C$ or there is no edge between $v$ and $C'$.

Let $A$ be the set of vertices that have a neighbor in $C$ and let $B$ be the other vertices. We claim that this is a bipartition of $G$. Indeed, every vertex that has a neighbor in $C$ is adjacent to two of the tree centers. Thus all vertices in $A$ have a common neighbor and $A$ is an independent set. All vertices in $B$ which are not in $C$ are adjacent to more than half of the vertices of color $C'$ in two of the stars. It remains to show that vertices in $C$ have a common neighbor with all vertices in $B$. However, each vertex in $B$ is adjacent to a vertex in $T$ which in turn is adjacent to every vertex in $C$. Thus vertices in $B$ have a common neighbor and $B$ is an independent set.

This proves that $G$ is bipartite such that each pair of vertices in one of the bipartition classes has a common neighbor. Isomorphism of such graphs can be solved using Corollary 5.

11 Specific Classes of Bounded Clique Number

In this section we show that isomorphism of $(H(1, b, 0), K_s)$-free graphs can be solved in polynomial time (see Fig. 1a). To do so, we start with a structural lemma concerning such graphs.

Lemma 13 Let $G$ be an $H(1, b, 0)$-free graph and let $H$ be an induced subgraph of $G$ isomorphic to $K_{1,2b-1}$. Let $M$ be a connected induced subgraph of $G$ such that $M$ and $H$ are disjoint and non-adjacent. No vertex of $G$ simultaneously has a neighbor in $H$ and in $M$ and a non-neighbor in $M$.

Proof Suppose otherwise that $v$ has a neighbor in $H$ and $M$ and a non-neighbor in $M$. Since $M$ is connected, $v$ has a neighbor $x$ in $M$ and a non-neighbor $y$ in $M$ such that $x$ and $y$ are adjacent. If $v$ is adjacent to at least $b$ leaf vertices of $H$ then $\{y, x, v\}$ together with these $b$ leaves induces the forbidden subgraph. Otherwise $v$ is non-adjacent to $b$ leaves of $H$. If $v$ is adjacent to the center $h$ of $H$ then the set $\{x, v, h\}$ together with $b$ leaves non-adjacent to $v$ induce a forbidden subgraph. If however $v$ is non-adjacent to the center, then by assumption there is a leaf $\ell$ to which $v$ is adjacent. In this case $v, \ell, h$ and $b$ leaves non-adjacent to $v$ induce a forbidden subgraph.

Theorem 16 Graph isomorphism for colored $(H(1, b, 0), K_s)$-free graphs can be solved in polynomial time.

Proof Let $G_1$ and $G_2$ be $(H(1, b, 0), K_s)$-free graphs. Recall that by Corollary 3 we can solve isomorphism of $G_1$ and $G_2$ in polynomial time if one of the graphs is $(K_{1,2b-1} \cup K_{1,2b-1})$-free.

© Springer
We now argue that for graphs that do contain $K_{1,2b-1} \cup K_{1,2b-1}$, we can use the standard modular decomposition functor for uncolored graphs to decompose the graphs. That is, if the graphs are not connected, we take the connected components as the modules. Analogously, if their complements are not connected we take the connected components of the complements as modules. In any other case, we decompose the graph into the maximal modules. These partition the vertices and can be computed in polynomial time (see [21]). We show that if a $H(1,b,0)$-free graph contains an induced double star $K_{1,2b-1} \cup K_{1,2b-1}$ then there is a non-trivial module. For this it suffices to show that the minimal modules containing each $K_{1,2b-1}$ are disjoint. Call the stars $H_1$ and $H_2$. Suppose the minimal module containing $H_1$ contains a vertex from $H_2$. The minimal module containing $H_1$ is obtained by starting with $S_0 = V(H_1)$ and repeatedly increasing this set by adding vertices. More precisely $S_{i+1} = S_i \cup \{v\}$ for some vertex $v$ that has a neighbor and a non-neighbor in $S_i$. Note that by induction all $S_i$ are connected. By our assumptions, in some step a vertex must be added which is not in $V(H_2)$ but has a neighbor in $H_2$. Let $i$ be the first step in which this happens and suppose $u$ is the vertex that is added. Then $u$ has a neighbor and a non-neighbor in the set $S_i$ which induces a connected graph and a neighbor in $H_2$, which is a contradiction to Lemma 13.

Since we can solve isomorphism of colored prime graphs in polynomial time, this concludes the proof using Theorem 9 with the classic decomposition functor.

12 Comprehensiveness of the Case Distinction

In this section we argue that the theorems developed throughout this paper together with the theorems from [26], with the exception of finitely many cases, resolve the complexity of graph isomorphism for $(H_1, H_2)$-free graphs. In fact, said theorems also either provide a polynomial time algorithm or a reduction from the general isomorphism problem to the respective classes.

**Proof of Theorem 2** Let $(H_1, H_2)$-free be the graph class for which we want to determine the complexity. By the results in [26], we may assume that one of the graphs, $H_2$ say, is a complete graph. Since graph isomorphism for $K_2$-free graphs is solvable in polynomial time, we further assume that $H_2$ has at least 3 vertices. By [26, Lemma 2] we may consequently assume that $H_1$ is a forest of subdivided stars, since the problem is graph isomorphism complete otherwise.

If $H_1$ contains 3 non-trivial components, then $H_1$ contains $3K_2$ and the problem is isomorphism complete [26, Lemma 5]. In the following, we can thus assume that $H_1$ has at most 2 non-trivial components. If $H_1$ does not contain a vertex of degree 3 then if $H_1$ is sufficiently large, it is either of the form $P_i \cup I_i$ with $i \leq 3$ or it contains the graph $2K_2 \cup I_2$. In the former case, graph isomorphism is polynomial-time solvable by Theorem 15. In the latter case, the isomorphism problem is graph isomorphism complete [26, Lemma 5].

*The case $H_2 = K_3$*. Suppose $H_2$ is the graph $K_3$. Since $H_2$ is fixed, we can assume that $H_1$ is sufficiently large and we can thus assume by the observation above that $H_1$ contains a vertex of degree 3. If $H_1$ does not contain a $P_4$ then $H_1$ is a union
of at most two stars plus isolated vertices. If there is at most one star, the problem is polynomial-time solvable by Theorem 15 (or by [26, Theorem 4]). Assuming there are two stars, if there is more than one isolated vertex then $H_1$ contains $2K_2 \cup 2K_1$ and the problem is isomorphism complete by [26, Lemma 5]. If neither of the stars is only a single edge then if there exists an isolated vertex in $H_1$ the problem is isomorphism complete by Theorem 6 and if there is no isolated vertex in $H_1$ it is polynomial-time solvable by Corollary 3. Finally, if one of the stars is only an edge then by Theorem 15 the problem is polynomial-time solvable.

If $H_1$ contains a $P_4$ and there are two non-adjacent vertices not in the same connected component as the $P_4$ then $H_1$ contains $P_4 \cup 2K_1$ and the problem is graph isomorphism complete [26, Lemma 5]. The assumption that this is not the case implies that the vertex of degree 3 is in the same component as the $P_4$. Since isomorphism of $(2K_2 \cup 2K_1)$-free triangle-free graphs is isomorphism complete [26, Lemma 5], by assuming that $H_1$ is sufficiently large, we may further assume there is at most one additional vertex not in the connected component of the $P_4$. If there is only one vertex of degree 1 non-adjacent to the vertex $h$ of degree at least 3, and additionally this vertex has distance 2 from $h$, then $H_1$ is an induced subgraph of $H(1, 0, b, 1)$ for some positive integer $b$. This implies that the problem is polynomial-time solvable by Theorem 15. If $H_1$ contains two leaves of distance at least 3 from the center, then if $H_1$ is sufficiently large it also contains $2K_2 \cup 2K_1$. If $H_1$ contains a leaf of distance at least 4 then $H_1$ contains $P_4 \cup 2K_2$. In both cases the problem is graph isomorphism complete by [26, Lemma 5].

(The case $H_2 = K_n$ with $n > 3$). Suppose now $H_2$ is the graph $K_n$ for some $n \geq 4$. Suppose first, $H_1$ contains two non-trivial components. If it contains an isolated vertex, then it contains $2K_2 \cup K_1$ and the problem is isomorphism complete by Theorem 5. If one of the components contains $P_4$ and the graph is not connected, then the graph contains $P_4 \cup K_1$ and the problem is isomorphism complete by [26, Theorem 3]. Otherwise, the two components form a double star and the problem is polynomial-time solvable by Corollary 3.

Thus we may assume now that there is only one non-trivial component. If there is no $P_4$ in $H_1$, the problem is solvable by [26, Theorem 4]. Otherwise, by [26, Theorem 3], we can assume that $H_1$ is connected. If $H_1$ is isomorphic to $P_5$, the problem is solvable by Theorem 14. If $H_1$ is isomorphic to $P_6$ the problem is isomorphism complete by Theorem 5. We can thus assume that $H_1$ contains a vertex of degree at least 3, which we call the center. If there is only one leaf not adjacent to the center and this leaf has distance 2 from the center, then the problem is polynomial-time solvable by Theorem 16. If there are two leaves not adjacent to the center then $H_1$ contains $P_4 \cup K_1$ and the problem is isomorphism complete by [26, Theorem 3].

If there is a leaf of distance at least 3 from the center and the center has degree 4 then the problem is graph isomorphism complete by Theorem 5. We can thus assume that the center has degree 3. Under these conditions, if $H_2 = K_4$ and $H_1$ is sufficiently large, then $H_1$ contains $P_6$ and the problem is isomorphism complete by Theorem 5. In the remaining cases $H_2 = K_n$ with $n \geq 5$ and $H_1$ has a leaf of distance at least 2 from the center. In this case the problem is isomorphism complete by Theorem 7. \qed

\textcopyright Springer
13 Conclusion

There is an intricate relationship between the boundedness of the clique width on a hereditary graph class and polynomial-time solvability of the isomorphism problem. First of all, the isomorphism problem for graphs of bounded clique width can be solved in polynomial time [18]. On the other hand, there are classes of unbounded clique width for which the isomorphism problem is solvable in polynomial time (for example the graphs of bounded degree or the graphs characterized by a forbidden star and a forbidden clique (see [26])). Typically, an isomorphism reduction to a graph class $C$ also yields a proof for the unboundedness of the clique width. For example, it is easy to see that every class of graphs encoded by a simple path encoding has unbounded clique width. The reason is that if $G'$ is the graph produced from a graph $G$ by the reduction detailed in the proof of Theorem 4 then $G$ can be obtained from $G'$ by a finite application of edge complementations between two sets, followed by an unbounded number of local complementations and vertex removals. This shows that the clique width of $G$ is bounded by a function of the clique width of $G'$. Thus, if $G$ has large clique width then $G'$ has large clique width.

Corollary 6 The graphs encoded by a simple path encoding have unbounded clique width.

This implies that many classes $(H_1, H_2)$-free have unbounded clique width. In particular the class $(P_4 \cup K_1, K_4)$-free has unbounded clique width, which was previously an unknown case (see [12]). Conversely, when a graph class is shown to be polynomial-time solvable, often this is due to some structural insight that actually amounts to showing that the clique width of the class is bounded.

As mentioned in the introduction, for various other computational problems, classification results have been considered for classes characterized by two forbidden graphs. For these problems, there had been extensive prior work and algorithmic techniques were already available. For the isomorphism problem, such techniques were lacking and the intention of this paper is to develop them. The fact that for new classes the complexity of isomorphism can be determined using these techniques, while other methods seem to fail, shows that these techniques provide something conceptually different.

Concerning a list of open cases that still remain for the classes characterized by two forbidden subgraphs, one has to analyze precisely to which classes the techniques can be applied. For example, Brandstädt and Kratsch [9] show that cycles of length 5 in $(P_5, P_4 \cup K_1)$-free graphs are disjoint. By coloring vertices inside a 5-cycle depending on whether they have a neighbor outside the cycle, we obtain a decomposition functor. Furthermore, there is a description of the relevant prime graphs in [9] which allows us to apply the modular decomposition technique. This is just an example and a comprehensive description of the various cases that can actually be handled with the techniques remains as future work.

Acknowledgments I thank Matasha Mazis for inspiring comments and suggestions.
References

1. Arvind, V., Das, B., Köbler, J., Toda, S.: Colored hypergraph isomorphism is fixed parameter tractable. In: FSTTCS, pp. 327–337 (2010)
2. Babai, L.: Moderately exponential bound for graph isomorphism. In: FCT, pp. 34–50 (1981)
3. Babai, L.: Automorphism groups, isomorphism, reconstruction. Handbook of Combinatorics, vol 2, pp. 1447–1540. MIT Press (1995)
4. Babai, L.: Graph isomorphism in quasipolynomial time [extended abstract]. In: STOC, pp. 684–697. ACM (2016)
5. Babai, L., Luks, E.M.: Canonical labeling of graphs. In: STOC, pp. 171–183 (1983)
6. Bacsó, G., Tuza, Z.: Dominating cliques in \(P_5\),-free graphs. Period. Math Hungar. 21(4), 303–308 (1990)
7. Bandelt, H.-J., Mulder, H.M.: Distance-hereditary graphs. J. Combinatorial Theory Ser. B 41(2), 182–208 (1986)
8. Booth, K.S., Colbourn, C.J.: Problems Polynomially Equivalent to Graph Isomorphism. Technical Report CS-77-04, Comp. Sci. Dep., Univ. Waterloo (1979)
9. Brandstädt, A., Kratsch, D.: On the structure of \((P_5, \text{gem})\)-free graphs. Discret. Appl. Math. 145(2), 155–166 (2005)
10. Colbourn, M.J., Colbourn, C.J.: Graph isomorphism and self-complementary graphs. SIGACT News 10(1), 25–29 (1978)
11. Curtis, A., Lin, M., McConnell, R., Nussbaum, Y., Soulignac, F., Spinrad, J., Szwarcbiter, J.: Isomorphism of graph classes related to the circular-ones property. Discret. Math. Theor. Comput. Sci. 15(1), 157–182 (2013)
12. Dabrowski, K., Paulusma, D.: Clique-width of graph classes defined by two forbidden induced subgraphs. Comput. J. 59(5), 650–666 (2016)
13. Dabrowski, K.K., Golovach, P.A., Paulusma, D.: Colouring of graphs with ramsey-type forbidden subgraphs. Theor. Comput. Sci. 522, 34–43 (2014)
14. Fuhlbrück, F.: Fixed-Parameter Tractability of the Graph Isomorphism and Canonization Problems. Diploma thesis, Humboldt-Universität zu Berlin (2013)
15. Goldberg, M.K.: A nonfactorial algorithm for testing isomorphism of two graphs. Discret. Appl. Math. 6(3), 229–236 (1983)
16. Golovach, P.A., Paulusma, D.: List coloring in the absence of two subgraphs. Discret. Appl. Math. 166, 123–130 (2014)
17. Grohe, M., Marx, D.: Structure theorem and isomorphism test for graphs with excluded topological subgraphs. In: STOC, pp. 173–192 (2012)
18. Grohe, M., Schweitzer, P.: Isomorphism testing for graphs of bounded rank width. In: FOCS, pp. 1010–1029. IEEE Computer Society (2015)
19. Gurevich, Y.: From Invariants to Canonization. Bulletin of the EATCS, 63 (1997)
20. Gurevich, Y.: From invariants to canonization. In: Current Trends in Theoretical Computer Science, pp. 327–331. World Scientific (2001)
21. Habib, M., Paul, C.: A survey of the algorithmic aspects of modular decomposition. Comput. Sci. Rev. 4(1), 41–59 (2010)
22. Juutti, T.A., Kaski, P.: Conflict propagation and component recursion for canonical labeling. In: TAPAS, pp. 151–162 (2011)
23. Köbler, J., Schöning, U., Torán, J.: The Graph Isomorphism Problem: Its Structural Complexity. Birkhäuser Verlag, Basel, Switzerland (1993)
24. Köbler, J., Verbitsky, O.: From invariants to canonization in parallel. In: CSR, pp. 216–227 (2008)
25. Král, D., Kratochvíl, J., Tuza, Z., Woeginger, G.J.: Complexity of coloring graphs without forbidden induced subgraphs. In: WG, pp. 254–262 (2001)
26. Kratsch, S., Schweitzer, P.: Graph isomorphism for graph classes characterized by forbidden induced subgraphs. In: WG, pp. 34–45 (2012)
27. Kratsch, S., Schweitzer, P.: Graph isomorphism for graph classes characterized by forbidden induced subgraphs. Discret. Appl. Math. 216, 240–253 (2017)
28. Lozin, V.V.: A decidability result for the dominating set problem. Theor. Comput. Sci. 411(44–46), 4023–4027 (2010)
29. Luks, E.M.: Isomorphism of graphs of bounded valence can be tested in polynomial time. J. Comput. Syst. Sci. 25(1), 42–65 (1982)
30. Miller, G.L.: Isomorphism testing and canonical forms for \(k\)-contractable graphs (a generalization of bounded valence and bounded genus). In: FCT, pp. 310–327 (1983)
31. Nakano, S.-i., Uehara, R., Uno, T.: A new approach to graph recognition and applications to distance-hereditary graphs. J. Comput. Sci. Technol. 24(3), 517–533 (2009)
32. Otachi, Y., Schweitzer, P.: Isomorphism on subgraph-closed graph classes: A complexity dichotomy and intermediate graph classes. In: ISAAC, pp. 111–118 (2013)
33. Rao, M.: Decomposition of (gem,co-gem)-free graphs. Unpublished available at http://www.labri.fr/perso/rao/publi/decompgemcogem.ps (2007)
34. Schweitzer, P.: Problems of Unknown Complexity: Graph Isomorphism and Ramsey Theoretic Numbers. PhD thesis, Universität des Saarlandes, Germany (2009)
35. Schweitzer, P.: Towards an isomorphism dichotomy for hereditary graph classes. In: Mayr, E.W., Ollinger, N. (eds.) 32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany, volume 30 of LIPIcs, pp. 689–702. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2015)
36. Seress, A.: Permutation Group Algorithms. Cambridge Tracts in Mathematics. Cambridge University Press (2003)