Subgraph Isomorphism on Graph Classes that Exclude a Substructure

Citation for published version (APA):

Bodlaender, H. L., Hanaka, T., Kobayashi, Y., Kobayashi, Y., Okamoto, Y., Otachi, Y., & van der Zanden, T. C. (2020). Subgraph Isomorphism on Graph Classes that Exclude a Substructure. *Algorithmica*, *82*(12), 3566-3587. https://doi.org/10.1007/s00453-020-00737-z

Document status and date:
Published: 01/12/2020

DOI:
10.1007/s00453-020-00737-z

Document Version:
Publisher's PDF, also known as Version of record

Document license:
Taverne

Algorithmica (2020) 82:3566–3587
https://doi.org/10.1007/s00453-020-00737-z

Subgraph Isomorphism on Graph Classes that Exclude a Substructure

Hans L. Bodlaender[1] · Tesshu Hanaka[2] · Yasuaki Kobayashi[3] · Yusuke Kobayashi[3] · Yoshio Okamoto[4,5] · Yota Otachi[6] · Tom C. van der Zanden[7]

Received: 28 June 2019 / Accepted: 13 June 2020 / Published online: 2 July 2020

Abstract
We study Subgraph Isomorphism on graph classes defined by a fixed forbidden graph. Although there are several ways for forbidding a graph, we observe that it is reasonable to focus on the minor relation since other well-known relations lead to either trivial or equivalent problems. When the forbidden minor is connected, we present a near dichotomy of the complexity of Subgraph Isomorphism with respect to the forbidden minor, where the only unsettled case is P_5, the path of five vertices. We then also consider the general case of possibly disconnected forbidden minors. We show fixed-parameter tractable cases and randomized XP-time solvable cases parameterized by the size of the forbidden minor H. We also show that by slightly generalizing the tractable cases, the problem becomes NP-complete. All unsettle cases are equivalent to P_5 or the disjoint union of two P_5's. As a byproduct, we show that Subgraph Isomorphism is fixed-parameter tractable parameterized by vertex integrity. Using similar techniques, we also observe that Subgraph Isomorphism is fixed-parameter tractable parameterized by neighborhood diversity.

Keywords Subgraph isomorphism · Minor-free graphs · Parameterized complexity

1 Introduction

Let Q and G be graphs. A *subgraph isomorphism* η is an injection from $V(Q)$ to $V(G)$ that preserves the adjacency in Q; that is, if $\{u, v\} \in E(Q)$, then $\{\eta(u), \eta(v)\} \in E(G)$. We say that Q is *subgraph-isomorphic* to G if there is a subgraph isomorphism from Q to G, and write $Q \preceq G$. In this paper, we study the following problem of deciding the existence of a subgraph isomorphism.

✉ Yota Otachi
otachi@nagoya-u.jp

Extended author information available on the last page of the article

Subgraph Isomorphism
Input: Two graphs G (the *host* graph) and Q (the *pattern* graph).
Question: $Q \preceq G$?

The problem Subgraph Isomorphism is one of the most general and fundamental graph problems and generalizes many other graph problems such as Graph Isomorphism, Clique, Hamiltonian Path/Cycle, and Bandwidth. Obviously, Subgraph Isomorphism is NP-complete in general. When both host and pattern graphs are restricted to be in a graph class $\mathcal{C}$, we call the problem Subgraph Isomorphism *on* $\mathcal{C}$. By slightly modifying known reductions in [7, 15], one can easily show that the problem is hard even for very restricted graph classes such as linear forests and cluster graphs. (See Proposition 1.1.)

Since most of the well-studied graph classes contain all linear forests or all cluster graphs, it is often hopeless to have a polynomial-time algorithm for an interesting graph class. This is sometimes true even if we further assume that the graphs are connected [20, 22]. On the other hand, it is polynomial-time solvable for trees [29]. This result was first generalized for 2-connected outerplanar graphs [25], and finally for k-connected partial k-trees [16, 28] (where the running time is XP parameterized by k). In [28], a polynomial-time algorithm for partial k-trees of bounded maximum degree is presented as well, which is later generalized to partial k-trees of log-bounded fragmentation [17]. When the pattern graph has constant treewidth, the celebrated color-coding technique [1] gives a fixed-parameter algorithm parameterized by the size of the pattern graph. It is also known that for chain graphs, co-chain graphs, and threshold graphs, Subgraph Isomorphism is polynomial-time solvable [20–22]. In the case where only the pattern graph has to be in a restricted graph class that is closed under vertex deletions, a complexity dichotomy with respect to the graph class is known [18].

Because of its unavoidable hardness in the general case, it is often assumed that the pattern graph is small. In such a setting, we can study the parameterized complexity[1] of Subgraph Isomorphism parameterized by the size of the pattern graph. Unfortunately, the W[1]-completeness of Clique [9] implies that this parameterization does not help in general. Indeed, the existence of a $2^{o(n \log n)}$-time algorithm for Subgraph Isomorphism is ruled out assuming the Exponential Time Hypothesis, where n is the total number of vertices [5]. So we need further restrictions on the considered graph classes even in the parameterized setting. For planar graphs, it is known to be fixed-parameter tractable [8, 11]. This result is later generalized to graphs of bounded genus [4]. For several graph parameters, the parameterized complexity of Subgraph Isomorphism parameterized by combinations of them is

[1] We assume that the readers are familiar with the concept of parameterized complexity. See e.g. [6] for basic definitions omitted here.

determined in [27]. In [3], it is shown that when the pattern graph excludes a fixed graph as a minor, the problem is fixed-parameter tractable parameterized by tree-width and the size of the pattern graph. The result in [3] implies also that Subgraph Isomorphism can be solved in subexponential time when the host graph also excludes a fixed graph as a minor.

1.1 Our Results

As mentioned above, the research on Subgraph Isomorphism has been done mostly when the size of the pattern graph is considered as a parameter. However, in this paper, we are going to study the general case where the pattern graph can be as large as the host graph.

We denote the path of n vertices by P_n, the complete graph of n vertices by K_n, and the star with ℓ leaves by $K_{1,\ell}$. The disjoint union of graphs X and Y is denoted by $X \cup Y$ and the disjoint union of k copies of a graph Z is denoted by kZ. The complement of a graph X is denoted by $\overline{X}$.

We believe the following fact is folklore but give a proof to be self-contained. Recall that a *linear forest* is the disjoint union of paths and a *cluster graph* is the disjoint union of complete graphs.

Proposition 1.1 Subgraph Isomorphism *is NP-complete on linear forests and on cluster graphs even if the input graphs have the same number of vertices.*

Proof Since the problem is clearly in NP, we show the NP-hardness. Recall that 3-Partition is the following problem: the input is $3m$ positive integers $a_1, \dots, a_{3m}$ with $\sum_{1 \le i \le 3m} a_i = mB$ such that $B/4 < a_i < B/2$ for all i. The task is to decide whether there is a partition of $a_1, \dots, a_{3m}$ into m triplets such that the members of each triplet sum up to B. The problem is known to be strongly NP-complete; that is, it is NP-complete even if a polynomial in m upperbounds all a_i [15].

To show the hardness on linear forests, we set the host graph G to be mP_B and the pattern graph Q to be $P_{a_1} \cup \cdots \cup P_{a_{3m}}$. Similarly, to show the hardness on cluster graphs, we set $G = mK_B$ and $Q = K_{a_1} \cup \cdots \cup K_{a_{3m}}$. It is straightforward to show that $Q \preceq G$ if and only if the corresponding instance of 3-Partition is a yes-instance. □

Our first observation is that forbidding a graph as an induced substructure (an induced subgraph, an induced topological minor, or an induced minor) does not help for making Subgraph Isomorphism tractable unless we make the graph class trivial. This can be done just by combining some easy observations and known results.

Observation 1.2 *Let $\mathcal{C}$ be the graph class that forbids a fixed graph H as either an induced subgraph, an induced topological minor, or an induced minor. Then,* Subgraph Isomorphism *on $\mathcal{C}$ is polynomial-time solvable if H has at most two vertices; otherwise, it is NP-complete.*

Proof We assume that H is a linear forest since otherwise it is NP-complete by Lemma 2.1. Hence, we may assume that H is forbidden as an induced subgraph as it is equivalent to the other cases.

If $H = 3K_1$, then Clique is NP-complete on $\mathcal{C}$ because Independent Set is NP-complete on triangle-free graphs [33]. If $H = P_3$, then $\mathcal{C}$ is the class of cluster graphs and thus Subgraph Isomorphism on $\mathcal{C}$ is NP-complete by Proposition 1.1. If $H = K_2 \cup K_1 = \overline{P_3}$, then $\mathcal{C}$ is the class of co-cluster graphs (or complete multi-partite graphs). It is known that if $|V(G)| = |V(Q)|$, then $Q \preceq G$ if and only if $\overline{G} \preceq \overline{Q}$ [20]. Thus, by Proposition 1.1, this case is NP-complete.

In the remaining cases, we can assume that H has order at most 2 since all other cases are NP-complete by the discussion above. For these cases, the allowed graphs are either edgeless or complete, and thus Subgraph Isomorphism is trivially polynomial-time solvable. □

Our main contribution in this paper is the following pair of results on Subgraph Isomorphism on graph classes forbidding a fixed graph as a substructure.

Theorem 1.3 *Let $\mathcal{C}$ be the graph class that forbids a fixed connected graph $H \neq P_5$ as either a subgraph, a topological minor, or a minor. Then,* Subgraph Isomorphism *on $\mathcal{C}$ is polynomial-time solvable if H is a subgraph of P_4; otherwise, it is NP-complete.*

Theorem 1.4 *Let $\mathcal{C}$ be the graph class that forbids a fixed (not necessarily connected) graph H as either a subgraph, a topological minor, or a minor. Then,* Subgraph Isomorphism *on $\mathcal{C}$ is*

- *fixed-parameter tractable parameterized by the order of H if H is a linear forest such that at most one component is of order 4 and all other components are of order at most 3;*
- *randomized XP-time solvable parameterized by the order of H if H is a linear forest such that each component is of order at most 4;*
- *NP-complete if either H is not a linear forest, H contains a component with six or more vertices, or H contains three components with five vertices.*

All other cases are randomized polynomial-time reducible to the case where H is P_5 or $2P_5$.

We prove Theorem 1.3 in Sect. 3 and Theorem 1.4 in Sect. 4.

2 Preliminaries and Basic Observations

A graph Q is a *minor* of G if Q can be obtained from G by removing vertices, removing edges, and contracting edges, where contracting an edge $\{u, v\}$ means adding a new vertex $w_{u,v}$, making the neighbors of u and v adjacent to $w_{u,v}$, and removing u and v. A graph Q is a *topological minor* of G if Q can be obtained by removing vertices, removing edges, and contracting edges, where contraction of an edge is allowed if one of the endpoints of the edge is of degree 2. A graph Q is a *subgraph* of G if Q can be obtained by removing vertices and edges. If we cannot remove edges but can do the other modifications as before, then we get the induced variants *induced minor*, *induced topological minor*, and *induced subgraph*.

Recall that a graph is a linear forest if it is the disjoint union of paths. In other words, a graph is a linear forest if and only if it does not contain a cycle nor a vertex of degree at least 3. Observe that in all graph containment relations mentioned above, if we do not forbid any linear forest from a graph class, then the class includes all linear forests. Thus, by Proposition 1.1, we have the following lemma.

Lemma 2.1 *If H is not a linear forest, then* Subgraph Isomorphism *is NP-complete for graphs that do not contain H as a minor, a topological minor, a subgraph, an induced minor, an induced topological minor, or an induced subgraph.*

2.1 Graphs Forbidding a Short Path as a Minor

By the discussion above, we can focus on a graph class forbidding a linear forest as a minor (or equivalently as a topological minor or a subgraph). We here characterize graph classes forbidding a short path as a minor.

Lemma 2.2 *A connected P_3-minor free graph is isomorphic to K_1 or K_2.*

Proof If a connected graph is not complete, then there is a path between nonadjacent vertices. This path contains P_3 as a subgraph. A complete graph with more than two vertices contains P_3 as a subgraph. □

Lemma 2.3 *A connected P_4-minor free graph is isomorphic to K_1, K_3, or $K_{1,s}$ for some $s \geq 1$.*

Proof Let H be a connected P_4-minor free graph. If H is not a tree, it has a cycle C as a subgraph. As H is P_4-minor free, this cycle C has length 3. If H contains a vertex v that is not in C but has a neighbor on C, then the vertices in C together with v induce a subgraph of H that contains P_4 as a subgraph. The connectivity of H implies that $H = C = K_3$.

Now assume that H is a tree with two or more vertices. If H has no universal vertex, then there are two edges $e_1, e_2 \in E(H)$ that do not share any endpoint. The edges e_1 and e_2 with the unique path connecting them form a path of at least four vertices. Thus H has a universal vertex, and hence it is a star. □

3 Forbidding a Connected Graph as a Minor

Here we first show that Subgraph Isomorphism on P_k-minor free graphs is linear-time solvable if $k \le 4$. Note that P_k-minor free graphs include all $P_{k'}$-minor free graphs if $k' \le k$.

The following result can be easily obtained from Lemma 2.3.

Lemma 3.1 Subgraph Isomorphism *on P_4-minor free graphs is linear-time solvable.*

Proof Let G be the host graph and Q be the pattern graph. We assume that $|V(Q)| \le |V(G)|$ since otherwise $Q \not\preceq G$. By Lemma 2.3, each component in both graphs is either an isolated vertex K_1, a triangle K_3, or a star $K_{1,s}$ for some $s \ge 1$.

We first remove isolated vertices. Let Q' and G' be the graphs obtained from Q and G, respectively, by removing all isolated vertices. Since $|V(Q)| \le |V(G)|$, $Q \preceq G$ if and only if $Q' \preceq G$. Also, since isolated vertices in G cannot be used to embed any vertex of Q', $Q' \preceq G$ if and only if $Q' \preceq G'$. In the following, we assume that Q and G do not have isolated vertices.

We next get rid of the triangles. A triangle K_3 in Q has to be matched to a triangle in G. Therefore, if Q contains t triangles, we can remove t triangles from each of G and Q, and obtain an equivalent instance. (If G does not contain t triangles, then we can immediately find that $Q \not\preceq G$.) Since Q contains no triangle anymore, all triangles K_3 in G can be replaced with the same number of $K_{1,2}$'s.

Now we have only stars $K_{1,s}$ with $s \ge 1$ in both graphs. The rest of the problem can be solved by greedily matching a maximum star in Q to a maximum star in G.

The preprocessing phase can be done in linear time. The matching phase can be done in linear time as well since we just need to bucket sort the component sizes in each graph and compare them. $\square$

The following theorem implies that Subgraph Isomorphism on P_k-minor free graphs is NP-complete for every $k \ge 6$.

Theorem 3.2 Subgraph Isomorphism *is NP-complete when the host graph is a forest without paths of length 6 and the pattern is a collection of stars.*

Proof The problem clearly is in NP. To show hardness, we reduce from Exact 3-Cover [15]:

Exact 3-Cover
Input: Collection $\mathcal{C}$ of subsets of a set U such that each $c \in \mathcal{C}$ has size 3.
Question: Is there a subcollection $\mathcal{C}' \subseteq \mathcal{C}$ such that $\bigcup_{C \in \mathcal{C}'} C = U$ and $|\mathcal{C}'| = |U|/3$?

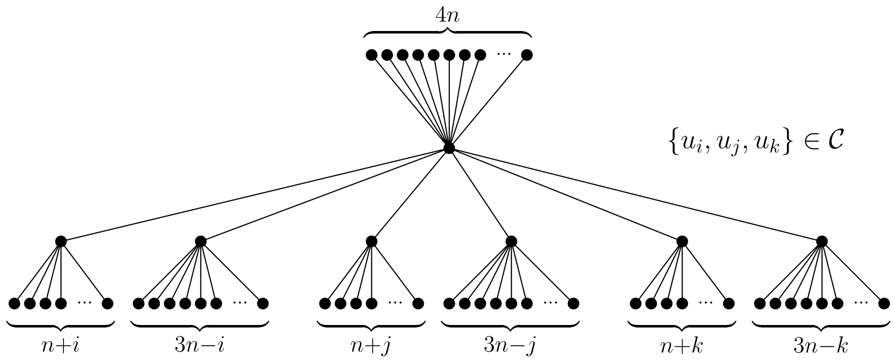

Fig. 1 The tree in G corresponding to $\{u_i, u_j, u_k\} \in \mathcal{C}$

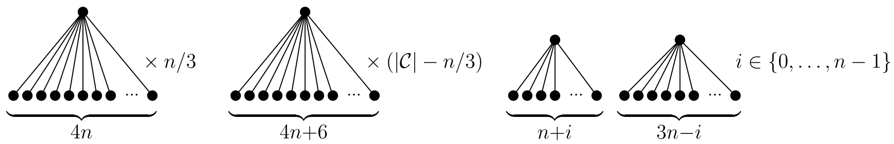

Fig. 2 The pattern graph Q

Suppose we have an instance $(\mathcal{C}, U)$ of Exact 3-Cover given, where $U = \{u_0, \ldots, u_{n-1}\}$. From $(\mathcal{C}, U)$, we construct the host graph G and the pattern Q.

The host G consists of the disjoint union of $|\mathcal{C}|$ trees as follows (see Fig. 1). For each set $C \in \mathcal{C}$, we take a tree in G as follows. Take a star $K_{1,4n+6}$. For each $u_i \in C$, do the following: take one of the leaves of the star, and add $n + i$ pendant vertices to it. Take another leaf of the star, and add $3n - i$ pendant vertices to it. I.e., if $C = \{u_i, u_j, u_k\}$, then the corresponding tree has seven vertices of degree more than 1: one vertex with degree $4n + 6$, which is also adjacent to each of the other six non-leaf vertices; the non-leaf vertices have degree $n + i + 1$, $3n - i + 1$, $n + j + 1$, $3n - j + 1$, $n + k + 1$, and $3n - k + 1$. Call the vertex of degree $4n + 6$ the *central* vertex of the component of C.

The pattern graph Q consists of a number of stars (see Fig. 2):

- We have $n/3$ stars $K_{1,4n}$.
- We have $|\mathcal{C}| - n/3$ stars $K_{1,4n+6}$.
- For each $i \in \{0, \ldots, n - 1\}$, we have stars $K_{1,n+i}$ and $K_{1,3n-i}$. Call these the *element stars*.

From $(\mathcal{C}, U)$, G and Q can be constructed in polynomial time. Now we show that $Q \preceq G$ if and only if $(\mathcal{C}, U)$ is a yes-instance of Exact 3-Cover. We assume that $n > 6$ in the following.

The if direction: Suppose that the Exact 3-Cover instance $(\mathcal{C}, U)$ has a solution $\mathcal{C}' \subseteq \mathcal{C}$.

We map each $K_{1,4n+6}$ of Q into a component M of G corresponding to a set $D \notin \mathcal{C}'$. The center of $K_{1,4n+6}$ is mapped to the central vertex of M and all leaves to its neighbors. The other vertices T are isolated and not used.

Embed each $K_{1,4n}$ of Q into a component L of G corresponding to a set $C \in \mathcal{C}'$, mapping the center of $K_{1,4n}$ to the central vertex of L, and the leaves of $K_{1,4n}$ to leaves neighboring the central vertex of L. After we have done so, we left in this component six stars: if $C = \{u_i, u_j, u_k\}$, then the vertices in L that we did not yet use form stars $K_{1,n+i}$, $K_{1,3n-i}$, $K_{1,n+j}$, $K_{1,3n-j}$, $K_{1,n+k}$, $K_{1,3n-k}$. We thus can embed the element stars corresponding to u_i, u_j, and u_k in these stars, and have embedded the entire pattern in the host graph since $\mathcal{C}'$ is a cover of U.

The only if direction: Suppose that $Q \preceq G$. Note that both Q and G have exactly $|\mathcal{C}|$ vertices of degree at least $4n$. Thus it follows that each vertex of degree at least $4n$ in Q must be mapped to a central vertex of a component in G. We can see that one of the following two cases must hold for the components in the host graph G.

Case 1: A star $K_{1,4n+6}$ *is embedded in the component*. This "uses up" the central vertex and all its neighbors. The only vertices in the component that are not in the image of the star $K_{1,4n+6}$ are leaves with its neighbor being used: these isolated vertices thus cannot be used for embedding any other stars. So all element stars must be embedded in components for which Case 2 holds.

Case 2: A star $K_{1,4n}$ *is embedded in the component*. At this point, note that the total number of vertices of element stars in Q equals $4n^2 + 2n$: each of the n elements has in total $4n$ leaves and two high degree vertices in its element stars. Also, the total number of vertices not used by the stars $K_{1,4n}$ in the Case 2-components equals $4n^2 + 2n$: we have $n/3$ components of Case 2 in G and each has $16n + 7$ vertices of which $4n + 1$ are used for embedding the star $K_{1,4n}$. Thus, each vertex in a Case 2-component M must be used for embedding a vertex. This is only possible if we embed in M the element stars of the elements in the set corresponding to M.

So, let $\mathcal{C}'$ be the sets whose component is of Case 2, i.e., where we embedded a $K_{1,4n}$ in its component. This subcollection $\mathcal{C}'$ is a solution for Exact 3-Cover: for each element u_i, its element stars are embedded in a component that corresponds to a set C that contains u_i, and by the argument above $C \in \mathcal{C}'$. □

By Lemma 2.1, if a connected graph H is not a path, then Subgraph Isomorphism on H-minor free graphs is NP-complete. Assume that H is a path P_k. If $k \geq 6$, then by Theorem 3.2 the problem is NP-complete. If $k \leq 4$, then by Lemma 3.1 the problem can be solved in polynomial time. This completes the proof of Theorem 1.3.

4 Forbidding a Disconnected Graph as a Minor

In this section, we study the more general cases where the forbidden minor H is not necessarily connected. By Lemma 2.1, we can focus on linear forests H. We already know, by Theorem 3.2, if H contains a component with six or more vertices the problem becomes NP-complete. Thus in the following we consider the case where the components of H have five or less vertices.

Using the results in this section, we can prove Theorem 1.4. Corollary 4.2 implies the positive case of Theorem 1.4. Theorems 3.2 and 4.5 together with Lemma 2.1 imply the negative cases. The discussion on the missing cases will be in Sect. 4.4.

4.1 Subgraph Isomorphism on ($P_4 \cup kP_3$)-Minor Free Graphs

We show that Subgraph Isomorphism on $(P_4 \cup kP_3)$-minor free graphs is fixed-parameter tractable when parameterized by k. To this end, we present an algorithm that is parameterized by the vertex integrity, which we think is of independent interest. The *vertex integrity* [2] of a graph is the minimum integer k such that there is a vertex set $S \subseteq V$ such that $|S| \le k$ and the maximum order of the components of $G - S$ is at most $k - |S|$. We call such S a vi(k) *set* of G. Note that the property of having vertex integrity at most k is closed under the subgraph relation.

This subsection is devoted to the proof of the following theorem.

Theorem 4.1 Subgraph Isomorphism *on graphs of vertex integrity at most k is fixed-parameter tractable when parameterized by k.*

By combining Theorem 4.1, Lemma 3.1, and the fact that kP_3-minor free graphs have vertex integrity at most $3k - 1$, we can prove the following.

Corollary 4.2 Subgraph Isomorphism *on $(P_4 \cup kP_3)$-minor free graphs is fixed-parameter tractable when parameterized by k.*

Proof Let G be the host graph and Q be the pattern graph. We first check whether the input graphs are P_4-minor free. This can be done in polynomial time since we just need to check the existence of a P_4 subgraph. If G is P_4-minor free but Q is not, then $Q \not\preceq G$. If both are P_4-minor free, then the problem can be solved in polynomial time by Lemma 3.1. Hence, in the following, we assume that G has a P_4 minor, and thus has a subgraph R isomorphic to P_4. Since G is $(P_4 \cup kP_3)$-minor free, $G - V(R)$ is kP_3-minor free.

Observe that a kP_3-minor free graph has vertex integrity at most $3k - 1$: by repeatedly removing vertices that form a P_3 subgraph at most $k - 1$ times, we can make the graph P_3-minor free, which has the maximum component order at most 2 by Lemma 2.2. Therefore, G itself has vertex integrity at most $3k - 1 + |V(R)| = 3k + 3$.

We now check whether Q has vertex integrity at most $3k + 3$, which can be done in FPT time parameterized by k [10]. If this is not the case, then $Q \not\preceq G$ as the vertex integrity cannot be larger in a subgraph. If Q has vertex integrity at most $3k + 3$, then we can apply Theorem 4.1. $\square$

To prove Theorem 4.1, we start with the following simple fact.

Lemma 4.3 *Let η be a subgraph isomorphism from Q to G. For every* $\mathsf{vi}(k)$ *set T of G, there exists a minimal* $\mathsf{vi}(k)$ *set S of Q such that $\eta(S) \subseteq T$.*

Proof Let $G' = G[\eta(V(Q))]$ and $T' = T \cap \eta(V(Q))$. The set T' is a $\mathsf{vi}(k)$ set of G' since $|T'| \leq |T|$ and $G' - T'$ is a subgraph of $G - T$. Let $S' = \eta^{-1}(T')$. Since η restricted to $V(Q) - S'$ is a subgraph isomorphism from $Q - S'$ to $G' - T'$, the maximum component order of $Q - S'$ cannot be larger than that of $G' - T$, and thus S' is a $\mathsf{vi}(k)$ set of Q. Now every minimal $\mathsf{vi}(k)$ set $S \subseteq S'$ of Q satisfies that $\eta(S) \subseteq T' \subseteq T$. $\square$

Our algorithm assumes that there is a subgraph isomorphism η from Q to G and proceeds as follows:

1. find a $\mathsf{vi}(k)$ set T of G;
2. guess a minimal $\mathsf{vi}(k)$ set S of Q such that $\eta(S) \subseteq T$;
3. guess the bijection between S and $R := \eta(S)$;
4. guess a subset $F \subseteq E(G - R)$ of the edges "unused" by η such that R is a $\mathsf{vi}(k)$ set of $G - F$;
5. solve the problem of deciding the extendability of the guessed parts as the feasibility problem of an integer linear program with a bounded number of variables.

Proof [Theorem 4.1] Let G and Q be graphs of vertex integrity at most k. Our task is to find a subgraph isomorphism η from Q to G in FPT time parameterized by k.

We first find a $\mathsf{vi}(k)$ set T of G and then guess a minimal $\mathsf{vi}(k)$ set S of Q such that $\eta(S) \subseteq T$ for some subgraph isomorphism η from Q to G. By Lemma 4.3, such a set S exists if η exists. Finding T can be done in $O(k^{k+1}n)$ time [10], where $n = |V(G)|$. To guess S, it suffices to list all minimal $\mathsf{vi}(k)$ set S of Q. The algorithm in [10] can be modified for this purpose. To find a $\mathsf{vi}(k)$ set, it traverses a $(k + 1)$-ary search tree of height k that contains all minimal $\mathsf{vi}(k)$ sets. By exhaustively traversing the search tree even after finding a $\mathsf{vi}(k)$ set, we can find all minimal $\mathsf{vi}(k)$ sets in the same running time. (See [5, Exercise 3.3] for the same discussion about minimal vertex covers of size k.)

We then guess the subset R of T such that $\eta(S) = R$. We also guess for each $s \in S$, the image $\eta(s) \in R$. That is, we guess an injection from S to T. The number of such injections is $\binom{|T|}{|S|} \cdot |S|! \leq k!$. If there is an edge $\{u, v\} \in E(Q[S])$ such that $\{\eta(u), \eta(v)\} \notin E(G[R])$, then we reject this guess. Otherwise, we try to further extend η.

Observe that R is not necessarily a $\mathsf{vi}(k)$ set of G. In the following, we guess "unnecessary" edges in $G - R$. That is, we guess a subset F of the edges that are

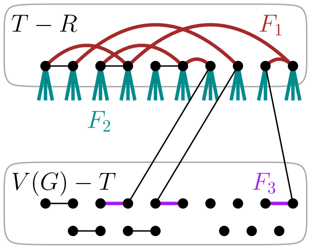

Fig. 3 F_1 is a set of edges in $G[T-R]$, F_2 a set of edges between $V(G)-T$ and $T-R$, and F_3 a set of edges in $G-T$. The example shows the case where $|R| = k-2$ and thus the components of $G-R-(F_1 \cup F_2 \cup F_3)$ have to have order at most 2

not used by η as images of any edges in Q. Furthermore, we select F so that R is a $\mathsf{vi}(k)$ set of $G-F$. Such F exists because η embeds $Q-S$ (and no other things) into $G-R$. Note that we cannot make a component of $G-R$ small by removing edges incident to at least one vertex of R, and thus we find F without such edges.

Guessing F: We now show that the number of candidates of F that we need to consider is bounded by some function in k. We partition F into three sets $F_1 = F \cap E(G[T-R])$, $F_2 = F \cap E(V(G)-T, T-R)$, and $F_3 = F \cap E(G-T)$ and then count the numbers of candidates separately. See Fig. 3.

Guessing F_1: For F_1, we just use all $2^{|E(G[T-R])|} < 2^{k^2}$ subsets of $E(G[T-R])$ as candidates. If R is not a $\mathsf{vi}(k)$ set of $G[T]-F_1$, we reject this F_1.
Guessing F_2: Since we are finding F such that R is a $\mathsf{vi}(k)$ set of $G-F$, each vertex in $T-R$ has less than k edges to $V(G)-T$ in $G-F$. Thus fewer than k^2 components of $V(G)-T$ have edges to $T-R$ in $G-F$. We guess such components $\mathcal{C}$.
Observe that each component in $V(G)-T$ is of order at most k and that each vertex of $V(G)-T$ can be partitioned into at most 2^k types with respect to the adjacency to T. This implies that the components of $V(G)-T$ can be classified into at most 4^{k^2} types (2^{k^2} for the isomorphism type and $(2^k)^k$ for the adjacency to T) in such a way that if two components C_1 and C_2 of $G-T$ are of the same type, then there is an automorphism of G that fixes T and maps C_1 to C_2. Given this classification of the components in $V(G)-T$, we only need to guess how many components of each type are included in $\mathcal{C}$. For this guess, we have at most $\binom{4^{k^2}+k^2-1}{k^2} < 4^{k^4+k^2}$ options.
For each guess $\mathcal{C}$, we guess the edges connecting the components in $\mathcal{C}$ to $T-R$ in $G-F$. Since $|\mathcal{C}| < k^2$ and $|C| \le k$ for each $C \in \mathcal{C}$, there are at most $k^3 \cdot |T-R| \le k^4$ candidate edges. We just try all $O(2^{k^4})$ subsets F_2' of such edges, and set $F_2 = E(V(G)-T, T-R) - F_2'$. In total, we have $O(2^{2k^4+k^2} \cdot 2^{k^4})$ options for F_2.
Guessing F_3: Recall that $G-T$ does not contain any component of order more than k. Hence, if $G-R-(F_1 \cup F_2)$ has a component of order more than k, then it consists of some vertices in $T-R$ and some components in $\mathcal{C}$. Thus, we only need to pick some edges of the components in $\mathcal{C}$ for F_3 to make R a $\mathsf{vi}(k)$ set of

$G-F$. We use all 2^{k^4} subsets of the edges of the components in $\mathcal{C}$ as a candidate of F_3.
In total, $F = F_1 \cup F_2 \cup F_3$ has at most $2^{k^2} \cdot 4^{k^4+k^2} \cdot 2^{k^4} \cdot 2^{k^4}$ candidates, and each candidate can be found in FPT time. We reject this guess F if R is not a $\mathsf{vi}(k)$ set of $G-F$. In the following, we assume that F is guessed correctly and denote $G-F$ by G'.

Extending η: Recall that we already know how η maps S to R and that each component in $Q-S$ and $G'-R$ is of order at most k. We now extend η by determining how η maps $Q-S$ to $G'-R$. By renaming vertices, we can assume that $S = \{s_1, \dots, s_q\}$, $R = \{r_1, \dots, r_q\}$, and $\eta(s_i) = r_i$ for $1 \le i \le q$.

We say that a vertex u in $Q-S$ *matches* a vertex v in $G'-R$ if $\{i \mid s_i \in N_Q(u) \cap S\} \subseteq \{i \mid r_i \in N_{G'}(v) \cap R\}$. A set of components $\{C_1, \dots, C_h\}$ of $Q-S$ *fits* a component D of $G'-R$ if there is an isomorphism ϕ from the disjoint union of $C_1, \dots, C_h$ to D such that for all $u \in \bigcup_i V(C_i)$ and $v \in V(D)$, $\phi(u) = v$ holds only if u matches v. Note that if $h > k$, then $\{C_1, \dots, C_h\}$ can fit no component of $G'-R$.

As we did before for guessing F_2, we classify the components of $Q-S$ and $G'-R$ into at most 4^{k^2} types. Two components C_1 and C_2 of $Q-S$ (or of $G'-R$) are of the same type if and only if there is an isomorphism ϕ from C_1 to C_2 such that $\phi(v_1) = v_2$ implies that $N_Q(v_1) \cap S = N_Q(v_2) \cap S$ (or $N_{G'}(v_1) \cap R = N_{G'}(v_2) \cap R$, respectively). We denote by $t(C)$ the type of a component C and by $t(\{C_1, \dots, C_h\})$ the multi-set $\{t(C_1), \dots, t(C_h)\}$. Observe that $\{C_1, \dots, C_h\}$ fits D if and only if all sets $\{C'_1, \dots, C'_h\}$ with $t(\{C'_1, \dots, C'_h\}) = t(\{C_1, \dots, C_h\})$ fits D' with $t(D') = t(D)$.

Observe that the guessed part $\eta|_S$ can be extended to a subgraph isomorphism η from Q to G' if and only if there is a partition of the components of $Q-S$ such that each part $\{C_1, \dots, C_h\}$ in the partition can be injectively mapped to a component D of $G'-R$ where $\{C_1, \dots, C_h\}$ fits D. To check the existence of such a partition, we only need to find for each pair of a multi-set $\mathcal{T}$ of types of a set of components in $Q-S$ and a type τ of a component in $G'-R$, how many sets of components of type $\mathcal{T}$ the map η embeds to components of type τ. We use the following ILP formulation to solve this problem.

Let n_τ and n'_τ be the numbers of type-τ components in $Q-S$ and $G'-R$, respectively. These numbers can be computed in FPT time parameterized by k.

For each type τ and for each multi-set $\mathcal{T}$ of types such that $\mathcal{T}$ fits τ, we use a variable $x_{\mathcal{T},\tau}$ to represent the number of type-$\mathcal{T}$ multi-sets of components in $Q-S$ that are mapped to type-τ components in $G'-R$. For each type τ of components in $G'-R$, we can embed at most n_τ sets of components in $Q-S$. This constraint is expressed as follows:

$$n_\tau \ge \sum_{\mathcal{T} : \mathcal{T} \text{ fits } \tau} x_{\mathcal{T},\tau} \quad \text{for each type } \tau. \tag{1}$$

For each type σ of components in $Q-S$, we need to embed all n_σ components of type σ into some components of $G-R'$. We can express this constraint as follows:

$$n_\sigma = \sum_{\mathcal{T},\tau\,:\,\sigma\in\mathcal{T}\text{ and }\mathcal{T}\text{ fits }\tau} \mu_{\mathcal{T},\sigma} \cdot x_{\mathcal{T},\tau} \quad \text{for each type } \sigma, \tag{2}$$

where $\mu_{\mathcal{T},\sigma}$ is the multiplicity of σ in $\mathcal{T}$. This completes the ILP formulation of the problem. We do not have any objective function and just ask for the feasibility. The construction can be done in FPT time parameterized by k.

Observe that there are at most $\binom{4^{k^2}+k-1}{k} < 4^{k^3+k}$ multi-sets $\mathcal{T}$ of types of components. Thus the ILP above has at most $4^{k^2} \cdot 4^{k^3+k}$ variables (the first factor for τ and the second for $\mathcal{T}$) and at most $4^{k^2} \cdot 4^{k^3+k} + 4^{k^2} \cdot 4^{k^2} \cdot 4^{k^3+k}$ constraints (the first term for (1) and the second for (2)) of length $O(4^{k^2} \cdot 4^{k^3+k})$. The coefficients are upper bounded by $|V(G')|$. It is known that the feasibility check of such an ILP can be done in FPT time parameterized by k [13, 19, 24]. Thus, the problem can be solved in FPT time when parameterized by k. (The running time is doubly exponential in k^3.) □

4.2 Subgraph Isomorphism on kP_4-Minor Free Graphs

We show that Subgraph Isomorphism on kP_4-minor free graphs is randomized XP-time solvable parameterized by k. Our randomized algorithm is a Monte Carlo algorithm with false negatives. That is, it always rejects a no-instance, but may reject a yes-instance with probability at most 1/2.

For a graph $G = (V, E)$, a set $S \subseteq V$ is a *P_4-hitting set* of G if $G - S$ does not contain P_4 as a minor (or equivalently as a subgraph). The *P_4-hitting number* of G is the minimum size of a P_4-hitting set of G. To show the main result of this section, we prove Theorem 4.4 below, which immediately gives the result claimed above on kP_4-minor free graphs as their P_4-hitting number is at most $4(k - 1)$.

Our algorithm will find a subgraph isomorphism η from Q to G as follows:

1. Find a P_4-hitting set T of G such that $|T| \le k$, and guess the "used" part R of T.
2. Guess $S = \eta^{-1}(R)$ and the mapping from S to R.
3. Color the vertices of $G - T$ and $Q - S$ according to the connections to R and S, respectively.
4. Guess how many vertices of each color c in G will remain unused after embedding the non-singleton components (triangles and stars) in $Q - S$ to $G - T$. Check whether the singleton components in Q can be embedded to the guessed vertices.
5. Construct an auxiliary bipartite multi-graph B from the components of G and the non-singleton components in Q (with some dummy vertices).
6. Find a perfect matching of B with a specific weight. Using such a matching, extend the guessed parts of η to a subgraph isomorphism from Q to G.

Theorem 4.4 Subgraph Isomorphism *on graphs with P_4-hitting number at most k admits a randomized $n^{O(2^k)}$-time algorithm with false negatives.*

Proof Let G and Q be graphs of P_4-hitting number at most k. We will find a subgraph isomorphism η from Q to G in randomized XP time parameterized by k. In the following, we denote by n and m the total numbers of vertices and edges, respectively, in G and Q.

Mapping P_4-hitting sets: We first find a P_4-hitting set T of G such that $|T| \le k$. This can be done in time $O(4^k(n+m))$ by branching on P_4-subgraphs and checking P_4-subgraph freeness. We guess $R = \eta(V(Q)) \cap T$, $S = \eta^{-1}(R)$, and $\eta(s) \in R$ for each $s \in S$. We reject the guess if $\{u, v\} \in E(Q[S])$ and $\{\eta(u), \eta(v)\} \notin E(G[R])$ for some $u, v \in S$. Also, if S is not a P_4-hitting set of Q, then we reject this guess as we cannot map $Q - S$ to $G - T$. We have $O(2^k)$ options for R, $O(n^k)$ options for S, and $O(k!)$ options for the mapping from S to R. By Lemma 2.3, each component of $G - T$ and $Q - S$ is either a singleton K_1, a triangle K_3, or a star $K_{1,\ell}$ for some $\ell \ge 1$.

Coloring vertices: We rename the vertices in R and S to have $R = \{r_1, \dots, r_q\}$, $S = \{s_1, \dots, s_q\}$, and $\eta(s_i) = r_i$ for $1 \le i \le q$. We set the *color* of a vertex $v \in G - T$, denoted $\mathsf{col}(v)$, to be the set $\{i \mid r_i \in N_G(v)\}$. Similarly, we set the color $\mathsf{col}(u)$ of a vertex $u \in Q - S$ to be $\{i \mid s_i \in N_Q(u)\}$. Observe that $u \in Q - S$ can be embedded to $v \in G - T$ only if $\mathsf{col}(u) \subseteq \mathsf{col}(v)$ (assuming that the guesses so far are correct).

For a set of vertices X of $G - T$ or $Q - S$ and a color $C \subseteq \{1, \dots, q\}$, we set $h_X(C)$ to be the number of vertices $v \in X$ such that $\mathsf{col}(v) = C$. We call h_X the *color histogram* of X.

In the later steps, it is convenient to identify color histograms with 2^q-digit n-ary numbers. Ordering the subsets of $\{1, \dots, q\}$ in an arbitrary way, the ith digit can represent the number of vertices having the ith subset as their color. Then for disjoint sets X and Y, it holds that $h_X + h_Y = h_{X \cup Y}$.

Guessing the color histogram of unused vertices: We guess the color histogram h_A of the set A of vertices that remain unused after embedding the non-singleton components (triangles and stars) in $Q - S$ to $G - T$. (Note that we do not guess A directly.) The number of possible options for h_A is $O(n^{2^k})$. At this point, we do not know whether there is an embedding of the non-singleton components consistent with h_A. For now, we assume the existence of such an embedding, and first test whether the singleton components of $Q - S$ can be embedded to the unused vertices of $G - T$ guessed as h_A.

We need to embed each singleton component u in $Q - S$ to a vertex v such that $\mathsf{col}(u) \subseteq \mathsf{col}(v)$. So the problem here can be reduced to the problem of finding a matching saturating U in the bipartite graph such that

- the vertex set is $U \cup U'$;
- U is the set of singleton components of $Q - S$;
- U' is a set of vertices that contains exactly $h_A(C)$ vertices of each color C;
- $u \in U$ and $v \in U'$ are adjacent if and only if $\mathsf{col}(u) \subseteq \mathsf{col}(v)$.

Since $|U \cup U'| \le n$, we can check this in polynomial time.

Embedding non-singleton components: We finally test the existence of an embedding of the stars and triangles in $Q - S$ to $G - T$ consistent with h_A. We reduce this task to the problem of deciding the existence of a perfect matching with a specific weight in the bipartite multi-graph $B = (X, Y \cup Z; E)$ defined as follows:

1. X corresponds to the components of $G - T$.
2. Y corresponds to the non-singleton components in $Q - S$.
3. Z is the set of dummy vertices such that $|Z| = |X| - |Y|$. (Observe that $|X| - |Y|$ has to be nonnegative if S is guessed correctly.)
4. For $x \in X$ and $y \in Y$, add one edge of weight h_D if there is a way to embed the component C_y corresponding to y to the component C_x corresponding to x such that the set of remaining vertices is D. (There could be multiple edges with different weights between x and y.)
5. For $x \in X$ and $z \in Z$, add an edge of weight h_D, where D is the set of vertices of the component C_x corresponding to x.

The graph B can be constructed in time $n^{O(2^k)}$. To see this, the 4th step is the only nontrivial one. For that step, we have n^{2^k} candidates for h_D. Each candidate can be checked in time polynomial in $n + 2^k$ since each component involved is K_1, K_3, or $K_{1,s}$.

From the construction, there exists an embedding of the non-singleton components in $Q - S$ to $G - T$ consistent with h_A if and only if B has a perfect matching of weight exactly h_A. Including an edge between $x \in X$ and $y \in Y$ of weight h_D into the perfect matching means mapping C_y to C_x in such a way that $V(C_x) \setminus \eta(V(C_y))$ has the color histogram h_D. Including an edge between $x \in X$ and $z \in Y$ means that C_x is not used to embed any non-singleton component of $Q - S$.

Now it suffices to find a perfect matching of B with weight exactly h_A. It is known that, given a multi-graph with the maximum edge weight bounded by W and a target total weight T, there is a Monte Carlo algorithm with false negatives that finds a perfect matching of weight exactly T (if any) in time polynomial in $|V(B)| + |E(B)| + W$ [31] (see also [18, 26, 27]). Since W is $n^{O(2^k)}$, the theorem holds. □

We may need completely new ideas to find a deterministic counterpart of Theorem 4.4. Although the randomized algorithm we used in the proof of Theorem 4.4 has been known for decades [31], it is still unknown whether there exists a deterministic algorithm for finding a perfect matching of given exact weight with running time polynomial in the number of vertices and the sum of edge weights.

To consider the tightness of Theorem 4.4 from a different direction, it would be interesting to ask whether Subgraph Isomorphism parameterized by P_4-hitting number is W[1]-hard. If this is the case, $f(k) \cdot n^{O(1)}$-time algorithms for some computable f cannot be achieved under some complexity assumptions [5].

4.3 Subgraph Isomorphism on $3P_5$-Minor Free Graphs

A *double star* $D_{a,b}$ is the graph obtained from two stars $K_{1,a}$ and $K_{1,b}$ by connecting the centers of the stars with an edge.

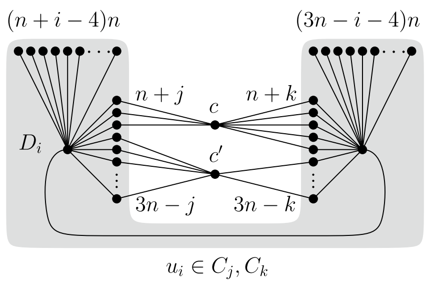

Fig. 4 The gadgets in the host graph G. Note that the special vertices c and c' are shared by all gadgets although their copies are depicted for each gadget for the readability. We omit the pendant vertices attached to c and c'

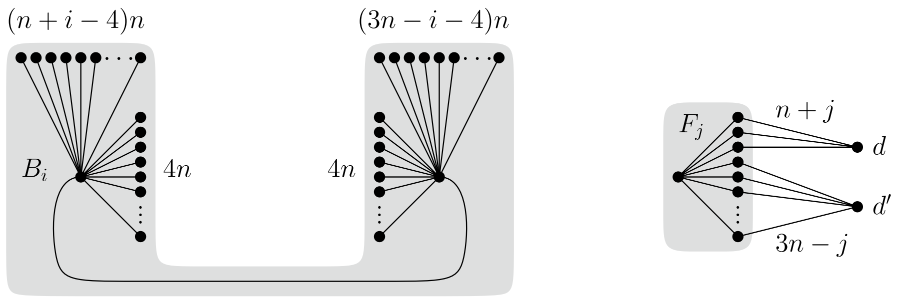

Fig. 5 The gadgets in Q. We omit the pendant vertices attached to d and d'

Theorem 4.5 Subgraph Isomorphism *on* $3P_5$*-minor free graphs is NP-complete.*

Proof The problem is clearly in NP. We show the NP-hardness by a reduction from 3-Sat with the restriction that each clause contains two or three literals and that each variable occurs exactly twice as a positive literal and exactly once as a negative literal. We call this variant 3-Sat(2, 1). It is known that 3-Sat(2, 1) is NP-complete [12]. Let $(U, \mathcal{C})$ be an instance of 3-Sat(2, 1) with the variables $U = \{u_0, \dots, u_{n-1}\}$ and the clauses $\mathcal{C} = \{C_0, \dots, C_{m-1}\}$.

We first construct the host graph G (see Fig. 4). It consists of $2n$ double stars, two special vertices c and c', and many pendant vertices attached to c and c'. For each variable $u_i \in U$, we take two isomorphic double stars $D_{(n+i)n,(3n-i)n}$ and call them D_i and $\overline{D}_i$. The double stars D_i and $\overline{D}_i$ correspond to the positive literal u_i and the negative literal $\overline{u}_i$, respectively. We add for each $u_i \in U$ some edges between leaves of the double stars D_i and $\overline{D}_j$ and the special vertices c and c' as follows. Let $u_i \in C_j, C_k$ and $\overline{u}_i \in C_\ell$. We arbitrarily and bijectively assign the two stars $K_{1,(n+i)n}$ and $K_{1,(3n-i)n}$ in D_i to C_j and C_ℓ. From the star in D_i assigned to C_h ($h \in \{j, \ell\}$), we pick $n+h$ leaves and make them adjacent to c, and then pick $3n-h$ leaves from the remaining and make them adjacent to c. Similarly, in $\overline{D}_i$, we choose one side of the double star, make $n+\ell$ of the leaves there adjacent to c, and make $3n-\ell$ of the remaining adjacent to c'. So far, we took $N := 4n^3 + 2n + 2$ vertices into G. By attaching N

and $2N$ pendant vertices to c and c', respectively, we complete the construction of G. Observe that if we remove c and c' from G, then it becomes a collection of double stars and isolated vertices, which is P_5-minor free. Thus the host graph G cannot have $3P_5$ as a minor.

The pattern graph Q consists of n double stars, m stars $K_{1,4n}$, and two additional vertices d and d' (see Fig. 5). For each variable $u_i \in U$, we take a double star $D_{(n+i)n,(3n-i)n}$ and call it B_i. For each clause $C_j \in \mathcal{C}$, we take a star $K_{1,4n}$ and call it F_j. For each star F_j, we arbitrarily select $n+j$ leaves and make them adjacent to d, and then make the remaining $3n-j$ leaves adjacent to d'. Finally, we attach N and $2N$ pendant vertices to d and d', respectively, Observe that if we remove d and d', then the the pattern graph becomes a collection of stars, double stars, and isolated vertices. Thus H is $3P_5$-minor free.

In the following, we show that $Q \preceq G$ if and only if $(U, \mathcal{C})$ is a yes-instance of 3-Sat(2, 1).

The if direction: Suppose that $(U, \mathcal{C})$ has a satisfying assignment $f : U \to \{\texttt{true}, \texttt{false}\}$. We construct a subgraph isomorphism $\eta : V(Q) \to V(G)$ as follows. We first set $\eta(d) = c$ and $\eta(d') = c'$. We map the pendants attached to them appropriately. For each $u_i \in U$, we map B_i to $\overline{D_i}$ if $f(u_i) = \texttt{true}$, and to D_i if $f(u_i) = \texttt{false}$.

Now observe that what we left unused in G for each u_i is the double star that corresponds to the clauses satisfied by the literal of u_i that is true under f. Furthermore, since f is a satisfying assignment of $(U, \mathcal{C})$, each clause $C_j \in \mathcal{C}$ has at least one star assigned to C_j included in an unused double star in G. Also, such a correspondence is injective by the construction. Therefore, we can map each F_j in Q into a corresponding star included in an unused double star in G by mapping the center to the center, the neighbors of d to the neighbors of c, and the neighbors of d' to the neighbors of c'.

The only if direction: Suppose that $Q \preceq G$ and thus there is a subgraph isomorphism $\eta : V(Q) \to V(G)$ from Q to G. Observe that $\eta(d) = c$ and $\eta(d') = c'$ because of their high degrees. Recall that each double star in Q and G has exactly $4n^2 + 2$ vertices and that B_i is isomorphic only to D_i and $\overline{D_i}$. Hence, B_i has to be mapped to D_i or $\overline{D_i}$ for every i.

Now we know that after the vertices $\{d, d'\} \cup \bigcup_i V(B_i)$ are mapped, the unused vertices in G induce a subgraph that contains either D_i or $\overline{D_i}$ for each i. From this subgraph, we construct a truth assignment $f : U \to \{\texttt{true}, \texttt{false}\}$: if D_i is left unused we set $f(u_i) = \texttt{true}$; otherwise we set $f(u_i) = \texttt{false}$.

Assume that D_i is left unused by $\{d, d'\} \cup \bigcup_i V(B_i)$ and $\eta(V(F_j)) \subseteq V(D_i)$ holds for some i. (The other case where D_i is replaced with $\overline{D_i}$ is the same.) Since each leaf in F_j is adjacent to either d or d', each image of them has to be adjacent to either c or c'. Thus F_j is mapped to one of the stars in D_i. Because of the connections to d and d' in Q and to c and c' in G, this is possible only if the star induced by $\eta(V(F_j))$ corresponds to the clause C_j. Thus D_i contains a star corresponding to C_j, and C_j includes the positive literal of u_i. Since D_i is left unused, we have $f(u_i) = \texttt{true}$, and thus C_j is satisfied by f. □

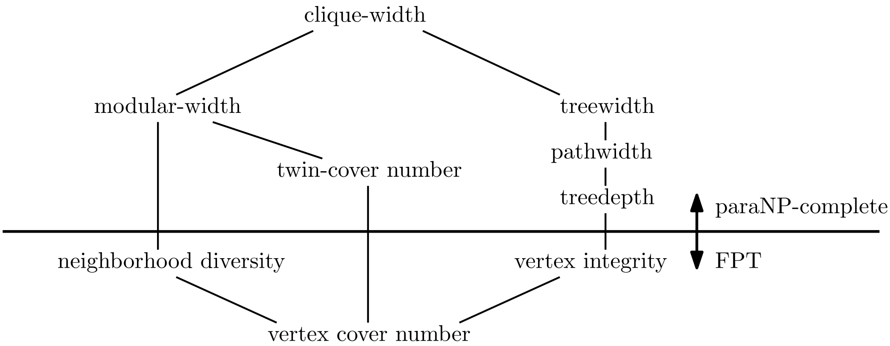

Fig. 6 Graph parameters and Subgraph Isomorphism. For each connection of parameters, there is a function in the parameter above that lower bounds the one below

4.4 Missing Cases

To complete the proof of Theorem 1.4, here we discuss the missing cases of Subgraph Isomorphism on H-minor free graphs. In the missing cases, H contains, as a minor, P_5 or $2P_5$ but no $3P_5$ nor P_6. In other words, there is $p \in \{1, 2\}$ such that H contains a pP_5-minor but no $(p+1)P_5$-minor nor P_6-minor.

We reduce this case to the case where the forbidden minor is pP_5 in randomized polynomial time. Since H is a linear forest, H is a minor of $pP_5 \cup kP_4$ for some constant $k < |V(H)|$. An H-minor free graph is $(pP_5 \cup kP_4)$-minor free, and thus it is either pP_5-minor free or $(k+5p)P_4$-minor free. (The idea here is basically the same with the one in the proof of Corollary 4.2.) If the host graph G is pP_5-minor free (or $(k+5p)P_4$-minor free) but the pattern graph Q is not, then we output a trivial no-instance (e.g., $G = K_1$ and $Q = K_2$). If both G and Q are pP_5-minor free, we just output the original input G and Q as the reduced pP_5 -minor free instance. If both G and Q are $(k+5p)P_4$-minor free, then by Theorem 4.4, we can solve the problem in randomized polynomial-time. Depending on whether $Q \preceq G$ or not, we output a trivial yes-instance (e.g., $G = Q = K_1$) or a trivial no-instance.

5 Structural Parameterizations of Subgraph Isomorphism

We conclude the paper with some remarks on structural parameterizations of Subgraph Isomorphism. Our results imply a few things in this direction. See Fig. 6. The proof of Theorem 3.2 implies that Subgraph Isomorphism is NP-complete even for graphs of tree-depth [32] at most 3. This result is tight by Lemma 3.1 since graphs of tree-depth at most 2 do not contain P_4 as a subgraph. Proposition 1.1 implies it is NP-complete even for graphs of constant twin-cover number [14] because cluster graphs have twin-cover number 0. For the parameterization by neighborhood diversity [23], we can use techniques similar to the ones we used for Theorem 4.1.

Two vertices u and v of a graph $G = (V, E)$ are *twins* if $N(u) \setminus \{v\} = N(v) \setminus \{u\}$. The *neighborhood diversity* of $G = (V, E)$ is the minimum integer k such that V can be partitioned into k sets $T_1, \ldots, T_k$ of pairwise twin vertices. Such a minimum partition can be found in linear time using fast modular decomposition algorithms [30, 34]. Observe that each part T_i in the partition is either complete or independent. Also, for parts T_i and T_j, there are either no edges or all possible edges. We say that T_i and T_j are *adjacent* if there are all possible edges, and *nonadjacent* otherwise.

Theorem 5.1 Subgraph Isomorphism *on graphs of neighborhood diversity at most k is fixed-parameter tractable parameterized by k.*

Proof Let G be the host graph and Q be the pattern graph, both with neighborhood diversity at most k. Let $T_1, \ldots, T_t$ be a partition of the vertices of G into pairwise twin vertices with $t \le k$, and similarly let $R_1, \ldots, R_r$ be a partition of the vertices of Q into pairwise twin vertices with $r \le k$.

We find a subgraph isomorphism η from Q to G by reducing the problem to at most 3^{k^2} instances of ILP as follows. By a variable $x_{i,j}$ we represent the number of the vertices that η maps from R_i to T_j. For each variable $x_{i,j}$, we guess whether $x_{i,j} = 0$, $x_{i,j} = 1$, or $x_{i,j} \ge 2$. Since we have at most k^2 variables $x_{i,j}$, this gives us at most 3^{k^2} options.

We reject this guess if at least one of the following holds:

- $x_{i,j} \ne 0$ and $x_{i',j'} \ne 0$ for adjacent R_i and $R_{i'}$ and nonadjacent T_j and $T_{j'}$;
- $x_{i,j} \ne 0$ and $x_{i',j} \ne 0$ for adjacent R_i and $R_{i'}$ and an independent T_j;
- $x_{i,j} \ge 2$ for a complete R_i and an independent T_j.

For guesses satisfying all the conditions above, we construct an ILP instance as follows. For each variable $x_{i,j}$, we add the guessed constraint $x_{i,j} = 0$, $x_{i,j} = 1$, or $x_{i,j} \ge 2$. For each $i \in \{1, \ldots, r\}$, we add the constraint $\sum_{1 \le j \le t} x_{i,j} = |R_i|$. For each $j \in \{1, \ldots, t\}$, we add the constraint $\sum_{1 \le i \le r} x_{i,j} \le |T_j|$. We can see that this ILP is feasible if and only if there is a subgraph isomorphism consistent with the guess. As we saw in the proof of Theorem 4.1 for vertex integrity, this feasibility check can be done in FPT time parameterized by k. (The dependency on k in the running time is $k^{O(k^2)}$.) □

Funding Partially supported by NETWORKS (the Networks project, funded by the Netherlands Organization for Scientific Research NWO), the ELC project (the project Exploring the Limits of Computation, funded by MEXT), JSPS/MEXT KAKENHI Grant Numbers JP24106004, JP18K11168, JP18K11169, JP18H04091, JP18H06469, JP15K00009, JST CREST Grant Number JPMJCR1402, and Kayamori Foundation of Informational Science Advancement. The authors thank Momoko Hayamizu, Kenji Kashiwabara, Hirotaka Ono, Ryuhei Uehara, and Koichi Yamazaki for helpful discussions. The authors are grateful to the anonymous reviewer of an earlier version of this paper who pointed out a gap in a proof. A preliminary version appeared in the proceedings of the 11th International Conference on Algorithms and Complexity (CIAC 2019), Lecture Notes in Computer Science 11485 (2019) 87–98.

References

1. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM **42**(4), 844–856 (1995). https://doi.org/10.1145/210332.210337
2. Barefoot, C.A., Entringer, R.C., Swart, H.C.: Vulnerability in graphs—a comparative survey. J. Combin. Math. Combin. Comput. **1**, 13–22 (1987)
3. Bodlaender, H.L, Nederlof, J., van der Zanden, T.C.: Subexponential time algorithms for embedding H-minor free graphs. In: ICALP 2016, vol. 55. LIPIcs, pp. 9:1–9:14 (2016). https://doi.org/10.4230/LIPIcs.ICALP.2016.9
4. Bonsma, P.S.: Surface split decompositions and subgraph isomorphism in graphs on surfaces. In: STACS 2012, vol. 14. LIPIcs, pp. 531–542 (2012). https://doi.org/10.4230/LIPIcs.STACS.2012.531
5. Cygan, M., Fomin, F.V., Golovnev, A., Kulikov, A.S., Mihajlin, I., Pachocki, J., Socała, A.: Tight lower bounds on graph embedding problems. J. ACM **64**(3), 18:1–18:22 (2017). https://doi.org/10.1145/3051094
6. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer, New York (2015)
7. Damaschke, P.: Induced subgraph isomorphism for cographs in NP-complete. In: WG 1990, vol. 484. LNCS, pp. 72–78 (1990). https://doi.org/10.1007/3-540-53832-1_32
8. Dorn, F.: Planar subgraph isomorphism revisited. In: STACS 2010, vol. 5. LIPIcs, pp. 263–274 (2010). https://doi.org/10.4230/LIPIcs.STACS.2010.2460
9. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness II: on completeness for W[1]. Theor. Comput. Sci. **141**(1&2), 109–131 (1995). https://doi.org/10.1016/0304-3975(94)00097-3
10. Drange, P.G., Dregi, M.S., van't Hof, P.: On the computational complexity of vertex integrity and component order connectivity. Algorithmica **76**(4), 1181–1202 (2016). https://doi.org/10.1007/s00453-016-0127-x
11. Eppstein, D.: Subgraph isomorphism in planar graphs and related problems. J. Gr. Algorithms Appl. **3**(3), 1–27 (1999). https://doi.org/10.7155/jgaa.00014
12. Fellows, M.R., Kratochvíl, J., Middendorf, M., Pfeiffer, F.: The complexity of induced minors and related problems. Algorithmica **13**(3), 266–282 (1995). https://doi.org/10.1007/BF01190507
13. Frank, A., Tardos, É.: An application of simultaneous diophantine approximation in combinatorial optimization. Combinatorica **7**(1), 49–65 (1987). https://doi.org/10.1007/BF02579200
14. Ganian, R.: Improving vertex cover as a graph parameter. Discrete Math. Theor. Comput. Sci. **17**(2), 77–100 (2015)
15. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, New York (1979)
16. Gupta, A., Nishimura, N.: The complexity of subgraph isomorphism for classes of partial k-trees. Theor. Comput. Sci. **164**(1&2), 287–298 (1996). https://doi.org/10.1016/0304-3975(96)00046-1
17. Hajiaghayi, M.T., Nishimura, N.: Subgraph isomorphism, log-bounded fragmentation, and graphs of (locally) bounded treewidth. J. Comput. Syst. Sci. **73**(5), 755–768 (2007). https://doi.org/10.1016/j.jcss.2007.01.003
18. Jansen, B.M.P., Marx, D.: Characterizing the easy-to-find subgraphs from the viewpoint of polynomial-time algorithms, kernels, and turing kernels. SODA **2015**, 616–629 (2015). https://doi.org/10.1137/1.9781611973730.42
19. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**(3), 415–440 (1987). https://doi.org/10.1287/moor.12.3.415
20. Kijima, S., Otachi, Y., Saitoh, T., Uno, T.: Subgraph isomorphism in graph classes. Discrete Math. **312**(21), 3164–3173 (2012). https://doi.org/10.1016/j.disc.2012.07.010
21. Kiyomi, M., Otachi, Y.: Finding a chain graph in a bipartite permutation graph. Inf. Process. Lett. **116**(9), 569–573 (2016). https://doi.org/10.1016/j.ipl.2016.04.006
22. Konagaya, M., Otachi, Y., Uehara, R.: Polynomial-time algorithms for subgraph isomorphism in small graph classes of perfect graphs. Discrete Appl. Math. **199**, 37–45 (2016). https://doi.org/10.1016/j.dam.2015.01.040
23. Lampis, M.: Algorithmic meta-theorems for restrictions of treewidth. Algorithmica **64**(1), 19–37 (2012). https://doi.org/10.1007/s00453-011-9554-x
24. Lenstra Jr., H.W.: Integer programming with a fixed number of variables. Math. Oper. Res. **8**(4), 538–548 (1983). https://doi.org/10.1287/moor.8.4.538

25. Lingas, A.: Subgraph isomorphism for biconnected outerplanar graphs in cubic time. Theor. Comput. Sci. **63**(3), 295–302 (1989). https://doi.org/10.1016/0304-3975(89)90011-X
26. Marx, D.: List edge multicoloring in graphs with few cycles. Inf. Process. Lett. **89**(2), 85–90 (2004). https://doi.org/10.1016/j.ipl.2003.09.016
27. Marx, D., Pilipczuk, M.: Everything you always wanted to know about the parameterized complexity of subgraph isomorphism (but were afraid to ask). In: STACS 2014, vol. 25. LIPIcs, pp. 542–553 (2014). https://doi.org/10.4230/LIPIcs.STACS.2014.542
28. Matoušek, J., Thomas, R.: On the complexity of finding iso- and other morphisms for partial *k*-trees. Discrete Math. **108**(1–3), 343–364 (1992). https://doi.org/10.1016/0012-365X(92)90687-B
29. Matula, D.W.: Subtree isomorphism in $O(n^{5/2})$. In: Alspach, B., Hell, P., Miller, D.J. (eds.) Algorithmic Aspects of Combinatorics. Annals of Discrete Mathematics, vol. 2, pp. 91–106. Elsevier, Amsterdam (1978)
30. McConnell, R.M., Spinrad, J.P.: Modular decomposition and transitive orientation. Discrete Math. **201**(1–3), 189–241 (1999). https://doi.org/10.1016/S0012-365X(98)00319-7
31. Mulmuley, K., Vazirani, U.V., Vazirani, V.V.: Matching is as easy as matrix inversion. Combinatorica **7**(1), 105–113 (1987). https://doi.org/10.1007/BF02579206
32. Nesetril, J., de Mendez, P.O.: Sparsity-Graphs, Structures, and Algorithms, vol 28. Algorithms and Combinatorics. Springer, New York (2012). https://doi.org/10.1007/978-3-642-27875-4
33. Poljak, S.: A note on stable sets and colorings of graphs. Commentationes Mathematicae Universitatis Carolinae **15**(2), 307–309 (1974)
34. Tedder, M., Corneil, D.G, Habib, M., Paul, C.: Simpler linear-time modular decomposition via recursive factorizing permutations. In: ICALP 2008 (1), vol. 5125. LNCS, pp. 634–645 (2008). https://doi.org/10.1007/978-3-540-70575-8_52

Affiliations

Hans L. Bodlaender[1] · Tesshu Hanaka[2] · Yasuaki Kobayashi[3] · Yusuke Kobayashi[3] · Yoshio Okamoto[4,5] · Yota Otachi[6] · Tom C. van der Zanden[7]

Hans L. Bodlaender
H.L.Bodlaender@uu.nl

Tesshu Hanaka
hanaka.91t@g.chuo-u.ac.jp

Yasuaki Kobayashi
kobayashi@iip.ist.i.kyoto-u.ac.jp

Yusuke Kobayashi
yusuke@kurims.kyoto-u.ac.jp

Yoshio Okamoto
okamotoy@uec.ac.jp

Tom C. van der Zanden
T.vanderZanden@maastrichtuniversity.nl

1 Utrecht University, Utrecht, The Netherlands

2 Chuo University, Bunkyo-ku, Tokyo, Japan

3 Kyoto University, Kyoto, Japan

4 The University of Electro-Communications, Chofu, Tokyo, Japan

[5] RIKEN Center for Advanced Intelligence Project, Tokyo, Japan

[6] Nagoya University, Nagoya 464-8601, Japan

[7] Maastricht University, Maastricht, The Netherlands