# Isomorphism Testing Parameterized by Genus and Beyond


**Daniel Neuen**

CISPA Helmholtz Center for Information Security,
Saarland Informatics Campus, Saarbrücken, Germany



## Abstract

We present an isomorphism test for graphs of Euler genus $g$ running in time $2^{\mathcal{O}(g^4 \log g)} n^{\mathcal{O}(1)}$. Our algorithm provides the first explicit upper bound on the dependence on $g$ for an fpt isomorphism test parameterized by the Euler genus of the input graphs. The only previous fpt algorithm runs in time $f(g)n$ for some function $f$ (Kawarabayashi 2015). Actually, our algorithm even works when the input graphs only exclude $K_{3,h}$ as a minor. For such graphs, no fpt isomorphism test was known before.

The algorithm builds on an elegant combination of simple group-theoretic, combinatorial, and graph-theoretic approaches. In particular, we introduce $(t,k)$-WL-bounded graphs which provide a powerful tool to combine group-theoretic techniques with the standard Weisfeiler-Leman algorithm. This concept may be of independent interest.



**2012 ACM Subject Classification** Theory of computation → Fixed parameter tractability; Theory of computation → Graph algorithms analysis; Mathematics of computing → Graphs and surfaces

**Keywords and phrases** graph isomorphism, fixed-parameter tractability, Euler genus, Weisfeiler-Leman algorithm

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2021.72

**Related Version** *Full Version*: `https://arxiv.org/abs/2106.14869`

**Funding** Research supported by the European Research Council (ERC) consolidator grant No. 725978 SYSTEMATICGRAPH.


## 1 Introduction

Determining the computational complexity of the Graph Isomorphism Problem is a long-standing open question in theoretical computer science (see, e.g., [13]). The problem is easily seen to be contained in NP, but it is neither known to be in PTIME nor known to be NP-complete. In a breakthrough result, Babai [1] recently obtained a quasipolynomial-time algorithm for testing isomorphism of graphs (i.e., an algorithm running in time $n^{\mathcal{O}((\log n)^c)}$ where $n$ denotes the number of vertices of the input graphs, and $c$ is a constant), achieving the first improvement over the previous best algorithm running in time $n^{\mathcal{O}(\sqrt{n/\log n})}$ [3] in over three decades. However, it remains wide open whether GI can be solved in polynomial time.

In this work, we are concerned with the parameterized complexity of isomorphism testing. While polynomial-time isomorphism tests are known for a large variety of restricted graph classes (see, e.g., [4, 7, 9, 11, 18, 24]), for several important structural parameters such as maximum degree or the Hadwiger number[1], it is still unknown whether isomorphism testing is fixed-parameter tractable (i.e., whether there is an isomorphism algorithm running in time $f(k)n^{\mathcal{O}(1)}$ where $k$ denotes the graph parameter in question, $n$ the number of vertices of the

[1] The Hadwiger number of a graph $G$ is the maximum number $h$ such that $K_h$ is a minor of $G$.




29th Annual European Symposium on Algorithms (ESA 2021).
Editors: Petra Mutzel, Rasmus Pagh, and Grzegorz Herman; Article No. 72; pp. 72:1–72:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

input graphs, and $f$ is some function). On the other hand, there has also been significant progress in recent years. In 2015, Lokshtanov et al. [17] obtained the first fpt isomorphism test parameterized by the tree-width $k$ of the input graph running in time $2^{\mathcal{O}(k^5 \log k)}n^5$. This algorithm was later improved by Grohe et al. [8] to a running time of $2^{\mathcal{O}(k\cdot(\log k)^c)}n^3$ (for some constant $c$). In the same year, Kawarabayashi [14] obtained the first fpt isomorphism test parameterized by the Euler genus $g$ of the input graph running time $f(g)n$ for some function $f$. While Kawarabayashi's algorithm achieves optimal dependence on the number of vertices of the input graphs, it is also extremely complicated and it provides no explicit upper bound on the function $f$. Indeed, the algorithm spans over multiple papers [14, 15, 16] and builds on several deep structural results for graphs of bounded genus.

In this work, we present an alternative isomorphism test for graphs of Euler genus $g$ running in time $2^{\mathcal{O}(g^4 \log g)}n^{\mathcal{O}(1)}$. In contrast to Kawarabayashi's algorithm, our algorithm does not require any deep graph-theoretic insights, but rather builds on an elegant combination of well-established and simple group-theoretic, combinatorial, and graph-theoretic ideas. In particular, this enables us to provide the first explicit upper bound on the dependence on $g$ for an fpt isomorphism test. Actually, the only property our algorithm exploits is that graphs of genus $g$ exclude $K_{3,h}$ as a minor for $h \geq 4g + 3$ [25]. In other words, our main result is an fpt isomorphism test for graphs excluding $K_{3,h}$ as a minor.

▶ **Theorem 1.** *The Graph Isomorphism Problem for graphs excluding $K_{3,h}$ as a minor can be solved in time $2^{\mathcal{O}(h^4 \log h)}n^{\mathcal{O}(1)}$.*

For this class of graphs, the best existing algorithm runs in time $n^{\mathcal{O}((\log h)^c)}$ for some constant $c$ [21], and no fpt isomorphism test was known prior to this work.

For the algorithm, we combine different approaches to the Graph Isomorphism Problem. On a high-level, our algorithm follows a simple decomposition strategy which decomposes the input graph $G$ into pieces such that the interplay between the pieces is simple. The main idea is to define the pieces in such a way that, after fixing a small number of vertices, the automorphism group of $G$ restricted to a piece $D \subseteq V(G)$ is similar to the automorphism group of a graph of maximum degree 3. This allows us to test isomorphism between the pieces using the group-theoretic graph isomorphism machinery dating back to Luks's polynomial-time isomorphism test for graphs of bounded maximum degree [18].

In order to capture the restrictions on the automorphism group, we introduce the notion of $(t, k)$-*WL-bounded graphs* which generalize so-called $t$-CR-bounded graphs. The class of $t$-CR-bounded graphs was originally defined by Ponomarenko [23] and was recently rediscovered in [21, 10, 22] in a series of works eventually leading to an algorithm testing isomorphism of graphs excluding $K_h$ as a topological subgraph in time $n^{\mathcal{O}((\log h)^c)}$. Intuitively speaking, a graph $G$ is $t$-*CR-bounded* if an initially uniform vertex-coloring $\chi$ can be turned into a discrete coloring (i.e., a coloring where every vertex has its own color) by repeatedly (a) applying the standard Color Refinement algorithm, and (b) splitting all color classes of size at most $t$. We define $(t, k)$-*WL-bounded* graphs in the same way, but replace the Color Refinement algorithm by the well-known Weisfeiler-Leman algorithm of dimension $k$ (see, e.g., [5, 12]). Maybe surprisingly, this natural extension of $t$-CR-bounded has not been considered so far in the literature, and we start by building a polynomial-time isomorphism test for such graphs using the group-theoretic methods developed by Luks [18] as well as a simple extension due to Miller [20]. Actually, it turns out that isomorphism of $(t, k)$-WL-bounded graphs can even be tested in time $n^{\mathcal{O}(k\cdot(\log t)^c)}$ using recent extensions [21] of Babai's quasipolynomial-time isomorphism test. However, since we only apply these methods for $t = k = 2$, there is no need for our algorithm to rely on such sophisticated subroutines.

Now, as the main structural insight, we prove that each 3-connected graph $G$ that excludes $K_{3,h}$ as a minor admits (after fixing 3 vertices) an isomorphism-invariant rooted tree decomposition $(T, \beta)$ such that the adhesion width (i.e., the maximal intersection between two bags) is bounded by $h$. Additionally, each bag $\beta(t)$, $t \in V(T)$, can be equipped with a set $\gamma(t) \subseteq \beta(t)$ of size $|\gamma(t)| \leq h^4$ such that, after fixing all vertices in $\gamma(t)$, $G$ restricted to $\beta(t)$ is $(2,2)$-WL-bounded. Given such a decomposition, isomorphisms can be computed by a simple bottom-up dynamic programming strategy along the tree decompositions. For each bag, isomorphism is tested by first individualizing all vertices from $\gamma(t)$ at an additional factor of $|\gamma(t)|! = 2^{\mathcal{O}(h^4 \log h)}$ in the running time. Following the individualization of these vertices, our algorithm can then simply rely on a polynomial-time isomorphism test for $(2,2)$-WL-bounded graphs. Here, we incorporate the partial solutions computed in the subtree below the current bag via a simple gadget construction.

To compute the decomposition $(T, \beta)$, we also build on the notion of $(2,2)$-WL-bounded graphs. Given a set $X \subseteq V(G)$, we define the $(2,2)$-*closure* to be the set $D = \text{cl}_{2,2}^G(X)$ of all vertices appearing in a singleton color class after artificially individualizing all vertices from $X$, and performing the $(2,2)$-WL procedure. As one of the main technical contributions, we can show that the interplay between $D$ and its complement in $G$ is simple (assuming $G$ excludes $K_{3,h}$ as a minor). To be more precise, building on various properties of the 2-dimensional Weisfeiler-Leman algorithm, we show that $|N_G(Z)| < h$ for every connected component $Z$ of $G - D$. This allows us to choose $D = \text{cl}_{2,2}^G(X)$ as the root bag of $(T, \beta)$ for some carefully chosen set $X$, and obtain the decomposition $(T, \beta)$ by recursion.

# 2 Preliminaries

## 2.1 Graphs

A *graph* is a pair $G = (V(G), E(G))$ consisting of a *vertex set* $V(G)$ and an *edge set* $E(G)$. All graphs considered in this paper are finite and simple (i.e., they contain no loops or multiple edges). Moreover, unless explicitly stated otherwise, all graphs are undirected. For an undirected graph $G$ and $v, w \in V(G)$, we write $vw$ as a shorthand for $\{v, w\} \in E(G)$. The *neighborhood* of a vertex $v \in V(G)$ is denoted by $N_G(v)$. The *degree* of $v$, denoted by $\deg_G(v)$, is the number of edges incident with $v$, i.e., $\deg_G(v) = |N_G(v)|$. For $X \subseteq V(G)$, we define $N_G(X) := (\bigcup_{v \in X} N_G(v)) \setminus X$. If the graph $G$ is clear from context, we usually omit the index and simply write $N(v)$, $\deg(v)$ and $N(X)$. We write $K_{\ell,h}$ to denote the complete bipartite graph on $\ell$ vertices on the left side and $h$ vertices on the right side. For two sets $A, B \subseteq V(G)$, we denote by $E_G(A, B) := \{vw \in E(G) \mid v \in A, w \in B\}$. Also, $G[A, B]$ denotes the graph with vertex set $A \cup B$ and edge set $E_G(A, B)$. Moreover, $G[A] := G[A, A]$ denotes the induced subgraph on $A$, and $G - A$ the subgraph induced by the complement of $A$, that is, the graph $G - A := G[V(G) \setminus A]$. For $F \subseteq E(G)$, we also define $G - F$ to be the graph obtained from $G$ by removing all edges contained in $F$ (the vertex set remains unchanged). A graph $H$ is a *subgraph* of $G$, denoted by $H \subseteq G$, if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. A graph $H$ is a *minor* of $G$ if $H$ can be obtained from $G$ by deleting vertices and edges, as well as contracting edges. The graph $G$ *excludes $H$ as a minor* if it does not have a minor isomorphic to $H$.

An *isomorphism* from $G$ to a graph $H$ is a bijection $\varphi\colon V(G) \to V(H)$ that respects the edge relation, that is, for all $v, w \in V(G)$, it holds that $vw \in E(G)$ if and only if $\varphi(v)\varphi(w) \in E(H)$. Two graphs $G$ and $H$ are *isomorphic*, written $G \cong H$, if there is an isomorphism from $G$ to $H$. We write $\varphi\colon G \cong H$ to denote that $\varphi$ is an isomorphism from $G$ to $H$. Also, $\mathrm{Iso}(G,H)$ denotes the set of all isomorphisms from $G$ to $H$. The automorphism group of $G$ is $\mathrm{Aut}(G) \coloneqq \mathrm{Iso}(G,G)$. Observe that, if $\mathrm{Iso}(G,H) \neq \emptyset$, it holds that $\mathrm{Iso}(G,H) = \mathrm{Aut}(G)\varphi \coloneqq \{\gamma\varphi \mid \gamma \in \mathrm{Aut}(G)\}$ for every isomorphism $\varphi \in \mathrm{Iso}(G,H)$.

A *vertex-colored graph* is a tuple $(G,\chi_V)$ where $G$ is a graph and $\chi_V \colon V(G) \to C$ is a mapping into some set $C$ of colors, called *vertex-coloring*. Similarly, an *arc-colored graph* is a tuple $(G,\chi_E)$, where $G$ is a graph and $\chi_E \colon \{(u,v) \mid \{u,v\} \in E(G)\} \to C$ is a mapping into some color set $C$, called *arc-coloring*. Observe that colors are assigned to directed edges, i.e., the directed edge $(v,w)$ may obtain a different color than $(w,v)$. We also consider vertex- and arc-colored graphs $(G,\chi_V,\chi_E)$ where $\chi_V$ is a vertex-coloring and $\chi_E$ is an arc-coloring. Typically, $C$ is chosen to be an initial segment $[n] \coloneqq \{1,\dots,n\}$ of the natural numbers. To be more precise, we generally assume that there is a linear order on the set of all potential colors which, for example, allows us to identify a minimal color appearing in a graph in a unique way. Isomorphisms between vertex- and arc-colored graphs have to respect the colors of the vertices and arcs.

## 2.2 Weisfeiler-Leman Algorithm

The Weisfeiler-Leman algorithm, originally introduced by Weisfeiler and Leman in its 2-dimensional version [28], forms one of the most fundamental subroutines in the context of isomorphism testing.

Let $\chi_1,\chi_2 \colon V^k \to C$ be colorings of $k$-tuples, where $C$ is a finite set of colors. We say $\chi_1$ *refines* $\chi_2$, denoted $\chi_1 \preceq \chi_2$, if $\chi_1(\bar v) = \chi_1(\bar w)$ implies $\chi_2(\bar v) = \chi_2(\bar w)$ for all $\bar v, \bar w \in V^k$. The colorings $\chi_1$ and $\chi_2$ are *equivalent*, denoted $\chi_1 \equiv \chi_2$, if $\chi_1 \preceq \chi_2$ and $\chi_2 \preceq \chi_1$.

We describe the *$k$-dimensional Weisfeiler-Leman algorithm* ($k$-WL) for all $k \geq 1$. For an input graph $G$ let $\chi^k_{(0)}[G] \colon (V(G))^k \to C$ be the coloring where each tuple is colored with the isomorphism type of its underlying ordered subgraph. More precisely, $\chi^k_{(0)}[G](v_1,\dots,v_k) = \chi^k_{(0)}[G](v'_1,\dots,v'_k)$ if and only if, for all $i,j \in [k]$, it holds that $v_i = v_j \Leftrightarrow v'_i = v'_j$ and $v_iv_j \in E(G) \Leftrightarrow v'_iv'_j \in E(G)$. If the graph is equipped with a coloring the initial coloring $\chi^k_{(0)}[G]$ also takes the input coloring into account. More precisely, for a vertex-coloring $\chi_V$, it additionally holds that $\chi_V(v_i) = \chi_V(v'_i)$ for all $i \in [k]$. And for an arc-coloring $\chi_E$, it is the case that $\chi_E(v_i,v_j) = \chi_E(v'_i,v'_j)$ for all $i,j \in [k]$ such that $v_iv_j \in E(G)$.

We then recursively define the coloring $\chi^k_{(i)}[G]$ obtained after $i$ rounds of the algorithm. For $k \geq 2$ and $\bar v = (v_1,\dots,v_k) \in (V(G))^k$ let $\chi^k_{(i+1)}[G](\bar v) \coloneqq \Big(\chi^k_{(i)}[G](\bar v), \mathcal{M}_i(\bar v)\Big)$ where

$$\mathcal{M}_i(\bar v) \coloneqq \Big\{\!\!\Big\{ \big(\chi^k_{(i)}[G](\bar v[w/1]),\dots,\chi^k_{(i)}[G](\bar v[w/k])\big) \;\Big|\; w \in V(G) \Big\}\!\!\Big\}$$

and $\bar v[w/i] \coloneqq (v_1,\dots,v_{i-1},w,v_{i+1},\dots,v_k)$ is the tuple obtained from $\bar v$ by replacing the $i$-th entry by $w$ (and $\{\!\{\dots\}\!\}$ denotes a multiset). For $k = 1$ the definition is similar, but we only iterate over neighbors of $v$, i.e., $\chi^1_{(i+1)}[G](v) \coloneqq \Big(\chi^1_{(i)}[G](v), \mathcal{M}_i(v)\Big)$ where

$$\mathcal{M}_i(v) \coloneqq \Big\{\!\!\Big\{ \chi^1_{(i)}[G](w) \;\Big|\; w \in N_G(v) \Big\}\!\!\Big\}.$$

By definition, $\chi^k_{(i+1)}[G] \preceq \chi^k_{(i)}[G]$ for all $i \geq 0$. Hence, there is a minimal $i_\infty$ such that $\chi^k_{(i_\infty)}[G] \equiv \chi^k_{(i_\infty+1)}[G]$ and for this $i_\infty$ the coloring $\chi^k_{\mathsf{WL}}[G] \coloneqq \chi^k_{(i_\infty)}[G]$ is the *$k$-stable* coloring of $G$. The $k$-dimensional Weisfeiler-Leman algorithm takes as input a (vertex- or arc-)colored graph $G$ and returns (a coloring that is equivalent to) $\chi^k_{\mathsf{WL}}[G]$. This can be implemented in time $\mathcal{O}(n^{k+1}\log n)$ (see [12]).

## 2.3 Group Theory

We introduce the group-theoretic notions required in this work. We refer to [26, 6] for further background.

**Permutation groups.** A *permutation group* acting on a set $\Omega$ is a subgroup $\Gamma \leq \mathrm{Sym}(\Omega)$ of the symmetric group. The size of the permutation domain $\Omega$ is called the *degree* of $\Gamma$. If $\Omega = [n]$, then we also write $S_n$ instead of $\mathrm{Sym}(\Omega)$. For $\gamma \in \Gamma$ and $\alpha \in \Omega$ we denote by $\alpha^\gamma$ the image of $\alpha$ under the permutation $\gamma$. For $A \subseteq \Omega$ and $\gamma \in \Gamma$ let $A^\gamma := \{\alpha^\gamma \mid \alpha \in A\}$. The set $A$ is $\Gamma$-*invariant* if $A^\gamma = A$ for all $\gamma \in \Gamma$. For a partition $\mathcal{P}$ of $\Omega$ let $\mathcal{P}^\gamma := \{A^\gamma \mid A \in \mathcal{P}\}$. Observe that $\mathcal{P}^\gamma$ is again a partition of $\Gamma$. We say $\mathcal{P}$ is $\Gamma$-*invariant* if $\mathcal{P}^\gamma = \mathcal{P}$ for all $\gamma \in \Gamma$.

For $A \subseteq \Omega$ and a bijection $\theta \colon \Omega \to \Omega'$ we denote by $\theta[A]$ the restriction of $\theta$ to the domain $A$. For a $\Gamma$-invariant set $A \subseteq \Omega$, we denote by $\Gamma[A] := \{\gamma[A] \mid \gamma \in \Gamma\}$ the induced action of $\Gamma$ on $A$, i.e., the group obtained from $\Gamma$ by restricting all permutations to $A$. More generally, for every set $\Lambda$ of bijections with domain $\Omega$, we denote by $\Lambda[A] := \{\theta[A] \mid \theta \in \Lambda\}$. Similarly, for a partition $\mathcal{P}$ of $\Omega$, we denote by $\theta[\mathcal{P}] \colon \mathcal{P} \to \mathcal{P}'$ the mapping defined via $\theta(A) := \{\theta(\alpha) \mid \alpha \in A\}$ for all $A \in \mathcal{P}$. As before, $\Lambda[\mathcal{P}] := \{\theta[\mathcal{P}] \mid \theta \in \Lambda\}$.

**Groups with restricted composition factors.** We shall be interested in a particular subclass of permutation groups, namely groups with restricted composition factors. Let $\Gamma$ be a group. A *subnormal series* is a sequence of subgroups $\Gamma = \Gamma_0 \geq \Gamma_1 \geq \cdots \geq \Gamma_k = \{\mathrm{id}\}$ such that $\Gamma_i$ is a normal subgroup of $\Gamma_{i-1}$ for all $i \in [k]$. The length of the series is $k$ and the groups $\Gamma_{i-1}/\Gamma_i$ are the factor groups of the series, $i \in [k]$. A *composition series* is a strictly decreasing subnormal series of maximal length. For every finite group $\Gamma$ all composition series have the same family (considered as a multiset) of factor groups (cf. [26]). A *composition factor* of a finite group $\Gamma$ is a factor group of a composition series of $\Gamma$.

▶ **Definition 2.** *For $d \geq 2$ let $\widehat{\Gamma}_d$ denote the class of all groups $\Gamma$ for which every composition factor of $\Gamma$ is isomorphic to a subgroup of $S_d$. A group $\Gamma$ is a $\widehat{\Gamma}_d$-*group *if it is contained in the class $\widehat{\Gamma}_d$.*

Let us point out the fact that there are two similar classes of groups usually referred by $\Gamma_d$ in the literature. The first is the class denoted by $\widehat{\Gamma}_d$ here originally introduced by Luks [18], while the second one, for example used in [2], in particular allows composition factors that are simple groups of Lie type of bounded dimension.

**Group-Theoretic Tools for Isomorphism Testing.** In this work, the central group-theoretic subroutine is an isomorphism test for hypergraphs where the input group is a $\widehat{\Gamma}_d$-group. Two hypergraphs $\mathcal{H}_1 = (V_1, \mathcal{E}_1)$ and $\mathcal{H}_2 = (V_2, \mathcal{E}_2)$ are isomorphic if there is a bijection $\varphi \colon V_1 \to V_2$ such that $E \in \mathcal{E}_1$ if and only if $E^\varphi \in \mathcal{E}_2$ for all $E \in 2^{V_1}$ (where $E^\varphi := \{\varphi(v) \mid v \in E\}$ and $2^{V_1}$ denotes the power set of $V_1$). We write $\varphi \colon \mathcal{H}_1 \cong \mathcal{H}_2$ to denote that $\varphi$ is an isomorphism from $\mathcal{H}_1$ to $\mathcal{H}_2$. Consistent with previous notation, we denote by $\mathrm{Iso}(\mathcal{H}_1, \mathcal{H}_2)$ the set of isomorphisms from $\mathcal{H}_1$ to $\mathcal{H}_2$. More generally, for $\Gamma \leq \mathrm{Sym}(V_1)$ and a bijection $\theta \colon V_1 \to V_2$, we define $\mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2) := \{\varphi \in \Gamma\theta \mid \varphi \colon \mathcal{H}_1 \cong \mathcal{H}_2\}$. In this work, we define the Hypergraph Isomorphism Problem to take as input two hypergraphs $\mathcal{H}_1 = (V_1, \mathcal{E}_1)$ and $\mathcal{H}_2 = (V_2, \mathcal{E}_2)$, a group $\Gamma \leq \mathrm{Sym}(V_1)$ and a bijection $\theta \colon V_1 \to V_2$, and the goal is to compute a representation[2] of $\mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2)$. The following algorithm forms a crucial subroutine.

[2] While $\mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2)$ may be exponentially large, it can be represented by a single isomorphism $\varphi \in \mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2)$ and a generating set for $\mathrm{Aut}_\Gamma(\mathcal{H}_1) := \mathrm{Iso}_\Gamma(\mathcal{H}_1, \mathcal{H}_1)$. For general background on how to perform computations on permutation groups, I refer to [27].

▶ **Theorem 3** (Miller [20]). *Let $\mathcal{H}_1 = (V_1, \mathcal{E}_1)$ and $\mathcal{H}_2 = (V_2, \mathcal{E}_2)$ be two hypergraphs and let $\Gamma \leq \mathrm{Sym}(V_1)$ be a $\widehat{\Gamma}_d$-group and $\theta\colon V_1 \to V_2$ a bijection. Then $\mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2)$ can be computed in time $(n+m)^{\mathcal{O}(d)}$ where $n := |V_1|$ and $m := |\mathcal{E}_1|$.*

▶ **Theorem 4** (Neuen [21]). *Let $\mathcal{H}_1 = (V_1, \mathcal{E}_1)$ and $\mathcal{H}_2 = (V_2, \mathcal{E}_2)$ be two hypergraphs and let $\Gamma \leq \mathrm{Sym}(V_1)$ be a $\widehat{\Gamma}_d$-group and $\theta\colon V_1 \to V_2$ a bijection. Then $\mathrm{Iso}_{\Gamma\theta}(\mathcal{H}_1, \mathcal{H}_2)$ can be computed in time $(n+m)^{\mathcal{O}((\log d)^c)}$ for some constant $c$ where $n := |V_1|$ and $m := |\mathcal{E}_1|$.*

Observe that both algorithms given by the two theorems tackle the same problem. The second algorithm is asymptotically much faster, but it is also much more complicated and the constant factors in the exponent of the running time are likely to be much higher. Since this paper only applies either theorem for $d = 2$, it seems to be preferable to use the first algorithm. Indeed, the first result is a simple extension of Luks's well-known isomorphism test for bounded-degree graphs [18], and thus the underlying algorithm is fairly simple. For all these reasons, we mostly build on Theorem 3. However, for future applications of the techniques presented in this work, it might be necessary to build on Theorem 4 to benefit from the improved run time bound. For this reason, we shall provide variants of our results building on Theorem 4 wherever appropriate.

## 3 Allowing Weisfeiler and Leman to Split Small Color Classes

In this section, we introduce the concept of $(t,k)$-*WL-bounded graphs* and provide a polynomial-time isomorphism test for such graphs for all constant values of $t$ and $k$. The final fpt isomorphism test for graphs excluding $K_{3,h}$ as a minor builds on this subroutine for $t = k = 2$.

The concept of $(t,k)$-WL-bounded graphs is a natural extension of $t$-CR-bounded graphs which were already introduced by Ponomarenko in the late 1980's [23] and which were recently rediscovered in [21, 10, 22]. Intuitively speaking, a graph $G$ is *$t$-CR-bounded*, $t \in \mathbb{N}$, if an initially uniform vertex-coloring $\chi$ (i.e., all vertices receive the same color) can be turned into the discrete coloring (i.e., each vertex has its own color) by repeatedly

- performing the Color Refinement algorithm (expressed by the letters "CR"), and
- taking a color class $[v]_\chi := \{w \in V(G) \mid \chi(w) = \chi(v)\}$ of size $|[v]_\chi| \leq t$ and assigning each vertex from the class its own color.

A very natural extension of this idea to replace the Color Refinement algorithm by the Weisfeiler-Leman algorithm for some fixed dimension $k$. This leads us to the notion of $(t,k)$-*WL-bounded graphs* (the letters "CR" are replaced by "$k$-WL"). In particular, $(t,1)$-WL-bounded graphs are exactly the $t$-CR-bounded graphs. Maybe surprisingly, it seems that this simple extension has not been considered so far in the literature.

▶ **Definition 5.** *A vertex- and arc-colored graph $G = (V, E, \chi_V, \chi_E)$ is* $(t,k)$-WL-bounded *if the sequence $(\chi_i)_{i \geq 0}$ reaches a discrete coloring where $\chi_0 := \chi_V$,*

$$\chi_{2i+1}(v) := \chi_{\mathsf{WL}}^{k}[V, E, \chi_{2i}, \chi_E](v, \dots, v)$$

*and*

$$\chi_{2i+2}(v) := \begin{cases} (v, 1) & \text{if } |[v]_{\chi_{2i+1}}| \leq t \\ (\chi_{2i+1}(v), 0) & \text{otherwise} \end{cases}$$

*for all $i \geq 0$.*

*Also, for the minimal $i_\infty \geq 0$ such that $\chi_{i_\infty} \equiv \chi_{i_\infty+1}$, we refer to $\chi_{i_\infty}$ as the* $(t,k)$-WL-stable *coloring of $G$ and denote it by $\chi_{(t,k)\text{-}\mathsf{WL}}[G]$.*

At this point, the reader may wonder why $(\chi_i)_{i\geq 0}$ is chosen as a sequence of vertex-colorings and not a sequence of colorings of $k$-tuples of vertices (since $k$-WL also colors $k$-tuples of vertices). While such a variant certainly makes sense, it still leads to the same class of graphs. Let $G$ be a graph and let $\chi \coloneqq \chi_{\mathsf{WL}}^k[G]$. The main insight is that, if there is some color $c \in \operatorname{im}(\chi)$ for which $|\chi^{-1}(c)| \leq t$, then there is also a color $c' \in \operatorname{im}(\chi)$ for which $|\chi^{-1}(c')| \leq t$ and $\chi^{-1}(c') \subseteq \{(v,\dots,v) \mid v \in V(G)\}$. In other words, one can not achieve any additional splitting of color classes by also considering non-diagonal color classes.

We also need to extend several notions related to $t$-CR-bounded graphs. Let $G$ be a graph and let $X \subseteq V(G)$ be a set of vertices. Let $\chi_V^*\colon V(G) \to C$ be the vertex-coloring obtained from individualizing all vertices in the set $X$, i.e., $\chi_V^*(v) \coloneqq (v,1)$ for $v \in X$ and $\chi_V^*(v) \coloneqq (0,0)$ for $v \in V(G) \setminus X$. Let $\chi \coloneqq \chi_{(t,k)\text{-}\mathsf{WL}}[G,\chi_V^*]$ denote the $(t,k)$-WL-stable coloring with respect to the input graph $(G,\chi_V^*)$. We define the $(t,k)$-*closure* of the set $X$ (with respect to $G$) to be the set

$$\operatorname{cl}_{t,k}^G(X) \coloneqq \{v \in V(G) \mid |[v]_\chi| = 1\}\,.$$

Observe that $X \subseteq \operatorname{cl}_{t,k}^G(X)$. For $v_1,\dots,v_\ell \in V(G)$ we also use $\operatorname{cl}_{t,k}^G(v_1,\dots,v_\ell)$ as a shorthand for $\operatorname{cl}_{t,k}^G(\{v_1,\dots,v_\ell\})$. If the input graph is equipped with a vertex- or arc-coloring, all definitions are extended in the natural way.

Now, we concern ourselves with designing a polynomial-time isomorphism test for $(t,k)$-WL-bounded graphs. Actually, we shall prove a slightly stronger result which turns out to be useful later on. The main idea for the algorithm is to build a reduction to the isomorphism problem for $(t,1)$-WL-bounded graphs for which such results are already known [23, 21]. Indeed, isomorphism of $(t,1)$-WL-bounded graphs can be reduced to the Hypergraph Isomorphism Problem for $\widehat{\Gamma}_t$-groups. Since one may be interested in using different subroutines for solving the Hypergraph Isomorphism Problem for $\widehat{\Gamma}_t$-groups (see the discussion at the end of Section 2.3), the main result is stated via an oracle for the Hypergraph Isomorphism Problem on $\widehat{\Gamma}_t$-groups.

▶ **Theorem 6.** *Let $G_1, G_2$ be two vertex- and arc-colored graphs and let $\chi_i \coloneqq \chi_{(t,k)\text{-}\mathsf{WL}}[G_i]$. Also let $\mathcal{P}_i = \{[v]_{\chi_i} \mid v \in V(G_i)\}$ be the partition into color classes of $\chi_i$. Then $\mathcal{P}_1^\varphi = \mathcal{P}_2$ for all $\varphi \in \operatorname{Iso}(G_1,G_2)$.*

*Moreover, using oracle access to the Hypergraph Isomorphism Problem for $\widehat{\Gamma}_t$-groups, in time $n^{\mathcal{O}(k)}$ one can compute a $\widehat{\Gamma}_t$-group $\Gamma \leq \operatorname{Sym}(\mathcal{P}_1)$ and a bijection $\theta\colon \mathcal{P}_1 \to \mathcal{P}_2$ such that*

$$\operatorname{Iso}(G_1,G_2)[\mathcal{P}_1] \subseteq \Gamma\theta.$$

*In particular, $\operatorname{Aut}(G_1)[\mathcal{P}_1] \in \widehat{\Gamma}_t$.*

▶ **Corollary 7.** *Let $G_1, G_2$ be two $(t,k)$-WL-bounded graphs. Then a representation for $\operatorname{Iso}(G_1,G_2)$ can be computed in time $n^{\mathcal{O}(k\cdot(\log t)^c)}$ for some absolute constant $c$.*

## 4 Structure Theory and Small Color Classes

Having established the necessary tools, we can now turn to the isomorphism test for graphs excluding $K_{3,h}$ as a minor. We start by giving a high-level overview on the algorithm. The main idea is to build on the isomorphism test for $(2,2)$-WL-bounded graphs described in the last section. Let $G_1$ and $G_2$ be two (vertex- and arc-colored) graphs that exclude $K_{3,h}$ as a minor. Using well-known reduction techniques building on isomorphism-invariant decompositions into triconnected[3] components (see, e.g., [11]), we may assume without loss of generality that $G_1$ and $G_2$ are 3-connected.

[3] A triconnected component is either 3-connected or a cycle.

The algorithm starts by individualizing three vertices. To be more precise, the algorithm picks three distinct vertices $v_1, v_2, v_3 \in V(G_1)$ and iterates over all choices of potential images $w_1, w_2, w_3 \in V(G_2)$ under some isomorphism between $G_1$ and $G_2$. Let $X_1 := \{v_1, v_2, v_3\}$ and $X_2 := \{w_1, w_2, w_3\}$. Also, let $D_i := \mathrm{cl}_{2,2}^{G_i}(X_i)$ denote the $(2,2)$-closure of $X_i$, $i \in \{1,2\}$. Observe that $D_i$ is defined in an isomorphism-invariant manner given the initial choice of $X_i$. Building on Theorems 3 and 6 it can be checked in polynomial time whether $G_1$ and $G_2$ are isomorphic restricted to the sets $D_1$ and $D_2$.

Now, the central idea is to follow a decomposition strategy. Let $Z_1^i, \dots, Z_\ell^i$ denote the vertex sets of the connected components of $G_i - D_i$, and let $S_j^i := N_{G_i}(Z_j^i)$ for $j \in [\ell]$ and $i \in \{1,2\}$. We recursively compute isomorphisms between all pairs of graphs $G_i[Z_j^i \cup S_j^i]$ for all $j \in [\ell]$ and $i \in \{1,2\}$. To be able to determine whether all these partial isomorphisms can be combined into a global isomorphism, the crucial insight is that $|S_j^i| < h$ for all $j \in [\ell]$ and $i \in \{1,2\}$.

▶ **Lemma 8.** *Let $G$ be a graph that excludes $K_{3,h}$ as a minor. Also let $X \subseteq V(G)$ and define $D := \mathrm{cl}_{2,2}^G(X)$. Let $Z$ be a connected component of $G - D$. Then $|N_G(Z)| < h$.*

Indeed, this lemma forms one of the main technical contributions of the paper. I remark that similar statements are exploited in [10, 21, 22] eventually leading to an isomorphism test running in time $n^{\mathcal{O}((\log h)^c)}$ for all graphs excluding $K_h$ as a topological subgraph. However, all these variants require the $(t,k)$-closure to be taken for non-constant values of $t$ (i.e., $t = \Omega(h)$). For the design of an fpt-algorithm, this is infeasible since we can only afford to apply Theorem 6 for constant values of $t$ and $k$ (since $D_i$ might be equal to $V(G_i)$).

The lemma above implies that the interplay between $D_i$ and $V(G_i) \setminus D_i$ is simple which allows for a dynamic programming approach. To be more precise, we can recursively list all elements of the set $\mathrm{Iso}((G_i[Z_j^i \cup S_j^i], S_j^i), (G_{i'}[Z_{j'}^{i'} \cup S_{j'}^{i'}], S_{j'}^{i'}))[S_j^i]$ for all $j, j' \in [\ell]$ and $i, i' \in \{1,2\}$ (i.e., we list all bijections $\sigma\colon S_j^i \to S_{j'}^{i'}$ that can be extended to an isomorphism between the corresponding subgraphs). To incorporate this information, we extend the graph $G_i[D_i]$ by simple gadgets obtaining graphs $H_i$ that are $(2,2)$-WL-bounded and such that $G_1 \cong G_2$ if and only if $H_1 \cong H_2$. (For technical reasons, the algorithm does not exactly implement this strategy, but closely follows the general idea.)

In order to realize this recursive strategy, it remains to ensure that the algorithm makes progress when performing a recursive call. Actually, this turns out to be a non-trivial task. Indeed, it may happen that $D_i = X_i$, there is only a single component $Z_1^i$ of $G_i - D_i$, and $N_{G_i}(Z_1^i) = D_i$. To circumvent this problem, the idea is to compute an isomorphism-invariant extension $\gamma(X_i) \supsetneq X_i$ such that $|\gamma(X_i)| \leq h^4$. Assuming such an extension can be computed, we simply extend the set $X_i$ until the algorithm arrives in a situation where the recursive scheme discussed above makes progress. Observe that this is guaranteed to happen as soon as $|X_i| \geq h$ building on Lemma 8. Also note that we can still artificially individualize all vertices from $X_i$ at a cost of $2^{\mathcal{O}(h^4 \log h)}$ (since any isomorphism can only map vertices from $X_1$ to vertices from $X_2$).

To compute the extension, we exploit the fact that $G_i$ is $(h-1,1)$-WL-bounded by [21, Corollary 24] (after individualizing 3 vertices). Simply speaking, for every choice of three distinct vertices in $X_i$, after individualizing these vertices and performing the 1-dimensional Weisfeiler-Leman algorithm, we can identify a color class of size at most $h-1$ to be added to the set $X_i$. Overall, assuming $|X_i| \leq h$, this gives an extension $\gamma(X_i)$ of size at most $h + h^3(h-1) \leq h^4$.

This completes the description of the general strategy. In the following sections, we provide more detailed arguments. We first provide a sketch on the proof of Lemma 8 in the next section. Afterwards, we compute the entire decompositions of the input graphs in Section 6. Finally, the dynamic programming strategy along the computed decompositions is implemented in Section 7.

## 5 Finding Disjoint and Connected Subgraphs

In this section, we give some details on the proof of Lemma 8. Let us start by introducing some additional notation for the 2-dimensional Weisfeiler-Leman algorithm.

Let $G$ be a graph and let $\chi \coloneqq \chi_{\mathsf{WL}}^2[G]$ be the coloring computed by the 2-dimensional Weisfeiler-Leman algorithm. We denote by $C_V = C_V(G, \chi) \coloneqq \{\chi(v, v) \mid v \in V(G)\}$ the set of *vertex colors* under the coloring $\chi$. Also, for $c \in C_V$, $V_c \coloneqq \{v \in V(G) \mid \chi(v, v) = c\}$ denotes the set of all vertices of color $c$. Moreover, we define the graph $G[[\chi]]$ with vertex set $V(G[[\chi]]) \coloneqq C_V(G, \chi)$ and edges $E(G[[\chi]]) \coloneqq \{c_1c_2 \mid \exists v_1 \in V_{c_1}, v_2 \in V_{c_2} : v_1v_2 \in E(G)\}$.

The next lemma builds the main technical step in the proof of Lemma 8.

▶ **Lemma 9.** *Let $G$ be a graph and let $\chi$ be a 2-stable coloring. Suppose that $G[[\chi]]$ is connected and $|V_c| \geq 3$ for every $c \in C_V$. Then there are vertex-disjoint, connected subgraphs $H_1, H_2, H_3 \subseteq G$ such that $V(H_r) \cap V_c \neq \emptyset$ for all $r \in \{1, 2, 3\}$ and $c \in C_V$.*

**Proof Idea.** Let $F$ be a spanning tree of $G[[\chi]]$ and fix an arbitrary root node $c_0 \in V(F) = C_V$. On a high-level, the graphs $H_1, H_2, H_3$ are constructed in a top-to-bottom fashion along the tree $F$. To start, let us select three arbitrary distinct vertices $v_1, v_2, v_3 \in V_{c_0}$ and add $v_r$ to the graph $H_r$. Now, let $d$ be a color which is already covered (i.e., $V_d \cap H_r \neq \emptyset$ for all $r \in \{1, 2, 3\}$), and let $c$ be a child of $d$ which is not covered. Consider the graph $G[V_d, V_c]$. Let $v_r \in V_d \cap V(H_r)$ for $r \in \{1, 2, 3\}$. If there are vertices $w_1, w_2, w_3 \in V_c$ such that $v_rw_r \in E(G)$ then we can simply add vertex $w_r$ as well as the edge $v_rw_r$ to $H_r$ in order to cover the color class $V_c$. Assuming we can always find such vertices, this strategy can be repeated going down the tree $F$ until all color classes are covered.

So suppose that there are no such vertices $w_1, w_2, w_3$. By Hall's Marriage Theorem, this means there is a set $V' \subseteq \{v_1, v_2, v_3\}$ such that $|N(V') \cap V_c| < |V'|$. Since $\chi$ is 2-stable, we get that $|N(v) \cap V_c| = |N(v') \cap V_c|$ for all $v, v' \in V_d$. Together, this means there is some $\delta \in \{1, 2\}$ such that $|N(v) \cap V_c| = \delta$ for all $v \in V_d$.

First, suppose that $\delta = 1$. Then $G[V_d, V_c]$ is isomorphic to a disjoint union of $\ell$ stars $K_{1,h}$, for some $\ell \geq 3$ and $h \geq 2$. In this situation, it is possible to contract the connected components of $G[V_d, V_c]$ to single vertices, and proceed by induction. At this point, we crucially exploit that using the 2-dimensional Weisfeiler-Leman algorithm allows us to show that all color classes in the contracted graph still have size at least 3 (such a statement is not true when using the Color Refinement algorithm). By induction, we obtain graphs $H_1', H_2', H_3'$. To obtain the original graphs, we simply uncontract any contracted vertices contained in $H_1', H_2', H_3'$.

In the other case, we have $\delta = 2$. Let us partition $V_d$ into *c-twin-classes* where vertices $v, v' \in V_d$ are declared to be *c-twins* if $N(v) \cap V_c = N(v') \cap V_c$. If there are at least 3 twin-classes, then it is again possible to contract the twin-classes to single vertices and proceed by induction. Here, the crucial observation is that the $c$-twin-classes are non-trivial since $|N(V') \cap V_c| < |V'|$ for some set $V' \subseteq \{v_1, v_2, v_3\}$. The critical case occurs if there are exactly 2 twin-classes meaning that $G[V_d, V_c]$ is isomorphic to a disjoint union of 2 copies of $K_{2,h}$, for some $h \geq 3$ (in this case $|V_c| = 4$). Now, the basic idea is to ensure that $v_1, v_2, v_3$ cover both connected components (which means there are vertices $w_1, w_2, w_3$ as above). However, this additional requirement comes with severe additional complications. First, information of this type needs to be propagated up the tree (i.e., vertices in the root color class may already need to be chosen appropriately to avoid problematic situations later on). But much more problematically, each child of $c$ may add a different restriction which all need to be met at the same time. Here, we again crucially rely on the 2-dimensional Weisfeiler-Leman algorithm to show that all requirements can indeed be met at the same time. Unfortunately, this comes at the price that each vertex of $V_d$ has to be contained in one of the graphs $H_r$, $r \in \{1,2,3\}$ (this allows us to choose different triples $(v_1, v_2, v_3)$ for different children of $d$). To ensure that $H_r$ remains connected, we introduce a second type of restriction that is passed down the tree, and which ensures that all vertices from $V_d$, which are added to $H_r$, end up in the same connected component of $H_r$. By carefully implementing the induction, it can be shown that all these additional requirements can indeed by realized. ◀

**Proof of Lemma 8.** Let $\chi$ be a 2-stable coloring such that $|[v]_\chi| = 1$ for all $v \in D$ and $|[w]_\chi| \geq 3$ for all $w \in V(G) \setminus D$. Suppose towards a contradiction that $|N_G(Z)| \geq h$, and pick $v_1, \dots, v_h \in N_G(Z)$ to be distinct vertices. Let $C \coloneqq \{\chi(v,v) \mid v \in Z\}$ be the set of vertex colors appearing in the set $Z$. Note that $(G[[\chi]])[C]$ is connected, and $|V_c| \geq 3$ for all $c \in C$. Let $W \coloneqq \{w \in V(G) \mid \chi(w,w) \in C\}$. Observe that $W \cap D = \emptyset$. By Lemma 9, there are connected, vertex-disjoint subgraphs $H_1, H_2, H_3 \subseteq G[W]$ such that $V(H_r) \cap V_c \neq \emptyset$ for all $r \in \{1,2,3\}$ and $c \in C$.

Now let $i \in [h]$. Since $v_i \in N_G(Z)$ there is some vertex $w_i \in Z \subseteq W$ such that $v_i w_i \in E(G)$. Let $c_i \coloneqq \chi(w_i, w_i)$. Observe that $c_i \in C$. Also, $V_{c_i} \subseteq N_G(v_i)$ since $|[v_i]_\chi| = 1$ and $\chi$ is 2-stable. This implies that $N_G(v_i) \cap V(H_r) \neq \emptyset$ for all $r \in \{1,2,3\}$, because $V(H_r) \cap V_{c_i} \neq \emptyset$. But this results in a minor isomorphic to $K_{3,h}$ with vertices $v_1, \dots, v_h$ on the right side, and vertices $V(H_1), V(H_2), V(H_3)$ on the left side. ◀

Besides Lemma 8, we also require a second tool which is used to define the extension sets $\gamma(X_i)$ which we needed to ensure the recursive algorithm makes progress.

▶ **Lemma 10.** *Let $G$ be a graph that excludes $K_{3,h}$ as a minor. Also let $X \subseteq V(G)$ and define $D \coloneqq \mathrm{cl}^G_{h-1,1}(X)$. Let $Z$ be a connected component of $G - D$. Then $|N_G(Z)| < 3$.*

The lemma essentially follows from [21, Lemma 23]. For the sake of completeness and due to its simplicity, a complete proof is still given below.

**Proof.** Let $\chi$ be a 1-stable coloring such that $|[v]_\chi| = 1$ for all $v \in D$ and $|[w]_\chi| \geq h$ for all $w \in V(G) \setminus D$. Suppose towards a contradiction that $|N_G(Z)| \geq 3$, and pick $v_1, v_2, v_3 \in N_G(Z)$ to be distinct vertices. Let $C \coloneqq \{\chi(v) \mid v \in Z\}$, and define $H$ to be the graph with $V(H) \coloneqq C$ and

$$E(H) \coloneqq \{c_1 c_2 \mid \exists v_1 \in \chi^{-1}(c_1), v_2 \in \chi^{-1}(v_2) \colon v_1 v_2 \in E(G)\}.$$

Let $T$ be a spanning tree of $H$. Also, for each $i \in \{1,2,3\}$, fix a color $c_i \in C$ such that $N_G(v_i) \cap \chi^{-1}(c_i) \neq \emptyset$. Let $T'$ be the induced subtree obtained from $T$ by repeatedly removing all leaves distinct from $c_1, c_2, c_3$. Finally, let $T''$ be the tree obtained from $T'$ by adding three fresh vertices $v_1, v_2, v_3$ where $v_i$ is connected to $c_i$. Observe that $v_1, v_2, v_3$ are precisely the leaves of $T''$. Now, $T''$ contains a unique node $c$ of degree three (possibly $c = c_i$ for some $i \in \{1,2,3\}$). Observe that $|\chi^{-1}(c)| \geq h$. We define $C_i$ to be the set of all internal vertices which appear on the unique path from $v_i$ to $c$ in the tree $T''$. Finally, define $U_i \coloneqq \{v_i\} \cup \bigcup_{c' \in C_i} \chi^{-1}(c')$.

Since $\chi$ is 1-stable and $|[v_i]_\chi| = 1$ we get that $G[U_i]$ is connected for all $i \in \{1,2,3\}$. Also, $E_G(U_i, \{w\}) \neq \emptyset$ for all $w \in \chi^{-1}(c)$ and $i \in \{1,2,3\}$. But this provides a minor isomorphic to $K_{3,h}$ with vertices $U_1, U_2, U_3$ on the left side and the vertices from $\chi^{-1}(c)$ on the right side. ◀

## 6 A Decomposition Theorem

Next, we use the insights gained in the last section to prove a decomposition theorem for graphs that exclude $K_{3,h}$ as a minor. In the following, all tree decompositions are rooted, i.e., there is a designated root node and we generally assume all edges to be directed away from the root.

▶ **Theorem 11.** *Let $h \geq 3$. Let $G$ be a 3-connected graph, and suppose $S \subseteq V(G)$ such that*
**(A)** *$G - E(S,S)$ excludes $K_{3,h}$ as a minor,*
**(B)** *$3 \leq |S| \leq h$,*
**(C)** *$G - S$ is connected, and*
**(D)** *$S = N_G(V(G) \setminus S)$.*
*Then there is a (rooted) tree decomposition $(T, \beta)$ of $G$, a function $\gamma\colon V(T) \to 2^{V(G)}$, and a vertex-coloring $\lambda$ such that*
**(I)** *$|V(T)| \leq 2 \cdot |V(G)|$,*
**(II)** *the adhesion width of $(T, \beta)$ is at most $h - 1$,*
**(III)** *for every $t \in V(T)$ with children $t_1, \dots, t_\ell$, one of the following options holds:*
  **a.** *$\beta(t) \cap \beta(t_i) \neq \beta(t) \cap \beta(t_j)$ for all distinct $i, j \in [\ell]$, or*
  **b.** *$\beta(t) = \beta(t) \cap \beta(t_i)$ for all $i \in [\ell]$,*
**(IV)** *$S \subsetneq \gamma(r)$ where $r$ denotes the root of $T$,*
**(V)** *$|\gamma(t)| \leq h^4$ for every $t \in V(T)$,*
**(VI)** *$\beta(t) \cap \beta(s) \subseteq \gamma(t) \subseteq \beta(t)$ for all $t \in V(T) \setminus \{r\}$, where $s$ denotes the parent of $t$, and*
**(VII)** *$\beta(t) \subseteq \mathrm{cl}_{2,2}^{(G,\lambda)}(\gamma(t))$ for all $t \in V(T)$.*
*Moreover, the decomposition $(T, \beta)$, the function $\gamma$, and the coloring $\lambda$ can be computed in polynomial time, and the output is isomorphism-invariant with respect to $(G, S, h)$.*

**Proof.** We give an inductive construction for the tree decomposition $(T, \beta)$ as well as the function $\gamma$ and the coloring $\lambda$. We start by arguing how to compute the set $\gamma(r)$.

▷ Claim 12. Let $v_1, v_2, v_3 \in S$ be three distinct vertices, and define $\chi := \chi^1_{\mathsf{WL}}[G, S, v_1, v_2, v_3]$. Then there exists some $v \in V(G) \setminus S$ such that $|[v]_\chi| < h$.

Proof. Let $H := (G - (S \setminus \{v_1, v_2, v_3\})) - E(\{v_1, v_2, v_3\}, \{v_1, v_2, v_3\})$. It is easy to see that $\chi|_{V(H)}$ is 1-stable for the graph $H$. Observe that $H - \{v_1, v_2, v_3\} = G - S$ is connected. Suppose there is no vertex $v \in V(G) \setminus S$ such that $|[v]_\chi| < h$. Then $\chi$ is $(h-1)$-CR-stable which implies that $\mathrm{cl}^G_{h-1,1}(v_1, v_2, v_3) = \{v_1, v_2, v_3\}$. On the other hand, $Z := V(H) \setminus \{v_1, v_2, v_3\}$ induces a connected component of $H - \{v_1, v_2, v_3\}$, and $N_H(Z) = \{v_1, v_2, v_3\}$ since $S = N_G(V(G) \setminus S)$. But this contradicts Lemma 10. ◁

Let $v_1, v_2, v_3 \in S$ be distinct. We define $\chi[v_1, v_2, v_3] := \chi^1_{\mathsf{WL}}[G, S, v_1, v_2, v_3]$. Also, let $c[v_1, v_2, v_3]$ denote the unique color such that
1. $c[v_1, v_2, v_3] \notin \{\chi[v_1, v_2, v_3](v) \mid v \in S\}$, and
2. $|(\chi[v_1, v_2, v_3])^{-1}(c[v_1, v_2, v_3])| \leq h - 1$

and which is minimal with respect to the linear order on the colors in the image of $\chi[v_1, v_2, v_3]$. Let $\gamma(v_1, v_2, v_3) \coloneqq (\chi[v_1, v_2, v_3])^{-1}(c[v_1, v_2, v_3])$. Observe that $\gamma(v_1, v_2, v_3)$ is defined in an isomorphism-invariant manner given $(G, S, h, v_1, v_2, v_3)$. Now, define

$$\gamma(r) \coloneqq S \cup \bigcup_{v_1,v_2,v_3 \in S \text{ distinct}} \gamma(v_1, v_2, v_3).$$

Clearly, $\gamma(r)$ is defined in an isomorphism-invariant manner given $(G, S, h)$. Moreover,

$$|\gamma(r)| \leq |S| + |S|^3 \cdot (h - 1) \leq |S|^3 \cdot h \leq h^4.$$

Finally, define $\beta(r) \coloneqq \mathrm{cl}^G_{2,2}(\gamma(r))$.

Let $Z_1, \dots, Z_\ell$ be the connected components of $G - \beta(r)$. Also, let $S_i \coloneqq N_G(Z_i)$ and $G_i$ be the graph obtained from $G[S_i \cup Z_i]$ by turning $S_i$ into a clique, $i \in [\ell]$. We have $|S_i| < h$ by Lemma 8. Also, $|S_i| \geq 3$ and $G_i$ is 3-connected since $G$ is 3-connected. Clearly, $G_i - S_i$ is connected and $S_i = N_{G_i}(V(G_i) \setminus S_i)$. Finally, $G_i - E(S_i, S_i)$ excludes $K_{3,h}$ as a minor because $G - E(S, S)$ excludes $K_{3,h}$ as a minor.

We wish to apply the induction hypothesis to the triples $(G_i, S_i, h)$. If $|V(G_i)| = |V(G)|$ then $\ell = 1$ and $S \subsetneq S_i$. In this case the algorithm still makes progress since the size of $S$ can be increased at most $h - 3$ times.

By the induction hypothesis, there are tree decompositions $(T_i, \beta_i)$ of $G_i$ and functions $\gamma_i \colon V(T_i) \to 2^{V(G_i)}$ satisfying Properties I - VII. We define $(T, \beta)$ to be the tree decomposition where $T$ is obtained from the disjoint union of $T_1, \dots, T_\ell$ by adding a fresh root vertex $r$ which is connected to the root vertices of $T_1, \dots, T_\ell$. Also, $\beta(r)$ is defined as above and $\beta(t) \coloneqq \beta_i(t)$ for all $t \in V(T_i)$ and $i \in [\ell]$. Finally, $\gamma(r)$ is again defined as above, and $\gamma(t) \coloneqq \gamma_i(t)$ for all $t \in V(T_i)$ and $i \in [\ell]$.

The algorithm clearly runs in polynomial time and the output is isomorphism-invariant (the coloring $\lambda$ is defined below). We need to verify that Properties I - VII are satisfied. Using the comments above and the induction hypothesis, it is easy to verify that Properties II, IV, V and VI are satisfied.

For Property VII it suffices to ensure that $\mathrm{cl}^{(G_i,\lambda)}_{2,2}(\gamma(t)) \subseteq \mathrm{cl}^{(G,\lambda)}_{2,2}(\gamma(t))$. Towards this end, it suffices to ensure that $\lambda(v) \neq \lambda(w)$ for all $v \in \beta(r)$ and $w \in V(G) \setminus \beta(r)$. To ensure this property holds on all levels of the tree, we can simply define $\lambda(v) \coloneqq \{\mathrm{dist}_T(r, t) \mid t \in V(T), v \in \beta(t)\}$.

Next, we modify the tree decomposition in order to ensure Property III. Consider a node $t \in V(T)$ with children $t_1, \dots, t_\ell$. We say that $t_i \sim t_j$ if $\beta(t) \cap \beta(t_i) = \beta(t) \cap \beta(t_j)$. Let $A_1, \dots, A_k$ be the equivalence classes of the equivalence relation $\sim$. For every $i \in [k]$ we introduce a fresh node $s_i$. Now, every $t_j \in A_i$ becomes a child of $s_i$ and $s_i$ becomes a child of $t$. Finally, we set $\beta(s_i) = \gamma(s_i) \coloneqq \beta(t) \cap \beta(t_j)$ for some $t_j \in A_i$. Observe that after this modification, Properties II and IV - VII still hold.

Finally, it remains to verify Property I. Before the modification described in the last paragraph, we have that $|V(T)| \leq |V(G)|$. Since the modification process at most doubles the number of nodes in $T$, the bound follows. ◀

## 7 An FPT Isomorphism Test for Graphs of Small Genus

Building on the decomposition theorem given in the last section, we can now prove the main result of this paper.

▶ **Theorem 13.** *Let $G_1, G_2$ be two (vertex- and arc-colored) graphs that exclude $K_{3,h}$ as a minor. Then one can decide whether $G_1$ is isomorphic to $G_2$ in time $2^{\mathcal{O}(h^4 \log h)} n^{\mathcal{O}(1)}$.*

**Proof Idea.** Suppose $G_i = (V(G_i), E(G_i), \chi_V^i, \chi_E^i)$ for $i \in \{1,2\}$. Using standard reduction techniques (see, e.g., [11]) we may assume without loss generality that $G_1$ and $G_2$ are 3-connected. Pick an arbitrary set $S_1 \subseteq V(G_1)$ such that $|S_1| = 3$ and $G_1 - S_1$ is connected. For every $S_2 \subseteq V(G_2)$ such that $|S_2| = 3$ and $G_2 - S_2$ is connected, the algorithm tests whether there is an isomorphism $\varphi\colon G_1 \cong G_2$ such that $S_1^\varphi = S_2$. Observe that $S_i = N_{G_i}(V(G_i) \setminus S_i)$ for both $i \in \{1,2\}$ since $G_1$ and $G_2$ are 3-connected. This implies that the triple $(G_i, S_i, h)$ satisfies the requirements of Theorem 11. Let $(T_i, \beta_i)$ be the tree decomposition, $\gamma_i\colon V(T_i) \to 2^{V(G_i)}$ be the function, and $\lambda_i$ be the vertex-coloring computed by Theorem 11 on input $(G_i, S_i, h)$.

Now, the basic idea is compute isomorphisms between $(G_1, S_1)$ and $(G_2, S_2)$ using dynamic programming along the tree decompositions. More precisely, we aim at recursively computing the set

$$\Lambda \coloneqq \mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2))[S_1]$$

(here, $\mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2))$ denotes the set of isomorphisms $\varphi\colon G_1 \cong G_2$ which additionally respect the vertex-colorings $\lambda_i$ and satisfy $S_1^\varphi = S_2$). Throughout the recursive algorithm, we maintain the property that $|S_i| \leq h$. Also, we may assume without loss of generality that $S_i$ is $\lambda_i$-invariant (otherwise, we replace $\lambda_i$ by $\lambda_i'$ defined via $\lambda_i'(v) \coloneqq (1, \lambda_i(v))$ for all $v \in S_i$, and $\lambda_i'(v) \coloneqq (0, \lambda_i(v))$ for all $v \in V(G_i) \setminus S_i$).

Let $r_i$ denote the root node of $T_i$. Let $\ell$ denote the number of children of $r_i$ in the tree $T_i$ (if the number of children is not the same, the algorithm concludes that $\mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2)) = \emptyset$). Let $t_1^i, \dots, t_\ell^i$ be the children of $r_i$ in $T_i$, $i \in \{1,2\}$. For $i \in \{1,2\}$ and $j \in [\ell]$ let $V_j^i$ denote the set of vertices appearing in bags below (and including) $t_j^i$. Also let $S_j^i \coloneqq \beta_i(r_i) \cap \beta_i(t_j^i)$ be the adhesion set to the $j$-th child, and define $G_j^i \coloneqq G_i[V_j^i]$. Finally, let $T_j^i$ denote the subtree of $T_i$ rooted at node $t_j^i$, and $\beta_j^i \coloneqq \beta_i|_{V(T_j^i)}$, $\gamma_j^i \coloneqq \gamma_i|_{V(T_j^i)}$ and $\lambda_j^i \coloneqq \lambda_i|_{V_j^i}$.

For every $i, i' \in \{1,2\}$, and every $j, j' \in [\ell]$, the algorithm recursively computes the set

$$\Lambda_{j,j'}^{i,i'} \coloneqq \mathrm{Iso}((G_j^i, \lambda_j^i, S_j^i), (G_{j'}^{i'}, \lambda_{j'}^{i'}, S_{j'}^{i'}))[S_j^i].$$

We argue how to compute the set $\Lambda$. Building on Theorem 11, Item III, we may assume that

**(a)** $S_j^i \neq S_{j'}^i$ for all distinct $j, j' \in [\ell]$ and $i \in \{1,2\}$, or
**(b)** $\beta(r_i) = S_j^i$ for all $j \in [\ell]$ and $i \in \{1,2\}$

(if $r_1$ and $r_2$ do not satisfy the same option, then $\mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2)) = \emptyset$).

We first cover Option b. In this case $|\beta(r_i)| = |S_j^i| \leq h - 1$ by Theorem 11, Item II. The algorithm iterates over all bijections $\sigma\colon \beta(r_1) \to \beta(r_2)$. Now,

$$\sigma \in \mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2))[\beta(r_1)] \quad \Leftrightarrow \quad \exists \rho \in \mathrm{Sym}([\ell])\ \forall j \in [\ell]\colon \sigma \in \Lambda_{j,\rho(j)}^{1,2}.$$

To test whether $\sigma$ satisfies the right-hand side condition, the algorithm constructs an auxiliary graph $H_\sigma$ with vertex set $V(H_\sigma) \coloneqq \{1,2\} \times [\ell]$ and edge set

$$E(H_\sigma) \coloneqq \{(1,j)(2,j') \mid \sigma \in \Lambda_{j,j'}^{1,2}\}.$$

Observe that $H_\sigma$ is bipartite with bipartition $(\{1\} \times [\ell], \{2\} \times [\ell])$. Now,

$$\sigma \in \mathrm{Iso}((G_1, \lambda_1, S_1), (G_2, \lambda_1, S_2))[\beta(r_1)] \quad \Leftrightarrow \quad H_\sigma \text{ has a perfect matching.}$$

It is well-known that the latter can be checked in polynomial time. This completes the description of the algorithm in case Option b is satisfied.

Next, suppose Option a is satisfied. Here, the central idea is to construct auxiliary vertex- and arc-colored graphs $H_i = (V(H_i), E(H_i), \mu_V^i, \mu_E^i)$ and sets $A_i \subseteq V(H_i)$ such that

**1.** $\beta_i(r_i) \subseteq A_i$ and $A_i \subseteq \mathrm{cl}_{2,2}^{H_i}(\gamma_i(r_i))$, and

**2.** $\mathrm{Iso}(H_1, H_2)[S_1] = \mathrm{Iso}(H_1[A_1], H_2[A_2])[S_1] = \Lambda$.

Towards this end, we set

$$V(H_i) \coloneqq V(G_i) \uplus \{(S_j^i, \gamma) \mid j \in [\ell], \gamma \in \Lambda_{j,j}^{i,i}\}$$

and

$$E(H_i) \coloneqq E(G_i) \cup \{(S_j^i, \gamma)v \mid j \in [\ell], \gamma \in \Lambda_{j,j}^{i,i}, v \in S_j^i\}.$$

Also, we set

$$A_i \coloneqq \beta(r_i) \cup \{(S_j^i, \gamma) \mid j \in [\ell], \gamma \in \Lambda_{j,j}^{i,i}\}.$$

The main idea is to use the additional vertices attached to the set $S_j^i$ to encode the isomorphism type of the graph $(G_j^i, \lambda_j^i, S_j^i)$. This information is encoded by the vertex- and arc-coloring building on sets $\Lambda_{j,j'}^{i,i'}$ already computed above. Let $\mathcal{S} \coloneqq \{S_j^i \mid i \in \{1,2\}, j \in [\ell]\}$, and define $S_j^i \sim S_{j'}^{i'}$ if $\Lambda_{j,j'}^{i,i'} \neq \emptyset$. Observe that $\sim$ is an equivalence relation. Let $\{\mathcal{P}_1, \dots, \mathcal{P}_k\}$ be the partition of $\mathcal{S}$ into the equivalence classes. We set

$$\mu_V^i(v) \coloneqq (0, \chi_V^i(v), \lambda_i(v))$$

for all $v \in S_i$,

$$\mu_V^i(v) \coloneqq (1, \chi_V^i(v), \lambda_i(v))$$

for all $v \in \gamma_i(r_i) \setminus S_i$,

$$\mu_V^i(v) \coloneqq (2, \chi_V^i(v), \lambda_i(v))$$

for all $v \in \beta_i(r_i) \setminus \gamma_i(r_i)$,

$$\mu_V^i(v) \coloneqq (3, \chi_V^i(v), \lambda_i(v))$$

for all $v \in V(G_i) \setminus \beta_i(r_i)$, and

$$\mu_V^i(S_j^i, \gamma) \coloneqq (4, q, q)$$

for all $q \in [k]$, $S_j^i \in \mathcal{P}_q$, and $\gamma \in \Lambda_{j,j}^{i,i}$. For every $q \in [k]$ fix some $i(q) \in \{1,2\}$ and $j(q) \in [\ell]$ such that $S_{j(q)}^{i(q)} \in \mathcal{P}_q$ (i.e., for each equivalence class, the algorithm fixes one representative). Also, for every $q \in [k]$ and $S_j^i \in \mathcal{P}_q$, fix a bijection $\sigma_j^i \in \Lambda_{j(q),j}^{i(q),i}$ such that $\sigma_{j(q)}^{i(q)}$ is the identity mapping. Finally, for $q \in [k]$, fix a numbering $S_{j(q)}^{i(q)} = \{u_1^q, \dots, u_{s(q)}^q\}$.

With this, we are ready to define the arc-coloring $\mu_E^i$. First, we set

$$\mu_E^i(v, w) \coloneqq (0, \chi_E^i(v, w))$$

for all $vw \in E(G_i)$. Next, consider an edge $(S_j^i, \gamma)v$ where $j \in [\ell]$, $\gamma \in \Lambda_{j,j}^{i,i}$, and $v \in S_j^i$. Suppose $S_j^i \in \mathcal{P}_q$. We set

$$\mu_E^i(v, (S_j^i, \gamma)) = \mu_E^i((S_j^i, \gamma), v) \coloneqq (1, c)$$

for the unique $c \in [s(q)]$ such that

$$v = (u_c^q)^{\sigma_j^i \gamma}.$$

This completes the description of the graphs $H_i$ and the sets $A_i$, $i \in \{1,2\}$. It can be checked that Properties 1 and 2 are satisfied (see the full version for details).

Now, recall that the algorithm aims at computing the set $\Lambda$. Building on Property 2, we can simply compute $\mathrm{Iso}(H_1[A_1], H_2[A_2])[S_1]$. Towards this end, the algorithm iterates through all bijections $\tau\colon \gamma_1(r_1) \to \gamma_2(r_2)$, and wishes to test whether there is an isomorphism $\varphi \in \mathrm{Iso}(H_1[A_1], H_2[A_2])$ such that $\varphi[\gamma_1(r_1)] = \tau$. Note that, since $\gamma_i(r_i)$ is $\mu_V^i$-invariant, it now suffices to solve this latter problem.

So fix a bijection $\tau\colon \gamma_1(r_1) \to \gamma_2(r_2)$ (if $|\gamma_1(r_1)| \neq |\gamma_2(r_2)|$ then the algorithm returns $\Lambda = \emptyset$). Let $\mu_1^*(v) := (1, v)$ for all $v \in \gamma_1(r_1)$, $\mu_1^*(v) := (0, \mu_V^1)$ for all $v \in V(H_1) \setminus \gamma_1(r_1)$, and $\mu_2^*(v) := (1, \tau^{-1}(v))$ for all $v \in \gamma_2(r_2)$ and $\mu_2^*(v) := (0, \mu_V^2)$ for all $v \in V(H_2) \setminus \gamma_2(r_2)$.

Intuitively speaking, $\mu_1^*$ and $\mu_2^*$ are obtained from $\mu_V^1$ and $\mu_V^2$ by individualizing all vertices from $\gamma_1(r_1)$ and $\gamma_r(r_2)$ according to the bijection $\tau$. Now, we can apply Theorem 6 on input graph $H_1^* = (H_1, \mu_1^*)$ and $H_2^* = (H_2, \mu_2^*)$, and parameters $t = k := 2$.

Building on Property 1, we obtain a $\widehat{\Gamma}_2$-group $\Gamma \leq \mathrm{Sym}(A_1)$ and a bijection $\theta\colon A_1 \to A_2$ such that $\mathrm{Iso}(H_1^*[A_1], H_2^*[A_2]) \subseteq \Gamma\theta$. Now, we can determine whether $H_1^*[A_1] \cong H_2^*[A_2]$ using Theorem 3. Using Property 2, this provides the answer to whether $\tau[S_1] \in \Lambda$ (recall that $S_1 \subseteq \gamma_1(r_1)$ by Theorem 11, Items IV and VI).

Overall, this completes the description of the algorithm. It only remains to analyse its running time. Let $n$ denote the number of vertices of $G_1$ and $G_2$.

The algorithm iterates over at most $n^3$ choices for the initial set $S_2$, and computes the decompositions $(T_i, \beta_i)$, the functions $\gamma_i$, and the colorings $\lambda_i$ in polynomial time. For the dynamic programming tables, the algorithm needs to compute $\mathcal{O}(n^2)$ many $\Lambda$-sets (using Theorem 11, Item I), each of which contains at most $h! = 2^{\mathcal{O}(h \log h)}$ many elements by Theorem 11, Item II. Hence, it remains to analyse the time required to compute the set $\Lambda$ given the $\Lambda_{j,j'}^{i,i'}$-sets. For Option b, this can clearly be done in time $2^{\mathcal{O}(h \log h)} n^{\mathcal{O}(1)}$.

So consider Option a. The graph $H_i$ can clearly be computed in time polynomial in its size. We have that $|V(H_i)| = 2^{\mathcal{O}(h \log h)} n$. Afterwards, the algorithm iterates over $|\gamma_1(r_1)|!$ many bijections $\tau$. By Theorem 11, Item V, we have that $|\gamma_1(r_1)|! = 2^{\mathcal{O}(h^4 \log h)}$. For each bijection, the algorithm then requires polynomial computation time by Theorems 6 and 3. Overall, this proves the bound on the running time. ◀

▶ Remark 14. The algorithm from the last theorem can be extended in two directions. First, if one of the input graphs does not exclude $K_{3,h}$ as a minor, it can modified to either correctly conclude that $G_1$ has a minor isomorphic to $K_{3,h}$, or to correctly decide whether $G_1$ is isomorphic to $G_2$. Indeed, the only part of the algorithm that exploits that the input graphs do not have minor isomorphic to $K_{3,h}$ is the computation of the tree decompositions $(T_i, \beta_i)$ from Theorem 11. In turn, this theorem only exploits forbidden minors via Lemmas 8 and 10. An algorithm can easily detect if one of the implications of those two statements is violated, in which case it can infer the existence of a minor $K_{3,h}$.

Secondly, using standard reduction techniques (see, e.g., [19]), one can also compute a representation of the set of all isomorphisms $\mathrm{Iso}(G_1, G_2)$ in the same time.

Since every graph of Euler genus $g$ excludes $K_{3,4g+3}$ as a minor [25], we obtain the following corollary.

▶ **Corollary 15.** *Let $G_1, G_2$ be two (vertex- and arc-colored) graphs of Euler genus at most $g$. Then one can decide whether $G_1$ is isomorphic to $G_2$ in time $2^{\mathcal{O}(g^4 \log g)} n^{\mathcal{O}(1)}$.*

## 8 Conclusion

We presented an isomorphism test for graphs excluding $K_{3,h}$ as a minor running in time $2^{\mathcal{O}(h^4 \log h)} n^{\mathcal{O}(1)}$. For this, we provided a polynomial-time isomorphism algorithm for $(t,k)$-WL-bounded graphs and argued that graphs excluding $K_{3,h}$ as a minor can be decomposed into parts that are $(2,2)$-WL-bounded after individualizing a small number of vertices.

Still, several questions remain open. Probably one of the most important questions in the area is whether isomorphism testing for graphs excluding $K_h$ as a minor is fixed-parameter tractable with parameter $h$. As graphs of bounded genus form an important subclass of graphs excluding $K_h$ as a minor, the techniques developed in this paper might also prove helpful in resolving this question.

As an intermediate step, one can also ask for an isomorphism test for graphs excluding $K_{\ell,h}$ as a minor running in time $f(h,\ell)n^{g(\ell)}$ for some functions $f,g$. Observe that this paper provides such an algorithm for $\ell = 3$. Indeed, combining ideas from [10, 22] with the approach taken in this paper, it seems the only hurdle towards such an algorithm is a generalization of Lemma 9. Given a connected graph $G$ for which $|V_c| \geq \ell$ for all $c \in C_V(G, \chi^2_{\mathsf{WL}}[G])$, is it always possible to find vertex-disjoint, connected subgraphs $H_1, \dots, H_\ell \subseteq G$ such that $V(H_r) \cap V_c \neq \emptyset$ for all $r \in [\ell]$ and $c \in C_V(G, \chi^2_{\mathsf{WL}}[G])$?

As another intermediate problem, one can also consider the class $\mathcal{G}_h$ of all graphs $G$ for which there is a set $X \subseteq V(G)$ of size $|X| \leq h$ such that $G - X$ is planar. Is isomorphism testing fixed-parameter tractable on $\mathcal{G}_h$ parameterized by $h$?

## References


1. László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 684–697. ACM, 2016. doi:10.1145/2897518.2897542.
2. László Babai, Peter J. Cameron, and Péter P. Pálfy. On the orders of primitive groups with restricted nonabelian composition factors. *J. Algebra*, 79(1):161–168, 1982. doi:10.1016/0021-8693(82)90323-4.
3. László Babai, William M. Kantor, and Eugene M. Luks. Computational complexity and the classification of finite simple groups. In *24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, USA, 7-9 November 1983*, pages 162–171. IEEE Computer Society, 1983. doi:10.1109/SFCS.1983.10.
4. Hans L. Bodlaender. Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees. *J. Algorithms*, 11(4):631–643, 1990. doi:10.1016/0196-6774(90)90013-5.
5. Jin-yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Comb.*, 12(4):389–410, 1992. doi:10.1007/BF01305232.
6. John D. Dixon and Brian Mortimer. *Permutation Groups*, volume 163 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1996. doi:10.1007/978-1-4612-0731-3.
7. Martin Grohe and Dániel Marx. Structure theorem and isomorphism test for graphs with excluded topological subgraphs. *SIAM J. Comput.*, 44(1):114–159, 2015. doi:10.1137/120892234.
8. Martin Grohe, Daniel Neuen, Pascal Schweitzer, and Daniel Wiebking. An improved isomorphism test for bounded-tree-width graphs. *ACM Trans. Algorithms*, 16(3):34:1–34:31, 2020. doi:10.1145/3382082.
9. Martin Grohe and Pascal Schweitzer. Isomorphism testing for graphs of bounded rank width. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1010–1029. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.66.

**10** Martin Grohe, Daniel Wiebking, and Daniel Neuen. Isomorphism testing for graphs excluding small minors. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 625–636. IEEE, 2020. `doi:10.1109/FOCS46700.2020.00064`.

**11** John E. Hopcroft and Robert Endre Tarjan. Isomorphism of planar graphs. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 131–152. Plenum Press, New York, 1972. `doi:10.1007/978-1-4684-2001-2_13`.

**12** Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. In Alan L. Selman, editor, *Complexity Theory Retrospective: In Honor of Juris Hartmanis on the Occasion of His Sixtieth Birthday, July 5, 1988*, pages 59–81. Springer New York, New York, NY, 1990. `doi:10.1007/978-1-4612-4478-3_5`.

**13** Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. `doi:10.1007/978-1-4684-2001-2_9`.

**14** Ken-ichi Kawarabayashi. Graph isomorphism for bounded genus graphs in linear time. *CoRR*, abs/1511.02460, 2015. `arXiv:1511.02460`.

**15** Ken-ichi Kawarabayashi and Bojan Mohar. Graph and map isomorphism and all polyhedral embeddings in linear time. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 471–480. ACM, 2008. `doi:10.1145/1374376.1374443`.

**16** Ken-ichi Kawarabayashi, Bojan Mohar, Roman Nedela, and Peter Zeman. Automorphism groups of maps in linear time. *CoRR*, abs/2008.01616, 2020. `arXiv:2008.01616`.

**17** Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Fixed-parameter tractable canonization and isomorphism test for graphs of bounded treewidth. *SIAM J. Comput.*, 46(1):161–189, 2017. `doi:10.1137/140999980`.

**18** Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.*, 25(1):42–65, 1982. `doi:10.1016/0022-0000(82)90009-5`.

**19** Rudolf Mathon. A note on the graph isomorphism counting problem. *Inf. Process. Lett.*, 8(3):131–132, 1979. `doi:10.1016/0020-0190(79)90004-8`.

**20** Gary L. Miller. Isomorphism of graphs which are pairwise k-separable. *Inf. Control.*, 56(1/2):21–33, 1983. `doi:10.1016/S0019-9958(83)80048-5`.

**21** Daniel Neuen. Hypergraph isomorphism for groups with restricted composition factors. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 88:1–88:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. `doi:10.4230/LIPIcs.ICALP.2020.88`.

**22** Daniel Neuen. Isomorphism testing for graphs excluding small topological subgraphs. *CoRR*, abs/2011.14730, 2020. `arXiv:2011.14730`.

**23** Ilia N. Ponomarenko. The isomorphism problem for classes of graphs. *Dokl. Akad. Nauk SSSR*, 304(3):552–556, 1989.

**24** Ilia N. Ponomarenko. The isomorphism problem for classes of graphs closed under contraction. *Journal of Soviet Mathematics*, 55(2):1621–1643, June 1991. `doi:10.1007/BF01098279`.

**25** Gerhard Ringel. Das geschlecht des vollständigen paaren graphen. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 28(3):139–150, October 1965. `doi:10.1007/BF02993245`.

**26** Joseph J. Rotman. *An Introduction to the Theory of Groups*, volume 148 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, fourth edition, 1995. `doi:10.1007/978-1-4612-4176-8`.

27 Ákos Seress. *Permutation Group Algorithms*, volume 152 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, 2003. doi:10.1017/CBO9780511546549.

28 Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *NTI, Series 2*, 1968. English translation by Grigory Ryabov available at https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf.