Algorithmica manuscript No.
(will be inserted by the editor)

Parameterized Complexity of Independent Set in H-Free Graphs

Édouard Bonnet · Nicolas Bousquet · Pierre Charbit · Stéphan Thomassé · Rémi Watrigant

Received: date / Accepted: date

Abstract In this paper, we investigate the complexity of Maximum Independent Set (MIS) in the class of H-free graphs, that is, graphs excluding a fixed graph as an induced subgraph. Given that the problem remains NP-hard for most graphs H, we study its fixed-parameter tractability and make progress towards a dichotomy between FPT and $W[1]$-hard cases. We first show that MIS remains $W[1]$-hard in graphs forbidding simultaneously $K_{1,4}$, any finite set of cycles of length at least 4, and any finite set of trees with at least two branching vertices. In particular, this answers an open question of Dabrowski *et al.* concerning C_4-free graphs. Then we extend the polynomial algorithm of Alekseev when H is a disjoint union of edges to an FPT algorithm when H is a disjoint union of cliques. We also provide a framework for

A preliminary version of this paper appeared in the proceedings of the 13th International Symposium on Parameterized and Exact Computation (IPEC 2018).

Édouard Bonnet
Université de Lyon, CNRS, ENS Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France
E-mail: edouard.bonnet@ens-lyon.fr

Nicolas Bousquet
CNRS, G-Scop Laboratory, Grenoble-INP, France
E-mail: nicolas.bousquet@grenoble-inp.fr

Pierre Charbit
Université Paris Diderot, IRIF, France
E-mail: charbit@irif.fr

Stéphan Thomassé
Université de Lyon, CNRS, ENS Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France
E-mail: stephan.thomasse@ens-lyon.fr

Rémi Watrigant
Université de Lyon, CNRS, ENS Lyon, Université Claude Bernard Lyon 1, LIP UMR5668, France
E-mail: remi.watrigant@ens-lyon.fr

solving several other cases, which is a generalization of the concept of *iterative expansion* accompanied by the extraction of a particular structure using Ramsey's theorem. Iterative expansion is a maximization version of the so-called *iterative compression*. We believe that our framework can be of independent interest for solving other similar graph problems. Finally, we present positive and negative results on the existence of polynomial (Turing) kernels for several graphs H.

Keywords Parameterized Algorithms · Independent Set · H-Free Graphs

1 Introduction

Given a simple graph G, a set of vertices $S \subseteq V(G)$ is an *independent set* if the vertices of this set are all pairwise non-adjacent. Finding an independent set with maximum cardinality is a fundamental problem in algorithmic graph theory, and is known as the MIS problem (MIS, for short) [15]. In general graphs, it is not only NP-hard, but also not approximable within $O(n^{1-\epsilon})$ for any $\epsilon > 0$ unless $P = NP$ [28], and $W[1]$-hard parameterized by the solution size [14] (unless otherwise stated, n always denotes the number of vertices of the input graph). Thus, it seems natural to study the complexity of MIS in restricted graph classes. One natural way to obtain such a restricted graph class is to forbid some given pattern to appear in the input. For a fixed graph H, we say that a graph is H*-free* if it does not contain H as an induced subgraph. Unfortunately, it turns out that for most graphs H, MIS in H-free graphs remains NP-hard, as shown by a very simple reduction observed independently by Poljak [24] and Alekseev [1]:

Theorem 1 ([1, 24]) *Let H be a connected graph which is neither a path nor a subdivision of the claw. Then* MIS *is NP-hard in H-free graphs.*

On the positive side, the case of P_t-free graphs has attracted a lot of attention during the last decade. While it is still open whether there exists $t \in \mathbb{N}$ for which MIS is NP-hard in P_t-free graphs, quite involved polynomial-time algorithms were discovered for P_5-free graphs [20], and very recently for P_6-free graphs [16]. In addition, we can also mention the recent following result: MIS admits a subexponential algorithm running in time $2^{O(\sqrt{tn \log n})}$ in P_t-free graphs for every $t \in \mathbb{N}$ [3].

The second open question concerns subdivisions of the claw. Let $S_{i,j,k}$ be a tree with exactly three vertices of degree one, being at distance i, j and k from the unique vertex of degree three. The complexity of MIS is still open in $S_{1,2,2}$-free graphs and $S_{1,1,3}$-free graphs. In this direction, the only positive results concern some subcases: it is polynomial-time solvable in $(S_{1,2,2}, S_{1,1,3}, dart)$-free graphs [18], $(S_{1,1,3}, banner)$-free graphs and $(S_{1,1,3}, bull)$-free graphs [19], where *dart*, *banner* and *bull* are particular graphs on five vertices.

Given the large number of graphs H for which the problem remains NP-hard, it seems natural to investigate the existence of fixed-parameter tractable

(FPT) algorithms[1], that is, determining the existence of an independent set of size k in a graph with n vertices in time $f(k)n^c$ for some computable function f and constant c. A very simple case concerns K_r-free graphs, that is, graphs excluding a clique of size r. In that case, Ramsey's theorem implies that every such graph G admits an independent set of size $\Omega(n^{\frac{1}{r-1}})$, where $n = |V(G)|$. In the FPT vocabulary, it implies that MIS in K_r-free graphs has a kernel with $O(k^{r-1})$ vertices.

To the best of our knowledge, the first step towards an extension of this observation within the FPT framework is the work of Dabrowski *et al.* [12] (see also Dabrowski's PhD manuscript [11]) who showed, among others, that for any positive integer r, Max Weighted Independent Set is FPT in H-free graphs when H is a clique of size r minus an edge. In the same paper, they settle the parameterized complexity of MIS on almost all the remaining cases of H-free graphs when H has at most four vertices. The conclusion is that the problem is FPT on those classes, except for $H = C_4$ which is left open. We answer this question by showing that MIS remains $W[1]$-hard in a subclass of C_4-free graphs. On the negative side, it was proved that MIS remains $W[1]$-hard in $K_{1,4}$-free graphs [17] We can also mention the case where H is the *bull* graph, which is a triangle with a pending vertex attached to two different vertices. For that case, a polynomial Turing kernel was obtained [27] then improved [9].

Finally, a subset of this paper's authors recently settled several other cases [5], such as the *cricket* graph, the $\bar{P}$ graph, or the path of size four where all but one endpoint are replaced by a clique of fixed size.

1.1 Our results

In Section 2, we present three reductions proving $W[1]$-hardness of MIS in graphs excluding several graphs as induced subgraphs, such as $K_{1,4}$, any fixed cycle of length at least four, and any fixed tree with two branching vertices. We actually show the stronger result that MIS remains W[1]-hard in graphs simultaneously excluding these graphs as induced subgraphs. We propose a definition of a graph decomposition whose aim is to capture all graphs which can be excluded using our reductions.

In Section 3, we extend the polynomial algorithm of Alekseev when H is a disjoint union of edges to an FPT algorithm when H is a disjoint union of cliques.

In Section 4, we present a general framework extending the technique of *iterative expansion*, which itself is the maximization version of the well-known iterative compression technique. We apply this framework to provide FPT algorithms when H is a clique minus a complete bipartite graph, when H is a clique minus a triangle, and when H is the so-called *gem* graph.

[1] For the sake of simplicity, "MIS" will denote the optimisation, decision and parameterized version of the problem (in the latter case, the parameter is the size of the solution), the correct use being clear from the context.

Finally, in Section 5, we focus on the existence of polynomial (Turing) kernels. We first strenghten some results of the previous section by providing polynomial (Turing) kernels in the case where H is a clique minus a claw. Then, we prove that for many H, MIS on H-free graphs does not admit a polynomial kernel, unless $NP \subseteq coNP/poly$.

Our results allow to obtain the complete quatrochotomy polynomial/polynomial kernel (PK)/no PK but polynomial Turing kernel/$W[1]$-hard for all possible graphs on four vertices.

1.2 Notation

For classical notation related to graph theory or fixed-parameter tractable algorithms, we refer the reader to the monographs [13] and [14], respectively. For an integer $r \geqslant 2$ and a graph H with vertex set $V(H) = \{v_1, \ldots, v_{n_H}\}$ with $n_H \leqslant r$, we denote by $K_r \setminus H$ the graph with vertex set $\{1, \ldots, r\}$ and edge set $\{ab : 1 \leqslant a, b \leqslant r$ such that $v_a v_b \notin E(H)\}$. For $X \subseteq V(G)$, we write $G \setminus X$ to denote $G[V(G) \setminus X]$. For two graphs G and H, we denote by $G \uplus H$ the *disjoint union* operation, that is, the graph with vertex set $V(G) \cup V(H)$ and edge set $E(G) \cup E(H)$. We denote by $G + H$ the *join* operation of G and H, that is, the graph with vertex set $V(G) \cup V(H)$ and edge set $E(G) \cup E(H) \cup \{uv : u \in V(G), v \in V(H)\}$. For two integers r, k, we denote by $Ram(r, k)$ the Ramsey number of r and k, *i.e.* the number such that every graph with at least $Ram(r, k)$ vertices contains either a clique of size r or an independent set of size k. We write for short $Ram(k) = Ram(k, k)$. Finally, for $\ell, k > 0$, we denote by $Ram_\ell(k)$ the minimum order of a complete graph whose edges are colored with ℓ colors to contain a monochromatic clique of size k. The following bounds are known: $Ram(r, k) \leqslant \binom{r+k-2}{r-1} = \binom{r+k-2}{k-1}$, and $Ram_\ell(k) \leqslant k^{\ell k}$.

2 $W[1]$-hardness

2.1 Main reduction

We show the following:

Theorem 2 *For any $p_1 \geqslant 4$ and $p_2 \geqslant 1$,* MIS *remains $W[1]$-hard in graphs excluding simultaneously the following graphs as induced subgraphs:*

- $K_{1,4}$
- $C_4, \ldots, C_{p_1}$
- *any tree T with two branching vertices[2] at distance at most p_2.*

Proof. Let $p = \max\{p_1, p_2\}$. We reduce from Grid Tiling, where the input is composed of k^2 sets $S_{i,j} \subseteq [m] \times [m]$ $(0 \leqslant i, j \leqslant k-1)$, called *tiles*, each

[2] A branching vertex in a tree is a vertex of degree at least 3.

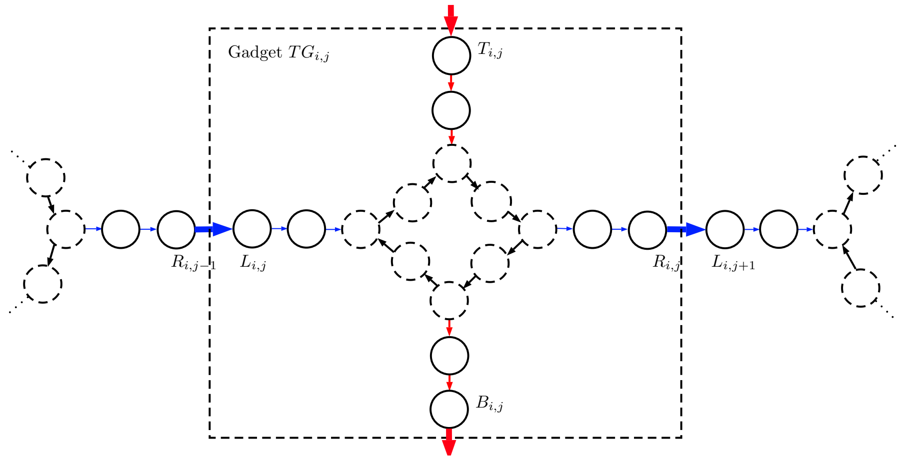


Fig. 1 Gadget $TG_{i,j}$ representing a tile and its adjacencies with $TG_{i,j-1}$ and $TG_{i,j+1}$, for $p = 1$. Each circle is a main clique on n vertices: dashed cliques are the cycle cliques (those of them connected to three other cliques are branching cliques), while others are path cliques. Black, blue and red arrows represent respectively type T_h, T_r and T_c edges (bold arrows are between two gadgets). Figures 2 and 3 represent some adjacencies in more details.

composed of n elements. The objective of Grid Tiling is to find an element $s^*_{i,j} \in S_{i,j}$ for each $0 \leqslant i, j \leqslant k-1$, such that $s^*_{i,j}$ agrees in the first coordinate with $s^*_{i,j+1}$, and agrees in the second coordinate with $s^*_{i+1,j}$, for every $0 \leqslant i, j \leqslant k-1$ (here and henceforth, $i+1$ and $j+1$ are taken modulo k). In such case, we say that $\{s^*_{i,j},\ 0 \leqslant i, j \leqslant k-1\}$ is a *feasible solution* of the instance. It is known that Grid Tiling is $W[1]$-hard parameterized by k [10, 21].

Before describing formally the reduction, let us give some definitions and ideas. Given $s = (a, b)$ and $s' = (a', b')$, we say that s is *row-compatible* (resp. *column-compatible*) with s' if $a \geqslant a'$ (resp. $b \geqslant b'$)[3]. Observe that a solution $\{s^*_{i,j},\ 0 \leqslant i, j \leqslant k-1\}$ is feasible if and only if $s^*_{i,j}$ is row-compatible with $s^*_{i,j+1}$ and column-compatible with $s^*_{i+1,j}$ for every $0 \leqslant i, j \leqslant k-1$.

We will represent each tile by a gadget partitioned into a constant number of cliques of size n. The vertices of each clique are in one-to-one correspondence with the elements of the corresponding tile. Overall the cliques will be arranged in a grid-like structure with degree three. By that we mean that a clique will be linked to at most three other cliques. While most of the cliques will have only two neighboring cliques, a clique linked to three other cliques will be called *branching clique*. The row-compatibility (resp. column-compatibility) will be encoded with a relatively simple interaction between two adjacent cliques. The main difficulty will be to prevent the undesired induced subgraphs to appear in the vicinity of branching cliques. We now formally describe the reduction.

[3] Notice that the row-compatibility (resp. column-compatibility) relation is not symmetric.

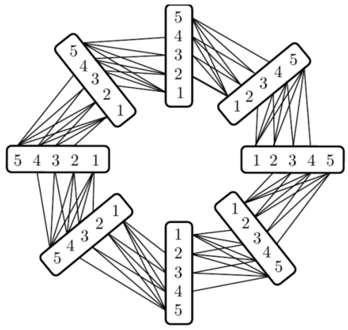


Fig. 2 Adjacencies between cycle cliques (represented by dashed circles in Figure 1).

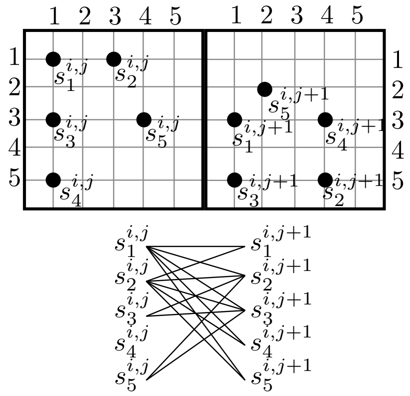


Fig. 3 Two consecutive tiles and the representation of their adjacencies (representing type T_r adjacencies).

2.1.1 Tile gadget.

For every tile $S_{i,j} = \{s_1^{i,j}, \ldots, s_n^{i,j}\}$, we construct a *tile gadget* $TG_{i,j}$, depicted in Figure 1. Notice that this gadget shares some ideas with the $W[1]$-hardness proof for MIS in $K_{1,4}$-free graphs by Hermelin *et al.* [17]. To define this gadget, we first describe an oriented graph with three types of arcs (type T_h, T_r and T_c, which respectively stands for *half-graph*, *row* and *column*, and this naming will become clearer later), and then explain how to represent the vertices and arcs of this graph to get the concrete gadget. Consider first a directed cycle on $4p+4$ vertices c_1, …, c_{4p+4} with arcs of type T_h. Then consider four oriented paths on $p+1$ vertices: P_1, P_2, P_3 and P_4. P_1 and P_3 are composed of arcs of type T_c, while P_2 and P_4 are composed of arcs of type T_r. Put an arc of type T_c between:

- the last vertex of P_1 and c_1,
- c_{2p+3} and the first vertex of P_3,

and an arc of type T_r between:

– c_{p+2} and the first vertex of P_2,
– the last vertex of P_4 and c_{3p+4}.

Now, we replace every vertex of this oriented graph by a clique on n vertices, and fix an arbitrary ordering on the vertices of each clique. The i^{th} vertex in this ordering is said to *have index i*. For each arc of type T_h between c and c', add a half-graph[4] between the corresponding cliques: connect the a^{th} vertex of the clique representing c with the b^{th} vertex of the clique representing c' whenever $a > b$. For every arc of type T_r from a vertex c to a vertex c', connect the a^{th} vertex of the clique representing c with the b^{th} vertex of the clique representing c' iff $s_a^{i,j}$ is *not* row-compatible with $s_b^{i,j}$. Similarly, for every arc of type T_c from a vertex c to a vertex c', connect the a^{th} vertex of the clique representing C with the b^{th} vertex of the clique representing c' iff $s_a^{i,j}$ is *not* column-compatible with $s_b^{i,j}$.

The cliques corresponding to vertices of this gadget are called the *main cliques* of $TG_{i,j}$, and the cliques corresponding to the central cycle on $4p+4$ vertices are called the *cycle cliques*. The main cliques which are not cycle cliques are called *path cliques*. The cycle cliques adjacent to one path clique are called *branching cliques*. We call *cycle of cliques* the set of all cycle cliques present in the same gadget $TG_{i,j}$. Two cycle of cliques are said *consecutive* if they lie on two gadgets $TG_{i,j}$ and $TG_{i,j+1}$, or $TG_{i,j}$ and $TG_{i+1,j}$. A *path of cliques* is any subgraph induced by the cliques corresponding to vertices forming a directed path in the oriented preliminary graph.

Finally, the clique corresponding to the vertex of degree one in the path attached to c_1 (resp. c_{p+2}, c_{2p+3}, c_{3p+4}) is called the *top* (resp. *right*, *bottom*, *left*) clique of $TG_{i,j}$, denoted by $T_{i,j}$ (resp. $R_{i,j}$, $B_{i,j}$, $L_{i,j}$). Let $T_{i,j} = \{t_1^{i,j}, \ldots, t_n^{i,j}\}$, $R_{i,j} = \{r_1^{i,j}, \ldots, r_n^{i,j}\}$, $B_{i,j} = \{b_1^{i,j}, \ldots, b_n^{i,j}\}$, and $L_{i,j} = \{\ell_1^{i,j}, \ldots, \ell_n^{i,j}\}$. For the sake of readability, we might omit the superscripts i, j when it is clear from the context.

Lemma 1 *Let K be an independent set of size $8(p+1)$ in $TG_{i,j}$. Then:*

(a) K intersects all the cycle cliques on the same index $x \in [n]$;
(b) if $K \cap T_{i,j} = \{t_{x_t}\}$, $K \cap R_{i,j} = \{r_{x_r}\}$, $K \cap B_{i,j} = \{b_{x_b}\}$, and $K \cap L_{i,j} = \{\ell_{x_\ell}\}$. Then:
  – *$s_{x_\ell}^{i,j}$ is row-compatible with $s_x^{i,j}$ which is row-compatible with $s_{x_r}^{i,j}$, and*
  – *$s_{x_t}^{i,j}$ is column-compatible with $s_x^{i,j}$ which is column-compatible with $s_{x_b}^{i,j}$.*

Proof. Observe that the vertices of $TG_{i,j}$ can be partitioned into $8(p+1)$ cliques (the main cliques), hence an independent set of size $8(p+1)$ intersects each main clique on exactly one vertex. Let C_1, C_2 and C_3 be three consecutive cycle cliques, and suppose K intersects C_1 (resp. C_2, C_3) on the x_1^{th} (resp. x_2^{th}, x_3^{th}) index. By definition of the gadget, it implies $x_1 \leqslant x_2 \leqslant x_3$. By applying

[4] Notice that our definition of half-graph slighly differs from the usual one, in the sense that we do not put edges relying two vertices of the same index. Hence, our construction can actually be seen as the complement of a half-graph (which is consistent with the fact that usually, both parts of a half-graph are independent sets, while they are cliques in our gadgets).

the same argument from C_3 along the cycle, we obtain $x_3 \leqslant x_1$, which proves (a). The proof of (b) directly comes from the definition of the adjacencies between cliques of type T_r and T_c, and from the fact that K intersects all cycle cliques on the same index. □

2.1.2 Attaching gadgets together.

For $i, j \in \{0, \ldots, k-1\}$, we connect the right clique of $TG_{i,j}$ with the left clique of $TG_{i,j+1}$ in a "type T_r spirit": for every $x, y \in [n]$, connect $r_x^{i,j} \in R_{i,j}$ with $\ell_y^{i,j+1} \in L_{i,j+1}$ iff $s_x^{i,j}$ is *not* row-compatible with $s_y^{i,j+1}$. Similarly, we connect the bottom clique of $TG_{i,j}$ with the top clique of $TG_{i+1,j}$ in a "type T_c spirit": for every $x, y \in [n]$, connect $b_x^{i,j} \in B_{i,j}$ with $t_y^{i+1,j} \in T_{i+1,j}$ iff $s_x^{i,j}$ is *not* column-compatible with $s_y^{i+1,j}$ (all incrementations of i and j are done modulo k). This terminates the construction of the graph G.

2.1.3 Equivalence of solutions.

We now prove that the input instance of Grid Tiling is positive if and only if G has an independent set of size $k' = 8(p+1)k^2$. First observe that G has k^2 tile gadgets, each composed of $8(p+1)$ main cliques, hence any independent set of size k' intersects each main clique on exactly one vertex. By Lemma 1, for all $i, j \in \{0, \ldots, k-1\}$, K intersects the cycle cliques of $TG_{i,j}$ on the same index $x_{i,j}$. Moreover, if $K \cap R_{i,j} = \{r_x^{i,j}\}$ and $K \cap L_{i,j+1} = \{\ell_{x'}^{i,j+1}\}$, then, by construction of G, $s_x^{i,j}$ is row-compatible with $s_{x'}^{i,j+1}$. Similarly, if $K \cap B_{i,j} = \{b_x^{i,j}\}$ and $K \cap T_{i+1,j} = \{t_{x'}^{i+1,j}\}$, then, by construction of G, $s_x^{i,j}$ is column-compatible with $s_{x'}^{i+1,j}$. By Lemma 1, it implies that $s_{x_{i,j}}^{i,j}$ is row-compatible with $s_{x_{i,j+1}}^{i,j+1}$ and column-compatible with $s_{x_{i+1,j}}^{i+1,j}$ (incrementations of i and j are done modulo k), thus $\{x_{x_{i,j}}^{i,j} : 0 \leqslant i, j \leqslant k-1\}$ is a feasible solution. Using similar ideas, one can prove that a feasible solution of the grid tiling instance implies an independent set of size k' in G.

2.1.4 Structure of the obtained graph.

Let us now prove that G does not contain the graphs mentioned in the statement as an induced subgraph:

No $\mathbf{K_{1,4}}$. We first prove that for every $0 \leqslant i, j \leqslant k-1$, the graph induced by the cycle cliques of $TG_{i,j}$ is claw-free. For the sake of contradiction, suppose that there exist three consecutive cycle cliques A, B and C containing a claw. W.l.o.g. we may assume that $b_x \in B$ is the center of the claw, and $a_\alpha \in A$, $b_\beta \in B$ and $c_\gamma \in C$ are the three endpoints. By construction of the gadgets (there is a half-graph between A and B and between B and C), we must have $\alpha < x < \gamma$. Now, observe that if $x < \beta$ then a_α must be adjacent to b_β, and if $\beta < x$, then b_β must be adjacent to c_γ, but both case are impossible since $\{a_\alpha, b_\beta, c_\gamma\}$ is supposed to be an independent set.

Similarly, each subgraph induced by P, a path of size $2(p+1)$ of cliques linking two consecutive cycles of cliques, is claw-free. Hence, for $K_{1,4}$ to appear in G its center would have to lie in a branching clique. However, in that case, a claw must exist either in the cycle of cliques or in P, which we already ruled out.

No $\mathbf{C_4, \ldots, C_{p_1}}$. The main argument is that the graph induced by any two main cliques does not contain any of these cycles. Then, we show that such a cycle cannot lie entirely in the cycle cliques of a single gadget $TG_{i,j}$. Indeed, if this cycle uses at most one vertex per main clique, then it must be of length at least $4p+4$. If it intersects a clique C on two vertices, then either it also intersect all the cycle cliques of the gadget, in which case it is of length $4p+5$, or it intersects an adjacent clique of C on two vertices, in which case these two cliques induce a C_4, which is impossible. Similarly, such a cycle cannot lie entirely in a path between the main cliques of two gadgets. Finally, the main cliques of two gadgets are at distance at least $2(p+1)$, hence such a cycle cannot intersect the main cliques of two gadgets.

No tree T with two branching vertices at distance at most $\mathbf{p_2}$. Using the same argument as for the $K_{1,4}$ case, observe that the claws contained in G can only appear in the cycle cliques where the paths are attached. However, observe that these cliques are at distance $2(p+1) > p_2$, thus, such a tree T cannot appear in G. □

As a direct consequence of Theorem 2, we get the following by setting $p_1 = p_2 = |V(H)| + 1$:

Corollary 1 *If H is not chordal, or contains as an induced subgraph a $K_{1,4}$ or a tree with two branching vertices, then* MIS *in H-free graphs is $W[1]$-hard.*

2.2 Capturing Hard Graphs

We introduce two variants of the hardness construction of Theorem 2, which we refer to as the *first construction.* The *second construction* is obtained by replacing each interaction between two main cliques by an anti-matching, except the one interaction in the middle of the path cliques which remains a half-graph (see Figure 4, middle). In an anti-matching, the same elements in the two adjacent cliques define the only non-edges. The correctness of this new reduction is simpler since the propagation of a choice is now straightforward. Observe however that the graph C_4 appears in this new construction. For the *third construction*, we start from the second construction and just add an anti-matching between two neighbors of each branching clique among the cycle cliques (see Figure 4, right). This anti-matching only constrains more the instance but does not destroy the intended solutions; hence the correctness.

To describe those connected graphs H which escape the disjunction of Theorem 2 (for which there is still a hope that MIS is FPT), we define a decomposition into cliques, similar yet different from clique graphs or tree decompositions of chordal graphs (a.k.a k-trees).

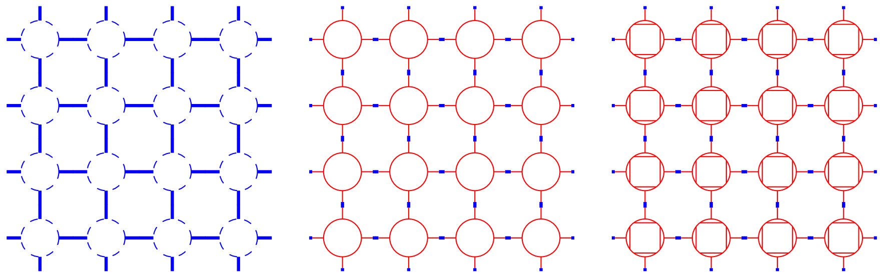


Fig. 4 A symbolic representation of the hardness constructions. To the left, only half-graphs (blue) are used between the cliques, as in the proof of Theorem 2. In the middle and to the right, the half-graphs (blue) are only used once in the middle of each path of cliques, and the rest of the interactions between the cliques are anti-matchings (red). The third construction (right) is a slight variation of the second (middle) where for each branching clique, we link by an anti-matching its two neighbors among the cycle cliques.

Definition 1 Let T be a graph on ℓ vertices $t_1, \dots, t_\ell$. We say that T *is a clique decomposition of* H if there is a partition of $V(H)$ into $(C_1, C_2, \dots, C_\ell)$ such that:

- for each $i \in [\ell]$, $H[C_i]$ is a clique, and
- for each pair $i \neq j \in [\ell]$, if $H[C_i \cup C_j]$ is connected, then $t_i t_j \in E(T)$.

Observe that, in the above definition, we do not require T to be a tree. Two cliques C_i and C_j are said *adjacent* if $H[C_i \cup C_j]$ is connected. We also write *a clique decomposition on* T *(of* H*)* to denote the choice of an actual partition $(C_1, C_2, \dots, C_\ell)$.

Let $\mathcal{T}_1$ be the class of trees with at most one branching vertex. Equivalently, $\mathcal{T}_1$ consists of paths and subdivisions of the claw.

Proposition 1 *For a fixed connected graph* H*, if no tree in* $\mathcal{T}_1$ *is a clique decomposition of* H*, then* MIS *in* H*-free graphs is* $W[1]$*-hard.*

Proof. This is immediate from the proof of Theorem 2 since H cannot appear in the first construction. □

At this point, we can focus on connected graphs H admitting a tree $T \in \mathcal{T}_1$ as a clique decomposition. The reciprocal of Proposition 1 cannot be true since a simple edge is a clique decomposition of C_4. The next definition further restricts the interaction between two adjacent cliques.

Definition 2 Let T be a graph on ℓ vertices $t_1, \dots, t_\ell$. We say that T *is a strong clique decomposition of* H if there is a partition of $V(H)$ into $(C_1, \dots, C_\ell)$ such that:

- for each $i \in [\ell]$, $H[C_i]$ is a clique,
- for each $t_i t_j \in E(T)$, $H[C_i \cup C_j]$ is a clique, and
- for each $t_i t_j \notin E(T)$, there is no edge between C_i and C_j.

An equivalent way to phrase this definition is that H can be obtained from T by *adding false twins*. Adding a false twin v' to a graph consists of duplicating one of its vertex v (i.e., v and v' have the same neighbors) and then adding an edge between v and v'.

We define *almost strong clique decompositions* which informally are strong clique decompositions where at most one edge can be missing in the interaction between two adjacent cliques.

Definition 3 Let T be a graph on ℓ vertices $t_1, \dots, t_\ell$. We say that T *is an almost strong clique decomposition of* H if there is a partition of $V(H)$ into $(C_1, \dots, C_\ell)$ such that:

- for each $i \in [\ell]$, $H[C_i]$ is a clique,
- for each $t_i t_j \in E(T)$, $H[C_i \cup C_j]$ is a clique potentially deprived of a single edge, and is connected, and
- for each $t_i t_j \notin E(T)$, there is no edge between C_i and C_j.

Finally, a *nearly strong clique decomposition* is slightly weaker than an almost strong clique decomposition: at most one interaction between two adjacent cliques is only required to be C_4-free. Formally:

Definition 4 Let T be a graph on ℓ vertices $t_1, \dots, t_\ell$ with a special edge $t_a t_b$. We say that T *is a nearly strong clique decomposition of* H if there is a partition of $V(H)$ into $(C_1, \dots, C_\ell)$ such that:

- for each $i \in [\ell]$, $H[C_i]$ is a clique,
- $H[C_a \cup C_b]$ is C_4-free,
- for each $t_i t_j \in E(T) \setminus \{t_a t_b\}$, $H[C_i \cup C_j]$ is a clique potentially deprived of a single edge, and is connected, and
- for each $t_i t_j \notin E(T)$, there is no edge between C_i and C_j.

Let $\mathcal{P}$ be the set of all the paths. Notice that $\mathcal{T}_1 \setminus \mathcal{P}$ is the set of all the subdivisions of the claw.

Theorem 3 *Let* H *be a fixed connected graph. If no* $P \in \mathcal{P}$ *is a nearly strong clique decomposition of* H *and no* $T \in \mathcal{T}_1 \setminus \mathcal{P}$ *is an almost strong clique decomposition of* H*, then* MIS *in* H*-free graphs is* $W[1]$*-hard.*

Proof. The idea is to mainly use the second construction and the fact that MIS in C_4-free graphs is $W[1]$-hard (due to the first construction). For every fixed graph H which cannot be an induced subgraph in the second construction, MIS is $W[1]$-hard. To appear in this construction, the graph H should have

- either a clique decomposition on a subdivision of the claw, such that the interaction between two adjacent cliques is the complement of a (non necessarily perfect) matching, or
- a clique decomposition on a path, such that the interaction between two adjacent cliques is the complement of a matching, except for at most one interaction which can be a C_4-free graph.

We now just observe that in both cases if, among the interactions between adjacent cliques, one complement of matching has at least two non-edges, then H contains an induced C_4. Hence the two items can be equivalently replaced by the existence of an almost strong clique decomposition on a subdivision of the claw, and a nearly strong clique decomposition on a path, respectively. □

Theorem 3 narrows down the connected open cases to graphs H which have a nearly strong clique decomposition on a path or an almost strong clique decomposition on a subdivision of the claw.

In the strong clique decomposition, the interaction between two adjacent cliques is very simple: their union is a clique. Therefore, it might be tempting to conjecture that if H admits $T \in \mathcal{T}_1$ as a strong clique decomposition, then MIS in H-free graphs is FPT. Indeed, those graphs H appear in both the first and the second $W[1]$-hardness constructions. Nevertheless, we will see that this conjecture is false: even if H has a strong clique decomposition $T \in \mathcal{T}_1$, it can be that MIS is $W[1]$-hard. The simplest tree of $\mathcal{T}_1 \setminus \mathcal{P}$ is the claw. We denote by $T_{i,j,k}$ the graph obtained by adding a universal vertex to the disjoint union of three cliques $K_i \uplus K_j \uplus K_k$. The claw is a strong clique decomposition of $T_{i,j,k}$ (for every natural numbers i, j, k).

Theorem 4 MIS *in $T_{1,2,2}$-free graphs is $W[1]$-hard.*

Proof. We show that $T_{1,2,2}$ does not appear in the third construction (Figure 4, right). We claim that, in this construction, the graph $T_{1,1,2}$, sometimes called cricket, can only appear in the two ways depicted on Figure 5 (up to symmetry).

Claim 5. *The triangle of the cricket cannot appear within the same main clique.*

Proof of claim: Otherwise the two leaves (*i.e.*, vertices of degree 1) of the cricket are in two distinct adjacent cliques. But at least one of those adjacent cliques is linked to the main clique of the triangle by an anti-matching. This is a contradiction to the corresponding leaf having two non-neighbors in the main clique of the triangle. ◁

We first study how the cricket can appear in a path of cliques. Let C be the main clique containing the universal vertex of the cricket. This vertex is adjacent to three disjoint cliques $K_1 \uplus K_1 \uplus K_2$. Due to the previous claim, the only way to distribute them is to put K_1 in the previous main clique, K_1 in the same main clique C, and K_2 in the next main clique. This is only possible if the interaction between C and the next main clique is a half-graph. In particular, this implies that the interaction between the previous main clique and C is an anti-matching. This situation corresponds to the left of Figure 5.

This also implies that the cricket cannot appear in a path of cliques without a half-graph interaction (anti-matchings only). We now turn our attention to the vicinity of a triangle of main cliques, which is proper to the third construction. By our previous remarks, we know that the universal vertex of

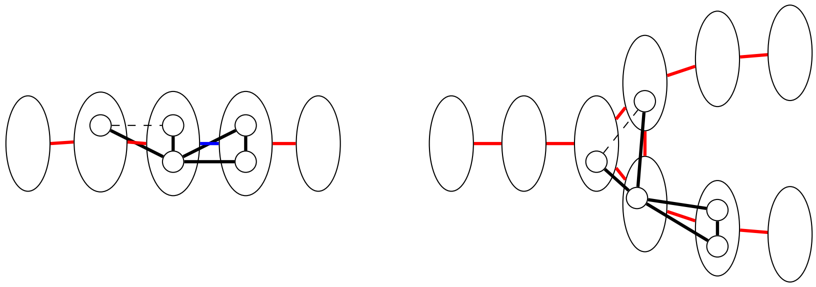


Fig. 5 The two ways the cricket appears in the third construction. The red edges between two adjacent cliques symbolize an anti-matching, whereas the blue edge symbolizes a C_4-free graph. In the left hand-side, one neighbor of the universal vertex with degree 2 could alternatively be in the same clique as the universal vertex.

the cricket has to be either alone in a main clique (by symmetry, it does not matter which one) of the triangle, or with exactly one of its neighbors of degree 2. Now, the only way to place $K_1 \uplus K_1 \uplus K_2$ is to put the two K_1 in the two other main cliques of the triangle, and the K_2 (or the single vertex rest of it) in the remaining adjacent main clique. Indeed, if the K_2 is in a main clique of the triangle, the K_1 in the third main clique of the triangle would have two non-edges towards to K_2. This is not possible with an anti-matching interaction. Therefore, the only option corresponds to the right of Figure 5.

To obtain a $T_{1,2,2}$, one needs to find a false twin to one of the leaves of the cricket. This is not possible since, in both cases, the two leaves are in two adjacent cliques with an anti-matching interaction. Therefore, adding the false twin would create a second non-neighbor to the remaining leaf. □

The graph $T_{1,1,1}$ is the claw itself for which MIS is solvable in polynomial time. The parameterized complexity for the graph $T_{1,1,2}$ (the cricket) remains open. As a matter of fact, this question is unresolved for $T_{1,1,s}$-free graphs, for any integer $s \geqslant 2$. Solving those cases would bring us a bit closer to a full dichotomy *FPT vs* $W[1]$*-hard*. Although, Theorem 4 suggests that this dichotomy will be rather subtle. In addition, this result infirms the plausible conjecture: *if* MIS *is FPT in* H*-free graphs, then it is FPT in* H'*-free graphs where* H' *can be obtained from* H *by adding false twins.*

The toughest challenge towards the dichotomy is understanding MIS in the absence of *paths of cliques*[5]. In Theorem 11, we make a very first step in that direction: we show that for every graph H with a strong clique decomposition on P_3, the problem is FPT. In the previous paragraphs, we dealt mostly with connected graphs H. In Theorem 6, we show that if H is a disjoint union of cliques, then MIS in H-free graphs is FPT. In the language of clique decompositions, this can be phrased as *H has a clique decomposition on an edgeless graph.*

[5] Actually, even the classical complexity of MIS in the absence of long induced paths is not well understood

3 Positive results I: disjoint union of cliques

For $r, q \geqslant 1$, let K_r^q be the disjoint union of q copies of K_r. The proof of the following theorem is inspired by the case $r = 2$ by Alekseev [2].

Theorem 6 Maximum Independent Set *is FPT in K_r^q-free graphs.*

Proof. We will prove by induction on q that a K_r^q-free graph has an independent set of size k or has at most $Ram(r,k)^{qk}n^{qr}$ independent sets. This will give the desired FPT-algorithm, as the proof shows how to construct this collection of independent sets. Note that the case $q = 1$ is trivial by Ramsey's theorem. We also assume $r \geq 3$, since the case $r = 2$ corresponds to Alekseev's algorithm[2].

Let G be a K_r^q-free graph and let $<$ be any fixed total ordering of $V(G)$ such that the largest vertex in this ordering belongs to a clique of size r (the case where G is K_r-free is trivial by Ramsey's theorem). Since a clique of size r can be found in polynomial time, such an ordering can be found in polynomial time. For any vertex x, define $x^+ = \{y, x < y\}$ and $x^- = V(G) \setminus x^+$.

Let us first explain how we will generate independent sets. We will prove next that the algorithm generates all of them. Let C be a fixed clique of size r in G and let c be the largest vertex of C with respect to $<$. Let V_1 be the set of vertices of c^+ which have no neighbor in C. Note that V_1 induces a K_r^{q-1}-free graph, so by induction either it contains an independent set of size k, and so does G, or it has at most $Ram(r,k)^{(q-1)k}n^{(q-1)r}$ independent sets. In the latter case, let $\mathcal{S}_1$ be the set of all independent sets of $G[V_1]$. Now in a second phase we define an initially empty set $\mathcal{S}_C$ and do the following. For each independent set S_1 in $\mathcal{S}_1$ (including the empty set), we denote by V_2 the set of vertices in c^- that have no neighbor in S_1. For every choice of a vertex x amongst the largest $Ram(r,k)$ vertices of V_2 in the order, we add x to S_1 and modify V_2 in order to keep only vertices that are smaller than x (with respect to $<$) and non adjacent to x. We repeat this operation $k-1$ times (or until V_2 becomes empty). At the end, we either find an independent set of size k (if V_2 is still not empty) or add S_1 to $\mathcal{S}_C$ (when V_2 becomes empty). By doing so we construct a family of at most $Ram(r,k)^k$ independent sets for each S_1, so in total we get indeed at most $Ram(r,k)^{kq}n^{(q-1)r}$ independent sets for each clique C. Finally we define $\mathcal{S}$ as the union over all r-cliques C of the sets $\mathcal{S}_C$, so that $\mathcal{S}$ has size at most the desired number.

We claim that if G does not contain an independent set of size k, then $\mathcal{S}$ contains all independent sets of G. It suffices to prove that for every independent set S, there exists a clique C for which $S \in \mathcal{S}_C$. Let S be an independent set, and define C to be a clique of size r such that its largest vertex c (with respect to $<$) satisfies the conditions:

- no vertex of C is adjacent to a vertex of $S \cap c^+$, and
- c is the smallest vertex such that a clique C satisfying the first item exists.

First remark that such a clique always exist, since we assumed that the largest vertex c_{last} of $<$ is contained in a clique of size r, which means that $S \cap c_{last}^+$

is empty and thus the first item is vacuously satisfied. Secondly, note that several cliques C might satisfy the two previous conditions. In that case, pick one such clique arbitrarily. This definition of C and c ensures that $S \cap c^+$ is an independent set in the set V_1 defined in the construction above (it might be empty). Thus, it will be picked in the second phase as some S_1 in $\mathcal{S}_1$ and for this choice, each time V_2 is considered, the fact that C is chosen to minimize its largest element c guarantees that there must be a vertex of S in the $Ram(r, k)$ largest vertices in V_2, otherwise we could find within those vertices an r-clique contradicting the choice of C (we can find an r-clique satisfying both points such that the maximum vertex is smaller than c). So it ensures that we will add S to the collection $\mathcal{S}_C$, which concludes our proof. □

4 Positive results II

4.1 Key ingredient: Iterative expansion and Ramsey extraction

In this section, we present the main idea of our algorithms. It is a generalization of iterative expansion, which itself is the maximization version of the well-known iterative compression technique. Iterative compression is a useful tool for designing parameterized algorithms for subset problems (*i.e.* problems where a solution is a subset of some set of elements: vertices of a graph, variables of a logic formula...*etc.*) [10, 25]. Although it has been mainly used for minimization problems, iterative compression has been successfully applied for maximization problems as well, under the name *iterative expansion* [7]. Roughly speaking, when the problem consists of finding a solution of size at least k, the iterative expansion technique consists of solving the problem where a solution S of size $k-1$ is given in the input, in the hope that this solution will imply some structure in the instance. In the following, we consider an extension of this approach where, instead of a single smaller solution, one is given a set of $f(k)$ smaller solutions $S_1, \ldots, S_{f(k)}$. As we will see later, we can further add more constraints on the sets $S_1, \ldots, S_{f(k)}$. Notice that all the results presented in this sub-section (Lemmas 2 and 3 in particular) hold for any hereditary graph class (including the class of all graphs). The use of properties inherited from particular graphs (namely, H-free graphs in our case) will only appear in Sections 4.2 and 4.3.

Definition 5 For a function $f : \mathbb{N} \to \mathbb{N}$, the f-ITERATIVE EXPANSION MIS problem takes as input a graph G, an integer k, and a set of $f(k)$ vertex-disjoint independent sets $S_1, \ldots, S_{f(k)}$, each of size $k-1$. The objective is to find an independent set of size k in G, or to decide that such an independent set does not exist.

Lemma 2 *Let $\mathcal{G}$ be a hereditary graph class.* MIS *is FPT in $\mathcal{G}$ iff* f-ITERATIVE EXPANSION MIS *is FPT in $\mathcal{G}$ for some computable function $f : \mathbb{N} \to \mathbb{N}$.*

Proof. Clearly if MIS is FPT, then f-Iterative Expansion MIS is FPT for any computable function f. Conversely, let f be a function for which f-Iterative Expansion MIS is FPT, and let G be a graph with $|V(G)| = n$.

We show by induction on k that there is an algorithm that either finds an independent set of size k, or answers that such a set does not exist, in FPT time parameterized by k. The initialization can obviously be computed in constant time. Assume we have an algorithm for $k-1$. Successively for i from 1 to $f(k)$, we construct an independent set S_i of size $k-1$ in $G \setminus (S_1, \dots, S_{i-1})$. If, for some i, we are unable to find such an independent set, then it implies that any independent set of size k in G must intersect $S_1 \cup \cdots \cup S_{i-1}$. We thus branch on every vertex v of this union, and, by induction, find an independent set of size $k-1$ in the graph induced by $V(G) \setminus N[v]$. If no step i triggered the previous branching, we end up with $f(k)$ vertex-disjoint independent sets $S_1, \dots, S_{f(k)}$, each of size $k-1$. We now invoke the algorithm for f-Iterative Expansion MIS to conclude. Let us analyze the running time of this algorithm: each step either branches on at most $f(k)(k-1)$ subcases with parameter $k-1$, or concludes in time $\mathcal{A}_f(n,k)$, the running time of the algorithm for f-Iterative Expansion MIS. Hence the total running time is $O^*(f(k)^k(k-1)^k\mathcal{A}_f(n,k))$, where the $O^*(.)$ suppresses polynomial factors. □

We will actually prove a stronger version of this result, by adding more constraints on the input sets $S_1, \dots, S_{f(k)}$, and show that solving the expansion version on this particular kind of input is enough to obtain the result for MIS.

Definition 6 Given a graph G and a set of $k-1$ vertex-disjoint cliques of G, $\mathcal{C} = \{C_1, \dots, C_{k-1}\}$, each of size q, we say that $\mathcal{C}$ is a set of *Ramsey-extracted cliques of size* q if the conditions below hold. Let $C_r = \{c_j^r : j \in \{1, \dots, q\}\}$ for every $r \in \{1, \dots, k-1\}$.

- For every $j \in [q]$, the set $\{c_j^r : r \in \{1, \dots, k-1\}\}$ is an independent set of G of size $k-1$.
- For any $r \neq r' \in \{1, \dots, k-1\}$, one of the four following case can happen:
  (i) for every $j, j' \in [q]$, $c_j^r c_{j'}^{r'} \notin E(G)$
  (ii) for every $j, j' \in [q]$, $c_j^r c_{j'}^{r'} \in E(G)$ iff $j \neq j'$
  (iii) for every $j, j' \in [q]$, $c_j^r c_{j'}^{r'} \in E(G)$ iff $j < j'$
  (iv) for every $j, j' \in [q]$, $c_j^r c_{j'}^{r'} \in E(G)$ iff $j > j'$
  In the case (i) (resp. (ii)), we say that the relation between C_r and $C_{r'}$ is *empty* (resp. *full*[6]). In case (iii) or (iv), we say the relation is *semi-full*.

Observe, in particular, that a set $\mathcal{C}$ of $k-1$ Ramsey-extracted cliques of size q can be partitioned into q independent sets of size $k-1$. As we will see later, these cliques will allow us to obtain more structure with the remaining vertices if the graph is H-free. Roughly speaking, if q is large, we will be able to

[6] Remark that in this case, the graph induced by $C_r \cup C_{r'}$ is the complement of a perfect matching.

extract from $\mathcal{C}$ another set $\mathcal{C}'$ of $k-1$ Ramsey-extracted cliques of size $q' < q$, such that every clique is a module[7] with respect to the solution $x_1^*, \ldots, x_k^*$ we are looking for. Then, by guessing the structure of the adjacencies between $\mathcal{C}'$ and the solution, we will be able to identify from the remaining vertices k sets $X_1, \ldots, X_k$, where each X_i has the same neighborhood as x_i^* *w.r.t.* $\mathcal{C}'$, and plays the role of "candidates" for this vertex. For a function $f : \mathbb{N} \to \mathbb{N}$, we define the following problem:

Definition 7 The f-Ramsey-extracted Iterative Expansion MIS problem takes as input an integer k and a graph G whose vertices are partitioned into non-empty sets $X_1 \cup \cdots \cup X_k \cup C_1 \cup \cdots \cup C_{k-1}$, where:

- $\{C_1, \ldots, C_{k-1}\}$ is a set of $k-1$ Ramsey-extracted cliques of size $f(k)$
- any independent set of size k in G is contained in $X_1 \cup \cdots \cup X_k$
- $\forall i \in \{1, \ldots, k\}$, $\forall v, w \in X_i$ and $\forall j \in \{1, \ldots, k-1\}$, $N(v) \cap C_j = N(w) \cap C_j = \emptyset$ or $N(v) \cap C_j = N(w) \cap C_j = C_j$
- the following bipartite graph $\mathcal{B}$ is connected: $V(\mathcal{B}) = B_1 \cup B_2$, $B_1 = \{b_1^1, \ldots, b_k^1\}$, $B_2 = \{b_1^2, \ldots, b_{k-1}^2\}$ and $b_j^1 b_r^2 \in E(\mathcal{B})$ iff X_j and C_r are adjacent.

The objective is the following:

- if G contains an independent set S such that $S \cap X_i \neq \emptyset$ for all $i \in \{1, \ldots, k\}$, then the algorithm must answer "YES". In that case the solution is called a *rainbow independent set.*
- if G does not contain an independent set of size k, then the algorithm must answer "NO".

Observe that in the case the graph contains an independent set of size k but no rainbow independent set, the algorithm is allowed to answer either *yes* or *no.* Eventually, this will imply a one-sided error Monte-Carlo algorithm with constant error probability for MIS. Definition 7 is illustrated by Figure 6.

Lemma 3 *Let $\mathcal{G}$ be a hereditary graph class. If there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ such that f-Ramsey-extracted Iterative Expansion MIS is FPT in $\mathcal{G}$, then g-Iterative Expansion MIS is FPT in $\mathcal{G}$, where $g(x) = Ram_{\ell_x}(f(x)2^{x(x-1)})\ \forall x \in \mathbb{N}$, with $\ell_x = 2^{(x-1)^2}$.*

Proof. Let $f : \mathbb{N} \to \mathbb{N}$ be such a function, and let G, k and $\mathcal{S} = \{S_1, \ldots, S_{g(k)}\}$ be an input of g-Iterative Expansion MIS. Recall that the objective is to find an independent set of size k in G, or to decide that $\alpha(G) < k$. We prove it by induction on k. If G contains an independent set of size k, then either there is one intersecting some set of $\mathcal{S}$, or every independent set of size k avoids the sets in $\mathcal{S}$. In order to capture the first case, we branch on every vertex v of the sets in $\mathcal{S}$, and make a recursive call with parameter $G \setminus N[v]$, $k-1$. In the remainder of the algorithm, we thus assume that any independent set of size k in G avoids every set of $\mathcal{S}$.

[7] A set of vertices M is a module if every vertex $v \notin M$ is adjacent to either all vertices of M, or none.

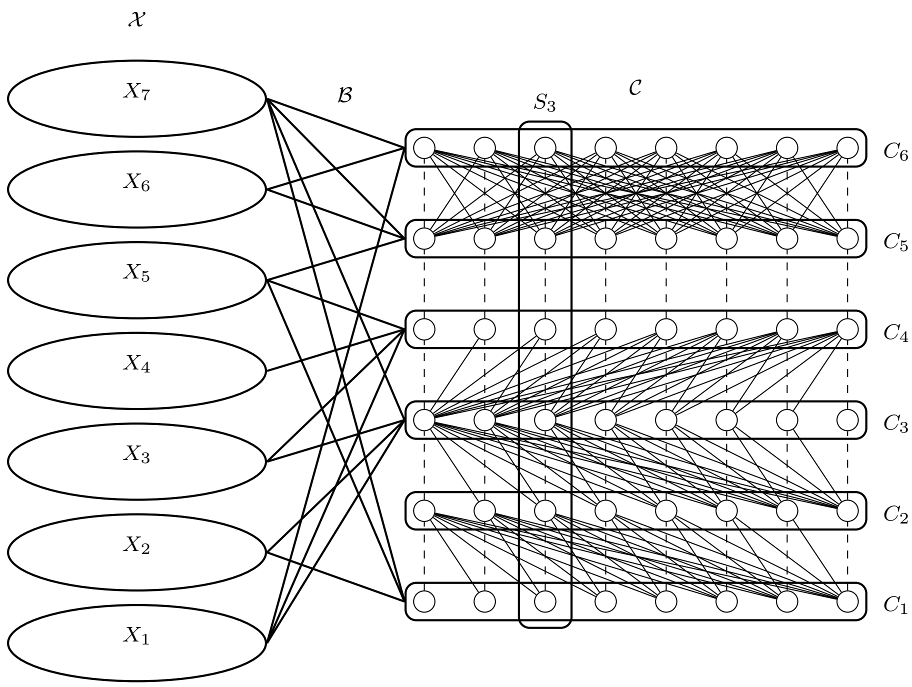


Fig. 6 The structure of the f-Ramsey-extracted Iterative Expansion MIS inputs.

We choose an arbitrary ordering of the vertices of each S_j. Let us denote by s_j^r the r^{th} vertex of S_j. Notice that given an ordered pair of sets of $k-1$ vertices (A, B), there are $\ell_k = 2^{(k-1)^2}$ possible sets of edges between these two sets. Let us denote by $c_1, \ldots, c_{2^{(k-1)^2}}$ the possible sets of edges, called *types*. We define an auxiliary edge-colored graph H whose vertices are in one-to-one correspondence with $S_1, \ldots, S_{g(k)}$, and, for $i < j$, there is an edge between S_i and S_j of color γ iff the type of (S_i, S_j) is γ. By Ramsey's theorem, since H has $Ram_{\ell_k}(f(k)2^{k(k-1)})$ vertices, it must admit a monochromatic clique of size at least $h(k) = f(k)2^{k(k-1)}$. *W.l.o.g.*, the vertex set of this clique corresponds to $S_1, \ldots, S_{h(k)}$. For $p \in \{1, \ldots, k-1\}$, let $C_p = \{s_1^p, \ldots, s_{h(k)}^p\}$. Observe that the Ramsey extraction ensures that each C_p is either a clique or an independent set. If C_p is an independent set for some p, then we can immediately conclude, since $h(k) \geqslant k$. Hence, we suppose that C_p is a clique for every $p \in \{1, \ldots, k-1\}$. We now prove that $C_1, \ldots, C_{k-1}$ are Ramsey-extracted cliques of size $h(k)$. First, by construction, for every $j \in \{1, \ldots, h(k)\}$, the set $\{s_j^p : p = 1, \ldots, k-1\}$ is an independent set. Then, let c be the type of the monochromatic clique of H obtained previously, represented by the adjacencies between two sets (A, B), each of size $k-1$. For every $p \in \{1, \ldots, k-1\}$, let a_p (resp. b_p) be the p^{th} vertex of A (resp. B). Let $p, q \in \{1, \ldots, k-1\}$, $p \neq q$. If none of $a_p b_q$ and $a_q b_p$ are edges in type c, then there is no edge between C_p and C_q, and their relation is thus empty. If both edges $a_p b_q$ and $a_q b_p$ exist in c, then the relation between C_p and C_q is full. Finally if exactly one edge among $a_p b_q$ and $a_q b_p$ exists in c, then the relation between C_p and C_q is semi-full. This concludes the fact that $\mathcal{C} = \{C_1, \ldots, C_{k-1}\}$ are Ramsey-extracted cliques of size $h(k)$.

Suppose that G has an independent set $X^* = \{x_1^*, \ldots, x_k^*\}$. Recall that we assumed previously that X^* is contained in $V(G) \setminus (C_1 \cup \cdots \cup C_{k-1})$. The next step of the algorithm consists of branching on every subset of $f(k)$ indices $J \subseteq \{1, \ldots, h(k)\}$, and restrict every set C_p to $\{s_j^p : j \in J\}$. For the sake of readability, we keep the notation C_p to denote $\{s_j^p : j \in J\}$ (the non-selected vertices are put back in the set of remaining vertices of the graph, *i.e.* we do not delete them). Since $h(k) = f(k)2^{k(k-1)}$, there must exist a branch where the chosen indices are such that for every $i \in \{1, \ldots, k\}$ and every $p \in \{1, \ldots, k-1\}$, x_i^* is either adjacent to all vertices of C_p or none of them. In the remainder, we may thus assume that such a branch has been made, with respect to the considered solution $X^* = \{x_1^*, \ldots, x_k^*\}$. Now, for every $v \in V(G) \setminus (C_1, \ldots, C_{k-1})$, if there exists $p \in \{1, \ldots, k-1\}$ such that $N(v) \cap C_p \neq \emptyset$ and $N(v) \cap C_p \neq C_p$, then we can remove this vertex, as we know that it cannot correspond to any x_i^*. Thus, we know that all the remaining vertices v are such that for every $p \in \{1, \ldots, k-1\}$, v is either adjacent to all vertices of C_p, or none of them.

In the following, we perform a color coding-based step on the remaining vertices. Informally, this color coding will allow us to identify, for every vertex x_i^* of the optimal solution, a set X_i of candidates, with the property that all vertices in X_i have the same neighborhood with respect to sets $C_1, \ldots, C_{k-1}$. We thus color uniformly at random the remaining vertices $V(G) \setminus (C_1, \ldots, C_{k-1})$ using k colors. The probability that the elements of X^* are colored with pairwise distinct colors is at least e^{-k}.

This random process can be derandomized using the so-called notion of perfect hash families. A (n, k)-*perfect hash family* is a family of functions $\mathcal{F}$ from $[n]$ to $[k]$ (which can be seen as colorings) such that for every set $S \in \binom{[n]}{k}$, there exists $f \in \mathcal{F}$ such that the restriction of f on S is injective. It is known [23] that a (n, k)-perfect hash family of size $e^k k^{O(\log k)} \log n$ can be constructed in time $e^k k^{O(\log k)} n \log n$. Hence, instead of coloring $V(G) \setminus (C_1, \ldots, C_{k-1})$ uniformly at random, we branch on every coloring $f \in \mathcal{F}$ and run the remainder of the algorithm. The definition of (n, k)-perfect hash family ensures that there is a coloring f such that X^* is a rainbow independent set with respect to f. Notice that this derandomization step implies a branching into $h(k) \log n$ subcases, for some computable function h. However, the depth of the branching tree (*i.e.* the maximum number of times this branching will be made in every computation path) is bounded by a function of k only. Since $(\log n)^k \leq g(k)n$ for some function g [26], the deterministic version of the algorithm is still FPT.

We are thus reduced to the case of finding a rainbow independent set. For every $i \in \{1, \ldots, k\}$, let X_i be the vertices of $V(G) \setminus (C_1, \ldots, C_{k-1})$ colored with color i. We now partition every set X_i into at most 2^{k-1} subsets $X_i^1, \ldots, X_i^{2^{k-1}}$, such that for every $j \in \{1, \ldots, 2^{k-1}\}$, all vertices of X_i^j have the same neighborhood with respect to the sets $C_1, \ldots, C_{k-1}$ (recall that every vertex of $V(G) \setminus (C_1, \ldots, C_{k-1})$ is adjacent to all vertices of C_p or none, for each $p \in \{1, \ldots, k-1\}$). We branch on every tuple $(j_1, \ldots, j_k) \in \{1, \ldots, 2^{k-1}\}$. Clearly the number of branches is bounded by a function of k only and, moreover, one branch $(j_1, \ldots, j_k)$ is such that x_i^* has the same neighborhood in $C_1 \cup \cdots \cup C_{k-1}$ as vertices of $X_i^{j_i}$ for every $i \in \{1, \ldots, k\}$. We assume in the following that such a branching has been made. For every $i \in \{1, \ldots, k\}$, we can thus remove vertices of X_i^j for every $j \neq j_i$. For the sake of readability, we rename $X_i^{j_i}$ as X_i. Let $\mathcal{B}$ be the bipartite graph with vertex bipartition (B_1, B_2), $B_1 = \{b_1^1, \ldots, b_k^1\}$, $B_2 = \{b_1^2, \ldots, b_{k-1}^2\}$, and $b_i^1 b_p^2 \in E(\mathcal{B})$ iff x_i^* is adjacent to C_p. Since every x_i^* has the same neighborhood as X_i with respect to $C_1, \ldots, C_{k-1}$, this bipartite graph actually corresponds to the one described in Definition 7 representing the adjacencies between X_i's and C_p's. We now prove that it is connected. Suppose it is not. Then, since $|B_1| = k$ and $|B_2| = k-1$, there must be a component with as many vertices from B_1 as vertices from B_2. However, in this case, using the fixed solution X^* on one side and an independent set of size $k-1$ in $C_1 \cup \cdots \cup C_{k-1}$ on the other side, it implies that there is an independent set of size k intersecting $\cup_{p=1}^{k-1} C_p$, a contradiction.

Hence, all conditions of Definition 7 are now fulfilled. It now remains to find an independent set of size k disjoint from the sets $\mathcal{C}$, and having a non-empty intersection with X_i, for every $i \in \{1, \ldots, k\}$. We thus run an algorithm solving f-Ramsey-extracted Iterative Expansion MIS on this input, which concludes the algorithm. □

The proof of the following result is immediate, by using successively Lemmas 2 and 3.

Theorem 7 *Let $\mathcal{G}$ be a hereditary graph class. If f-*Ramsey-extracted Iterative Expansion MIS *is FPT in $\mathcal{G}$ for some computable function f, then* MIS *is FPT in $\mathcal{G}$.*

We now apply this framework to two families of graphs H.

4.2 Clique minus a smaller clique

Theorem 8 *For any $r \geqslant 2$ and $2 \leqslant s < r$,* MIS *in $(K_r \setminus K_s)$-free graphs is FPT if $s \leqslant 3$, and $W[1]$-hard otherwise.*

Proof. The case $s = 2$ was already known [12]. The result for $s \geqslant 4$ comes from Theorem 2. We now deal with the case $s = 3$. We solve the problem in $(K_{r+3} \setminus K_3)$-free graphs, for every $r \geqslant 2$ (the problem is polynomial for $r = 1$, since it it corresponds exactly to the case of claw-free graphs). Let G, k be an input of the problem. We present an FPT algorithm for f-Ramsey-extracted Iterative Expansion MIS with $f(x) = r$ for every $x \in \mathbb{N}$. The result for MIS can then be obtained using Theorem 7.

We thus assume that $V(G) = X_1 \cup \cdots \cup X_k \cup C_1 \cup \cdots \cup C_{k-1}$ where all cliques C_p have size r. Consider the bipartite graph $\mathcal{B}$ representing the adjacencies between $\{X_1, \ldots, X_k\}$ and $\{C_1, \ldots, C_{k-1}\}$, as in Definition 7 (for the sake of readability, we will make no distinction between the vertices of $\mathcal{B}$ and the sets $\{X_1, \ldots, X_k\}$ and $\{C_1, \ldots, C_{k-1}\}$). We may first assume that $|X_i| \geqslant Ram(r, k)$ for every $i \in \{1, \ldots, k\}$, since otherwise we can branch on every vertex v of X_i and make a recursive call with input $G \setminus N[v]$, $k-1$. Hence, for every $i \in \{1, \ldots, k\}$, we may assume that X_i contains a clique on r vertices (indeed, if it does not, then it must contain an independent set of size k, in which case we are done). Suppose now that G contains an independent set $S^* = \{x_1^*, \ldots, x_k^*\}$, with $x_i \in X_i$ for all $i \in \{1, \ldots, k\}$. The first step is to consider the structure of $\mathcal{B}$, using the fact that G is $(K_r \setminus K_3)$-free. We have the following:

Claim 9. *$\mathcal{B}$ is a path, or we can conclude in polynomial time.*

Proof of claim: We first prove that for every $i \in \{1, \ldots, k\}$, the degree of X_i in $\mathcal{B}$ is at most 2. Indeed, assume by contradiction that it is adjacent to C_a, C_b and C_c. Since $|X_i| \geqslant Ram(r, k)$, by Ramsey's theorem, it either contains an independent set of size k, in which case we are done, or a clique K of size r. However, observe in this case that K together with s_1^a, s_1^b and s_1^c (which are pairwise non-adjacent) induces a graph isomorphic to $K_{r+3} \setminus K_3$.

Then, we show that for every $i \in \{1, \ldots, k-1\}$, the degree of C_i in $\mathcal{B}$ is at most 2. Assume by contradiction that C_i is adjacent to X_a, X_b and X_c. If the instance is positive, then there must be an independent set of size three with non-empty intersection with each of X_a, X_b and X_c. If such an independent set does not exist (which can be checked in cubic time), we can immediately answer NO. Now observe that C_i (which is of size r) together with this independent set induces a graph isomorphic to $K_{r+3} \setminus K_3$.

To summarize, $\mathcal{B}$ is a connected bipartite graph of maximum degree 2 with k vertices in one part, $k-1$ vertices in the other part. It must be a path. $\triangleleft$

W.l.o.g., we may assume that for every $i \in \{2, \ldots, k-1\}$, X_i is adjacent to C_{i-1} and C_i, and that X_1 (resp. X_k) is adjacent to C_1 (resp. C_{k-1}). We now concentrate on the adjacencies between sets X_i. We say that an edge $xy \in E(G)$ is a *long edge* if $x \in X_i$, $y \in X_j$ with $|j-i| \geqslant 2$ and $2 \leqslant i, j \leqslant k-1$, $i \neq j$.

Claim 10. *$\forall x \in X_2 \cup \cdots \cup X_{k-1}$, x is incident to at most $(k-2)(Ram(r,3)-1)$ long edges.*

Proof of claim: In order to prove it, let us show that for $i, j \in \{2, \ldots, k-1\}$ such that $|j-i| \geqslant 2$, $i \neq j$, and for every $x \in X_i$, $|N(x) \cap X_j| \leqslant Ram(r,3)-1$. Assume by contradiction that there exists $x \in X_i$ which has at least $Ram(r,3)$ neighbors $Y \subseteq X_j$. By Ramsey's theorem, either Y contains an independent set of size 3 or a clique of size r. In the first case, C_j together with these three vertices induces a graph isomorphic to $K_{r+3} \setminus K_3$. Hence we may assume that Y contains a clique Y' of size r. But in this case, Y' together with x, s_1^{j-1}, s_1^j induce a graph isomorphic to $K_{r+3} \setminus K_3$ as well. $\triangleleft$

Recall that the objective is to find an independent set of size k with non-empty intersection with X_i, for every $i \in \{1, \ldots, k\}$. We assume $k \geqslant 5$, otherwise the problem is polynomial. The algorithm starts by branching on every pair of non-adjacent vertices $(x_1, x_k) \in X_1 \times X_k$, and removing the union of their neihborhoods in $X_2 \cup \cdots \cup X_{k-1}$. For the sake of readability, we still denote by $X_2, \ldots, X_{k-1}$ these reduced sets. If such a pair does not exist or the removal of their neighborhood empties some X_i, then we immediately answer NO (for this branch). Informally speaking, we just guessed the solution within X_1 and X_k (the reason for this is that we cannot bound the number of long edges incident to vertices of these sets). We now concentrate on the graph G', which is the graph induced by $X_2 \cup \cdots \cup X_{k-1}$. Clearly, it remains to decide whether G' admits an independent set of size $k-2$ with non-empty intersection with X_i, for every $i \in \{2, \ldots, k-1\}$.

The previous claim showed that the structure of G' is quite particular: roughly speaking, the adjacencies between consecutive X_i's is arbitrary, but the number of long edges is bounded for every vertex. The key observation is that if there were no long edge at all, then a simple dynamic programming algorithm would allow us to conclude. Nevertheless, using the previous claim, we can actually upper bound the number of long edges incident to a vertex of the solution by a function of k only (recall that r is a constant). We can then get rid of these problematic long edges using the so-called technique of *random separation* [6]. Let $S = \{x_2, \ldots, x_{k-1}\}$ be a solution of our problem (with $x_i \in X_i$ for every $i \in \{2, \ldots, k-1\}$). Let us define $D = \{y : xy$ is a long edge and $x \in S\}$. By the previous claim, we have $|D| \leqslant (Ram(r,3)-1)(k-2)^2$. The idea of random separation is to delete each vertex of the graph with probability $\frac{1}{2}$. At the end, we say that a removal is *successful* if both of the two following conditions hold: (i) no vertex of S has been removed, and (ii) all vertices of D have been removed (other vertices but S may have also been removed). Observe that the probability that a removal is successful is at least $2^{-k^2 Ram(r,3)}$. In such a case, we can remove all remaining long edges (more formally, we remove their endpoints): indeed, for a remaining long edge xy, we know that there exists a solution avoiding both x and y, hence we can safely delete x and y.

Similarly to the color coding step of Lemma 3, this can be derandomized using (n,t)-universal sets: a (n,t)-*universal set* is a family $\mathcal{U}$ of subsets of $[n]$ such that for any $S \subseteq [n]$ of size t, the family $\{A \cap S : A \in \mathcal{U}\}$ contains all 2^t subsets of S. It is known [23] that for any $n, t \geqslant 1$, one can construct an (n,t)-universal set of size $2^t t^{O(\log t)} \log n$ in time $2^t t^{O(\log t)} n \log n$. Let $\mathcal{U}$ be an (n,t)-universal set for $t = k + (Ram(r,3)-1)(k-2)^2$. Instead of deleting vertices of G randomly, branch on every set $U \in \mathcal{U}$, and for each branch, delete vertices from U. Then there must be a branch where $D \subseteq U$ and $S \nsubseteq U$, hence vertices of D are deleted while those of S are not. As previously, this implies branching into $h(k) \log n$ subcases for some computable function of k, but since the depth of the branching tree is a function of k only, the running time of the deterministic version is still FPT.

We still denote by $X_2, \ldots, X_{k-1}$ the reduced sets, for the sake of readability. We thus end up with a graph composed of sets $X_2, \ldots, X_{k-1}$, with edges between X_i and X_j only if $[j-i| = 1$. In that case, observe that there is a solution if and only if the following dynamic programming returns *true* on input $P(3, x_2)$ for some $x_2 \in X_2$:

$$P(i, x_{i-1}) = \begin{cases} true & \text{if } i = k \\ false & \text{if } X_i \subseteq N(x_{i-1}) \\ \bigvee_{x_i \in X_i \setminus N(x_{i-1})} P(i+1, x_i) & \text{otherwise.} \end{cases}$$

Informally, this dynamic programming relies on the fact that the only adjacencies between sets X_i are between consecutive sets, hence we only need to remember the previous choice when constructing a solution from $i = 2$ to $k-1$. Hence, $P(i, x_{i-1})$ represents whether there exists a rainbow solution in $\cup_{j=i-1}^{k-1} X_j$ containing $x_{i-1} \in X_{i-1}$. Clearly this dynamic programming runs in $O(mnk)$ time, where m and n are the number of edges and vertices of the remaining graph, respectively. Moreover, it can easily be turned into an algorithm returning a solution of size $k-2$ if it exists.

□

4.3 Clique minus a complete bipartite graph

For every three positive integers r, s_1, s_2 with $s_1 + s_2 < r$, we consider the graph $K_r \setminus K_{s_1,s_2}$. Another way to see $K_r \setminus K_{s_1,s_2}$ is as a P_3 *of cliques* of size s_1, $r - s_1 - s_2$, and s_2. More formally, every graph $K_r \setminus K_{s_1,s_2}$ can be obtained from a P_3 by adding $s_1 - 1$ false twins of the first vertex, $r - s_1 - s_2 - 1$, for the second, and $s_2 - 1$, for the third.

Theorem 11 *For any $r \geqslant 2$ and $s_1 \leqslant s_2$ with $s_1 + s_2 < r$,* MIS *in $K_r \setminus K_{s_1,s_2}$-free graphs is FPT.*

Proof. It is more convenient to prove the result for $K_{3r} \setminus K_{r,r}$-free graphs, for any positive integer r. It implies the theorem by choosing this new r to be larger than s_1, s_2, and $r - s_1 - s_2$. We will show that for $f(x) := 3r$ for every $x \in \mathbb{N}$, f-Ramsey-extracted Iterative Expansion MIS in $K_{3r} \setminus K_{r,r}$-free graphs is FPT. By Theorem 7, this implies that MIS is FPT in this class. Let $C_1, \ldots, C_{k-1}$ (whose union is denoted by $\mathcal{C}$) be the Ramsey-extracted cliques of size $3r$, which can be partitioned, as in Definition 7, into $3r$ independent sets $S_1, \ldots, S_{3r}$, each of size $k-1$. Let $\mathcal{X} = \bigcup_{i=1}^{k} X_i$ be the set in which we are looking for an independent set of size k. We recall that between any X_i and any C_j there are either all the edges or none. Hence, the whole interaction between $\mathcal{X}$ and $\mathcal{C}$ can be described by the bipartite graph $\mathcal{B}$ described in Definition 7. Firstly, we can assume that each X_i is of size at least $Ram(r, k)$, otherwise we can branch on $Ram(r, k)$ choices to find one vertex in an optimum solution (and decrease k by one). By Ramsey's theorem, we can assume that each X_i contains a clique of size r (if it contains an independent set of size k, we are done). Our general strategy is to leverage the fact that the input graph is $(K_{3r} \setminus K_{r,r})$-free to describe the structure of $\mathcal{X}$. Hopefully, this structure will be sufficient to solve our problem in FPT time.

We define an auxiliary graph Y with $k-1$ vertices. The vertices $y_1, \ldots, y_{k-1}$ of Y represent the Ramsey-extracted cliques of $\mathcal{C}$ and two vertices y_i and y_j are adjacent iff the relation between C_i and C_j is not empty (equivalently the relation is full or semi-full). It might seem peculiar that we concentrate the structure of $\mathcal{C}$, when we will eventually discard it from the graph. It is an indirect move: the simple structure of $\mathcal{C}$ will imply that the interaction between $\mathcal{X}$ and $\mathcal{C}$ is simple, which in turn, will severely restrict the subgraph induced by $\mathcal{X}$. More concretely, in the rest of the proof, we will (1) show that Y is a clique, (2) deduce that $\mathcal{B}$ is a complete bipartite graph, (3) conclude that $\mathcal{X}$ cannot contain an induced $K_r^2 = K_r \uplus K_r$ and run the algorithm of Theorem 6 (which is even stronger than simply solving the colored version of the problem: Theorem 6 returns YES if and only if the instance contains an independent set of size k).

Suppose that there is $y_{i_1}y_{i_2}y_{i_3}$ an induced P_3 in Y, and consider C_{i_1}, C_{i_2}, C_{i_3} the corresponding Ramsey-extracted cliques. For $s < t \in [3r]$, let $C_i^{s \to t} := C_i \cap \bigcup_{s \leqslant j \leqslant t} S_j$. In other words, $C_i^{s \to t}$ contains the elements of C_i having indices between s and t. Since $|C_i| = 3r$, each C_i can be partitioned into three sets, of r elements each: $C_i^{1 \to r}$, $C_i^{r+1 \to 2r}$ and $C_i^{2r+1 \to 3r}$. Recall that the relation between C_{i_1} and C_{i_2} (resp. C_{i_2} and C_{i_3}) is either full or semi-full, while the relation between C_{i_1} and C_{i_3} is empty. This implies that at least one of the four following sets induces a graph isomorphic to $K_{3r} \setminus K_{r,r}$:

- $C_{i_1}^{1 \to r} \cup C_{i_2}^{r+1 \to 2r} \cup C_{i_3}^{1 \to r}$
- $C_{i_1}^{1 \to r} \cup C_{i_2}^{r+1 \to 2r} \cup C_{i_3}^{2r+1 \to 3r}$
- $C_{i_1}^{2r+1 \to 3r} \cup C_{i_2}^{r+1 \to 2r} \cup C_{i_3}^{1 \to r}$
- $C_{i_1}^{2r+1 \to 3r} \cup C_{i_2}^{r+1 \to 2r} \cup C_{i_3}^{2r+1 \to 3r}$

Hence, Y is a disjoint union of cliques (since it is P_3-free). Let us assume that Y is the union of at least two (maximal) cliques.

Recall that the bipartite graph $\mathcal{B}$ is connected. Thus there is $b_h^1 \in B_1$ (corresponding to X_h) adjacent to $b_i^2 \in B_2$ and $b_j^2 \in B_2$ (corresponding to C_i and C_j, respectively), such that y_i and y_j lie in two different connected components of Y (in particular, the relation between C_i and C_j is empty). Recall that X_h contains a clique of size at least r. This clique induces, together with any r vertices in C_i and any r vertices in C_j, a graph isomorphic to $K_{3r} \setminus K_{r,r}$; a contradiction. Hence, Y is a clique.

Now, we can show that $\mathcal{B}$ is a complete bipartite graph. Each X_h has to be adjacent to at least one C_i (otherwise this trivially contradicts the connectedness of $\mathcal{B}$). If X_h is not linked to C_j for some $j \in \{1, \ldots, k-1\}$, then a clique of size r in X_h (which always exists) induces, together with $C_i^{1 \to r} \cup C_j^{2r+1 \to 3r}$ or with $C_i^{2r+1 \to 3r} \cup C_j^{1 \to r}$, a graph isomorphic to $K_{3r} \setminus K_{r,r}$.

Since $\mathcal{B}$ is a complete bipartite graph, every vertex of C_1 dominates all vertices of $\mathcal{X}$ In particular, $\mathcal{X}$ is in the intersection of the neighborhood of the vertices of some clique of size r. This implies that the subgraph induced by $\mathcal{X}$

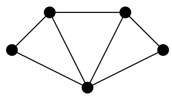


Fig. 7 The gem.

is $(K_r \uplus K_r)$-free. Hence, we can run the FPT algorithm of Theorem 6 on this graph. □

4.4 The gem

Let the *gem* be the graph obtained by adding a universal vertex to a path on four vertices (see Figure 7). Using our framework once again, we are able to obtain the following result:

Theorem 12 *There is an FPT algorithm for* MIS *in gem-free graphs.*

Proof. Let $f(x) := 1$ for every $x \in \mathbb{N}$. We prove that f-Ramsey-extracted Iterative Expansion MIS admits an FPT algorithm in *gem*-free graphs. By the definition of f, we have $C_p = \{c_p\}$ for every $p \in \{1, \dots, k-1\}$. Recall that the objective is to find a rainbow independent set in G, or to decide that $\alpha(G) < k$. Since the bipartite graph $\mathcal{B}$ representing the adjacencies between $\{X_1, \dots, X_k\}$ and $\{c_1, \dots, c_{k-1}\}$ is connected, it implies that for every $i \in \{1, \dots, k\}$, there exists $p \in \{1, \dots, k-1\}$ such that c_p dominates all vertices of X_i. Since G is *gem*-free, it implies that $G[X_i]$ is P_4-free for every $i \in \{1, \dots, k\}$. Since P_4-free graphs (*a.k.a* cographs) are perfect, the size of a maximum independent set equals the size of a clique cover. If $G[X_i]$ contains an independent set of size k (which can be tested in polynomial time), then we are done. Otherwise, we can, still in polynomial time, partition the vertices of X_i into at most $k-1$ sets $X_i^1, \dots, X_i^{q_i}$, where $G[X_i^j]$ induces a clique for every $j \in \{1, \dots, q_i\}$. We now perform a branching for every tuple $(j_1, \dots, j_k)$, where $j_i \in \{1, \dots, q_i\}$ for every $i \in \{1, \dots, k\}$, which, informally, allows us to guess the clique $X_i^{j_i}$ which contains the element of the rainbow independent set we are looking for. For the sake of readability, we allow ourselves this slight abuse of notation: we rename $X_i^{j_i}$ into simply X_i. Thus, for every $i \in \{1, \dots, k\}$, $G[X_i]$ is a clique.

Now, let $i, j \in \{1, \dots, k\}$, $i \neq j$. Let us analyse the adjacencies between X_i and X_j. We say that $\{a, b, c, d\} \subseteq X_i \cup X_j$ is a *balanced diamond* if $a, b \in X_i$ $(a \neq b)$, $c, d \in X_j$ $(c \neq d)$ and all vertices $\{a, b, c, d\}$ are pairwise adjacent but $\{b, d\}$. We have the following claim:

Claim 13. *If the graph induced by $X_i \cup X_j$ has a balanced diamond, then X_i and X_j are twins in $\mathcal{B}$.*

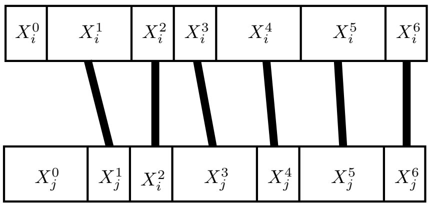


Fig. 8 Schema of the adjacencies between X_i and X_i when they do not contain a balanced diamond ($q = 6$). An edge represent a complete relation between the corresponding subsets.

Proof of claim: Suppose they are not. *W.l.o.g.* we assume that X_i is adjacent to $\{c_p\}$ while X_j is not, for some $p \in \{1, \ldots, k-1\}$. Then the vertices of the balanced diamond together with c_p induce a *gem*. ◁

The remainder of the proof consists of "cleaning" the adjacencies (X_i, X_j) having no balanced diamond (but at least one edge between them). In that case, observe that X_i and X_j can respectively be partitioned into X_i^0, X_i^1, $\ldots$, X_i^q and X_j^0, X_j^1, $\ldots$, X_j^q (where X_i^0 and X_j^0 are potentially empty) such that $X_i^r \cup X_j^r$ induces a clique for every $r \in \{1, \ldots, q\}$, and there is no edge between X_i^r and $X_j^{r'}$ whenever $r \neq r'$ or $r = 0$ or $r' = 0$ (see Figure 8). In each branch of the next branching rule, the sets $\{X_1, \ldots, X_k\}$ will be modified into $\{X'_1, \ldots, X'_k\}$. For the sake of readability, we chose to state the rule as a random one, and then explain how to derandomize it.

Branching rule: *Let $i, j \in \{1, \ldots, k\}$, $i \neq j$ such that $X_i \cup X_j$ has no balanced diamond. Then perform the following branching:*

- *Branch 1: $X'_i = X_i^0$ and $X'_z = X_z$ for $z \in [k] \setminus \{i\}$*
- *Branch 2: $X'_j = X_j^0$ and $X'_z = X_z$ for $z \in [k] \setminus \{j\}$*
- *Branch 3: pick a set $T \subseteq \{1, \ldots, q\}$ uniformly at random, then:*
  - $X'_i = \bigcup_{r \in T} X_i^r$
  - $X'_j = \bigcup_{r \notin T} X_j^r$
  - $X'_z = X_z$ *for* $z \in [k] \setminus \{i, j\}$

Consider the graph $\mathcal{G}(X_1, \ldots, X_k)$ having one vertex per set X_i, and an edge between X_i and X_j if these two sets are adjacent. We now prove the following:

Claim 14. *The graph $\mathcal{G}(X'_1, \ldots, X'_k)$ has one edge less than $\mathcal{G}(X_1, \ldots, X_k)$*

Proof of claim: In all three branches, observe that there is no edge between X'_i and X'_j. ◁

Claim 15. *If G has no independent set of size k, then no graph obtained after the branching contains an independent set of size k.*

Proof of claim: Observe that in all branches, $\bigcup_{z=1}^{k} X'_z \subseteq \bigcup_{z=1}^{k} X_z$, that is, each graph obtained in each branch is an induced subgraph of G. ◁

Claim 16. *If G has a rainbow independent set, then with probability at least $\frac{1}{2}$, at least one branch leads to a graph having a rainbow independent set.*

Proof of claim: Suppose that G contains a rainbow independent set S^*. If S^* intersects X_i^0, then S^* also exists in the graph of the first branch. If S^* intersects X_j^0, then S^* also exists in the graph of the second branch. The last case is where S^* intersects $X_i^{r_1}$ and $X_j^{r_2}$, for some $r_1, r_2 \in \{1, \ldots q\}$. In that case, there is a probability of $\frac{1}{2}$ that $r_1 \in T$ and $r_2 \notin T$, which concludes the proof of the claim $\lhd$

The derandomization of this branching rule uses uses once again (n,t)-universal sets. However, this case is simpler since we actually need a $(q,2)$-universal set, which can be easily constructed as follows. For every $i \in \{1, \ldots, \lceil \log q \rceil\}$, define T_i to be the set of all integers $r \leq q$ whose binary representation contains a one at the i^{th} bit. Then let $\mathcal{U} = \{T_i, i = 1 ... \lceil \log q \rceil\}$. This family is of size $\lceil \log n \rceil$ and can be constructed in $O(n \log n)$ time. The deterministic version of the previous branching rule contains the same first two branches, and replaces the random third one by $|\mathcal{U}|$ branches, where, instead of picking $T \subseteq \{1, \ldots, q\}$ at random, we branch on every $T \in \mathcal{U}$. Now, Claims 14 and 15 remain the same, while Claim 16 can be replaced by the fact that if G has a rainbow independent set, then at least one branch leads to a graph having a rainbow independent set. Its correctness follows from the fact that by construction of $\mathcal{U}$, for every $r_1, r_2 \in \{1, \ldots, q\}$, $r_1 \neq r_2$, there exists $T \in \mathcal{U}$ such that $r_1 \in T$ and $r_2 \notin T$. As in Lemma 3, this implies branching into $O(\log n)$ subcases, but since the depth of the branching tree is a function of k only, the running time of the deterministic version is still FPT.

We apply the previous branching rule exhaustively, hence we now assume it cannot apply. For the sake of readability, we keep the notation $X_1, \ldots, X_k$ in order to denote our instance, even after an eventual application of the previous branching rule. For every X_i, X_j with $i \neq j$, there is either (i) no edge between X_i and X_j, or (ii) a balanced diamond induced by $X_i \cup X_j$. Hence, Claim 13 implies that each connected component of the graph induced by $\bigcup_{i=1}^{k} X_i$ is a module with respect to the clique $\{c_1, \ldots, c_{k-1}\}$. In particular, each connected component is dominated by some c_p, with $p \in \{1, \ldots, k-1\}$, and is thus P_4-free (otherwise, a P_4 together with this vertex c_p induce a gem), which means that we can decide in polynomial time whether G contains an independent set of size k, by deciding the problem in every connected component separately (since MIS is polynomial-time solvable in P_4-free graphs). This concludes the proof, since by Claim 14, the previous branching rule can be applied at most $\binom{k}{2}$ times. $\square$

5 Polynomial (Turing) kernels

In this section we investigate some special cases of Section 4.3, in particular when H is a clique of size r minus a claw with s branches, for $s < r$. Although Theorem 11 proves that MIS is FPT for every possible values of r and s, we show that when $s \geqslant r-2$, the problem admits a polynomial Turing kernel, while for $s \leqslant 2$, it admits a polynomial kernel. Notice that the latter result is somehow tight, as Corollary 4 shows that MIS cannot admit a polynomial kernel in $(K_r \setminus K_{1,s})$-free graphs whenever $s \geqslant 3$.

5.1 Positive results

The main ingredient of the two following results is a constructive version of the Erdős-Hajnal theorem for the concerned graph classes:

Lemma 4 (Constructive Erdős-Hajnal for $K_r \setminus K_{1,s}$) *For every $r \geqslant 2$ and $s < r$, there exists a polynomial-time algorithm which takes as input a connected $(K_r \setminus K_{1,s})$-free graph G, and constructs either a clique or an independent set of size $n^{\frac{1}{r-1}}$, where n is the number of vertices of G.*

Proof. First consider the case $s = r-1$, *i.e.* the forbidden graph is K_{r-1} plus an isolated vertex. If G contains a vertex v with non-neighborhood N of size at least $n^{\frac{r-2}{r-1}}$, then, since $G[N]$ is K_{r-1}-free, by Ramsey's theorem, it must contains an independent set of size $|N|^{\frac{1}{r-2}} = n^{\frac{1}{r-1}}$, which can be found in polynomial time. We may now assume that the maximum non-degree[8] of G is $n^{\frac{r-2}{r-1}} - 1$. We construct a clique $v_1, \ldots, v_q$ in G by picking an arbitrary vertex v_1, removing its non-neighborhood, then picking another vertex v_2, removing its non-neighborhood, and repeating this process until the graph becomes empty. Using the above argument on the maximum non-degree, this process can be applied $\frac{n}{n^{\frac{r-2}{r-1}}} = n^{\frac{1}{r-1}}$ times, corresponding to the size of the constructed clique.

Now, we make an induction on $r-1-s$ (the base case is above). If G contains a vertex v with neighborhood N of size at least $n^{\frac{r-2}{r-1}}$, then, since $G[N]$ is $(K_{r-1} \setminus K_s)$-free, by induction it admits either a clique or an independent set of size $|N|^{\frac{1}{r-2}} = n^{\frac{1}{r-1}}$, which can be found in polynomial time. We may now assume that the maximum degree of G is $n^{\frac{r-2}{r-1}} - 1$. We construct an independent set $v_1, \ldots, v_q$ in G by picking an arbitrary vertex v_1, removing its neighborhood, and repeating this process until the graph becomes empty. Using the above argument on the maximum degree, this process can be applied $\frac{n}{n^{\frac{r-2}{r-1}}} = n^{\frac{1}{r-1}}$ times, corresponding to the size of the constructed independent set. □

Theorem 17 *For every $r \geqslant 2$,* MIS *in $(K_r \setminus K_{1,r-2})$-free graphs has a polynomial Turing kernel.*

Proof. The problem is polynomial for $r=2$ and $r=3$, hence we suppose $r \geqslant 4$. Suppose we have an algorithm $\mathcal{A}$ which, given a graph J and an integer i such that $|V(J)| = O(i^{r-1})$, decides whether J has an independent set of

[8] The non-degree of a vertex is the size of its non-neighborhood.

size i in constant time. Having a polynomial algorithm for MIS assuming the existence of $\mathcal{A}$ implies a polynomial Turing kernel for the problem [10]. To do so, we will present an algorithm $\mathcal{B}$ which, given a *connected* graph G and an integer k, outputs a polynomial (in $|V(G)|$) number of instances of size $O(k^{r-1})$, such that one of them is positive iff the former one is. With this algorithm in hand, we obtain the polynomial Turing kernel as follows: let G and k be an instance of MIS. Let $V_1, \ldots, V_\ell$ be the connected components of G. For every $j \in \{1, \ldots, \ell\}$, we determine the size of a maximum independent set k_j of $G[V_j]$ by first invoking, for successive values $i = 1, \ldots, k$, the algorithm $\mathcal{B}$ on input $(G[V_j], i)$, and then $\mathcal{A}$ on each reduced instance. At the end of the algorithm, we answer YES iff $\sum_{j=1}^{\ell} k_i \geqslant k$.

We now describe the algorithm $\mathcal{B}$. Let (G, k) be an input, with $n = |V(G)|$. We first invoke Lemma 4. If the algorithm outputs an independent set of size at least $s = n^{\frac{1}{r-1}}$, then either $k \leqslant s$ and we are done (we output a trivially positive instance), or $k > n^{\frac{1}{r-1}}$ which implies that the instance is a kernel with $O(k^{r-1})$ vertices. Hence, we assume that the algorithm outputs a clique C of size at least $n^{\frac{1}{r-1}}$. We assume that $|C| > r^2$, since otherwise the instance is already reduced.

Let $B = N(C)$. First observe that for every $u \in B$, $|N_C(u)| \geqslant |C| - (r-3)$. Indeed, if $|N_C(u)| \leqslant |C| - (r-2)$, then the graph induced by $r-2$ non-neighbors of u in C together with u and a neighbor of u in C (which exists since $|C| > r^2$) is isomorphic to $K_r \setminus K_{1,r-2}$. Secondly, we claim that $V(G) = C \cup B$: for the sake of contradiction, take $v \in N(B) \setminus C$, and let $u \in B$ be such that $uv \in E(G)$. By the previous argument, u has at least $|C| - r + 3 \geqslant r - 2$ neighbors in C which, in addition to u and v, induce a graph isomorphic to $K_r \setminus K_{1,r-2}$.

The algorithm outputs, for every $u \in B$, the graph induced by $B \setminus N[u]$ (with parameter $k-1$), and, for every $u \in B$ and every $v \in C$ such that $uv \notin E(G)$, the graph induced by $B \setminus (N[u] \cup N[v])$ (with parameter $k-2$). The correctness of the algorithm follows from the fact that if G has an independent set S of size $k > 1$, then either:

- $S \cap C = \emptyset$, in which case $S \setminus \{u\}$ lies entirely in $B \setminus N[u]$ for any $u \in S$, or
- $S \cap C = \{v\}$ for some $v \in C$, in which case $S \setminus \{u, v\}$ lies entirely in $B \setminus (N[u] \cup N[v])$ for any $u \in S \cap B$.

We now argue that each of these instances has $O(k^{r-3})$ vertices. To do so, observe that for any $u \in B$, $B \setminus N[u]$ does not contain K_{r-2} as an induced subgraph: indeed, since $|C| > r^2$, then any set of $r-1$ vertices of B must have a common neighbor in C (since the union of the non-neighborhoods of these $r-1$ vertices in C is of size at most $(r-1)(r-3)$). Now, take (for the sake of contradiction) any clique K of size $r-2$ in $B \setminus N[u]$, and consider a common neighbor $x \in C$ of $K \cup \{u\}$. Then $K \cup \{u, x\}$ induces a graph isomorphic to $K_r \setminus K_{1,r-2}$, which is impossible. Since each of these instances is K_{r-2}-free, applying Ramsey's theorem to each of them allows us to either construct an independent set of size $k-1$ in one of them (and thus output an independent set of size k in G), or to prove that each of them has at most

$O(k^{r-3})$ vertices. At the end, this algorithm outputs $O(n^2)$ instances, each having $O(k^{r-3})$ vertices. □

Since a $(K_r \setminus K_{1,r-1})$-free graph is $(K_{r'} \setminus K_{1,r'-2})$-free for $r' = r + 1$, we have the following:

Corollary 2 *For every $r \geqslant 2$,* MIS *in $(K_r \setminus K_{1,r-1})$-free graphs has a polynomial Turing Kernel.*

In other words, $(K_r \setminus K_{1,r-1})$ is a clique of size $r-1$ plus an isolated vertex. Observe that the previous corollary can actually be proved in a very simple way: informally, we can "guess" a vertex v of the solution, and return its non-neighborhood together with parameter $k-1$. Since this non-neighborhood is K_{r-1}-free, it can be reduced to a $O(k^{r-2})$-sized instance. This is perhaps the most simple example of a problem admitting a polynomial Turing kernel but no polynomial kernel, unless $NP \subseteq coNP/poly$ (as we will prove later in Theorem 19). By considering the complement of graphs, it implies the following even simpler observation: Maximum Clique has a $O(k^2)$ Turing kernel on *claw*-free graphs, but no polynomial kernel, under the same complexity-theoretic assumption.

Theorem 18 *For every $r \geqslant 3$,* MIS *in $(K_r \setminus K_{1,2})$-free graphs has a kernel with $O(k^{r-1})$ vertices.*

Proof. For $r = 3$, the problem is polynomial, so we assume $r \geqslant 4$. We first invoke Lemma 4. If the algorithm outputs an independent set of size at least $s = n^{\frac{1}{r-1}}$, then either $k \leqslant s$ and we are done (we output a trivially positive instance), or $k > n^{\frac{1}{r-1}}$ which implies that the instance is a kernel with $O(k^{r-1})$ vertices. Hence, we assume that the algorithm outputs a clique C of size at least $n^{\frac{1}{r-1}}$. We assume that this clique is maximal. We present a reduction rule in the case $|C| > (k-1)(r-4)+1$. If this rule cannot apply, then it means that the number of vertices of the reduced instance is $O(k^{r-1})$.

First observe that for every $u \in N(C)$, then either $|N_C(u)| = |C|-1$, or $|N_C(u)| \leqslant r-4$ (recall that $N_C(u) = N(u) \cap C$). Indeed, first observe that $N_C(u) < |C|$, since C is maximal. Then, suppose that $r-3 \leqslant |N_C(u)| \leqslant |C|-2$. Then u together with $r-3$ of its neighbors in C and 2 of its non-neighbors in C induce a graph isomorphic to $K_r \setminus K_{1,2}$, a contradiction. Let $B = \{u \in N(C) : |N_C(u)| = |C|-1\}$ and $D = \{u \in N(C) : |N_C(u)| \leqslant r-4\}$.

We claim that $C \cup B$ is a complete $|C|$-partite graph. To do so, we prove that for $u, v \in B$, $N_C(u) = N_C(v)$ implies $uv \notin E(G)$, and $N_C(u) \neq N_C(v)$ implies $uv \in E(G)$. Suppose that $N_C(u) = N_C(v) = C \setminus \{x\}$. If $uv \in E(G)$, then u, v, x together with $r-3$ vertices of C different from x induce a graph isomorphic to $K_r \setminus K_{1,2}$, which is impossible. Suppose now that $N_C(u) = C \setminus \{x_u\}$, $N_C(v) = C \setminus \{x_v\}$, with $x_u \neq x_v$. If $uv \notin E(G)$, then u, v, x_u together with $r-3$ vertices of C different from x_u and x_v induce a graph isomorphic to $K_r \setminus K_{1,2}$, which is impossible.

Thus, we now write $C \cup B = S_1 \cup \cdots \cup S_{|C|}$, where, for every $i, j \in \{1, \ldots, |C|\}$, $i \neq j$, S_i induces an independent set, and $S_i \cup S_j$ induces a complete bipartite graph. We assume $|S_1| \geqslant |S_2| \geqslant \cdots \geqslant |S_{|C|}|$. Recall that $|C| > (k-1)(r-4)+1$. Using the same arguments as previously, we can show that every vertex of D is adjacent to at most $r-4$ different parts among $C \cup B$: if a vertex $u \in D$ is adjacent to $r-3$ parts, then taking one vertex in each of these parts together with u and 2 non-neighbors of u in C induces a graph isomorphic to $K_r \setminus K_{1,2}$. Hence, for every $u \in D$, we have $|\{S_i : N(u) \cap S_i \neq \emptyset\}| \leqslant r-4$. Let $q = (k-1)(r-4)+1$. The reduction consists of removing $S_{q+1} \cup \cdots \cup S_{|C|}$. Clearly it runs in polynomial time.

Let G' denote the reduced instance. We now prove the safeness of this reduction rule. Obviously, if G' has an independent set of size k, then G does, since G' is an induced subgraph of G. It remains to show that the converse is also true. Let X be an independent set of G of size k. If $X \cap \left(\cup_{i=q+1}^{|C|} S_i\right) = \emptyset$, then X is also an independent set of size k in G', thus we suppose $X \cap \left(\cup_{i=q+1}^{|C|} S_i\right) = X_r \neq \emptyset$, which implies that $|X \cap D| \leq k-1$. In particular, since $C \cup B$ is a complete multipartite graph, there is a unique $i \in \{1, \ldots, |C|\}$ such that $X \cap S_i \neq \emptyset$, and $i \geqslant q+1$. Since every vertex of D is adjacent to at most $r-4$ parts of $C \cup B$, and since $q = (k-1)(r-4)+1$, there must exist $j \in \{1, \ldots, q\}$ such that $N(X \cap D) \cap S_j = \emptyset$. Moreover, $|S_j| \geqslant |S_i|$. Hence, $(X \setminus S_i) \cup S_j$ is an independent set of size at least k in G'.

Recall that we apply this reduction rule as long as $|C| > (k-1)(r-4)+1$. If it is not the case, then the instance has $O(k^{r-1})$ vertices, since, by Lemma 4, we have $|C| \geqslant n^{\frac{1}{r-1}}$, and thus $n \leqslant (kr+5)^{r-1}$, which concludes the proof. □

Observe that a $(K_r \setminus K_2)$-free graph is $(K_{r+1} \setminus K_{1,2})$-free, hence we have the following, which answers a question of [12].

Corollary 3 *For every $r \geqslant 1$,* MIS *in $(K_r \setminus K_2)$-free graphs has a kernel with $O(k^{r-1})$ vertices.*

5.2 Kernel lower bounds

We now give a sufficient criteria for a graph H to preclude any polynomial kernel for MIS in H-free graphs. In a nutshell, we characterize graphs which cannot appear in the "straightforward" cross-composition consisting in taking the complete join of several instances.

Definition 8 Given a graph H, a *join* is a bipartition of $V(H)$ into two non-empty subsets (A, B) such that for every $a \in A$ and $b \in B$, $ab \in E(H)$.

Theorem 19 *Let H be any fixed graph such that (i)* MIS *is NP-hard in H-free graphs, and (ii) H has no join. Then* MIS *does not admit a polynomial kernel in H-free graphs unless NP $\subseteq$ coNP/poly.*

Proof. We construct an OR-cross-composition from MIS in H-free graphs. For more details about cross-compositions, see [4]. Let $G_1, \ldots, G_t$ be a sequence of H-free graphs, and let $G' = G_1 + \cdots + G_t$ (recall that $+$ is the join operation, that it, there are all possible edges between $V(G_i)$ and $V(G_j)$, $i \neq j$). Then we have the following:

- $\alpha(G') = \max_{i=1\dots t} \alpha(G_i)$, since, by construction of G', any independent set cannot intersect the vertex set of two distinct graphs G_i and G_j.
- G' is H-free. Indeed, suppose that $X \subseteq V(G')$ induces a graph isomorphic to H, and let $X_j = X \cap V(G_j)$ for every $j \in [t]$. Since every G_i is H-free, at least two sets X_j, $X_{j'}$, $j \neq j'$ are non-empty. But then $(X_j, \cup_{s \neq j} X_s)$ is a join in H, a contradiction.

These two arguments imply a cross-composition from MIS in H-free graphs to MIS in H-free graphs. □

Naturally, the previous lower bound also holds for graphs H containing a graph H' as an induced subgraph fulfulling the statement of the theorem (since the class of H'-free graphs is included in the class of H-free graphs).

We now use this theorem to show that the polynomial kernel obtained in the previous section for $(K_r \setminus K_{1,s})$-free graphs, $s \leqslant 2$, is somehow tight.

Corollary 4 *For $r \geqslant 4$, and every $3 \leqslant s \leqslant r-1$,* MIS *in $(K_r \setminus K_{1,s})$-free graphs does not admit a polynomial kernel unless NP ⊆ coNP/poly.*

Proof. Observe that for these values of r and s, $(K_r \setminus K_{1,s})$ always contain as an induced subgraph the graph H defined as the disjoint union of K_1 and K_3, which does not have a join, while MIS is NP-hard in H-free (since it contains a triangle K_3). □

It would be interesting to find out whether there exist graphs H not falling into the statement of Theorem 19 for which there is no polynomial kernel. In other words: is Theorem 19 the only way to obtain kernel lower bounds in this case?

6 Conclusion and open problems

We made some signifiant progress toward the FPT/$W[1]$-hard dichotomy for MIS in H-free graphs, for a fixed graph H. At the cost of one reduction, we showed that it is $W[1]$-hard as soon as H is not chordal, even if we simultaneously forbid induced $K_{1,4}$ and trees with at least two branching vertices. Tuning this construction, it is also possible to show that if a connected H is not roughly a "path of cliques" or a "subdivided claw of cliques", then MIS is $W[1]$-hard. More formally, with the definitions of Section 2.2, the remaining connected open cases are when H has an almost strong clique decomposition on a subdivided claw or a nearly strong clique decomposition on a path. In this language, we showed that for every connected graph H with a strong clique decomposition on a P_3, there is an FPT algorithm. However, we also proved that for a very simple graph H with a strong clique decomposition on the claw, MIS is $W[1]$-hard. This suggests that the FPT/$W[1]$-hard dichotomy will be somewhat subtle. For instance, easy cases for the parameterized complexity do *not* coincide with easy cases for the classical complexity where each vertex can be blown into a clique. For graphs H with a clique decomposition on a path, the first unsolved cases are H having:

- an almost strong clique decomposition on P_3;
- a nearly strong clique decomposition on P_3;
- a strong clique decomposition on P_4.

For graphs H with a clique decomposition on the claw, an interesting open question is the case of $T_{1,1,s}$-free graphs (see notation preceding Theorem 4). We observe that a randomized FPT algorithm was later found in the $T_{1,1,2}$-free (or *cricket*-free) case [5], while $W[1]$-hardness on $T_{1,2,2}$-free is established in this paper (see Theorem 4)

For disconnected graphs H, we obtained an FPT algorithm when H is a cluster (*i.e.*, a disjoint union of cliques). We conjecture that, more generally, the disjoint union of two easy cases is an easy case; formally, *if* MIS *is FPT in G-free graphs and in H-free graphs, then it is FPT in $G \uplus H$-free graphs.*

A natural question regarding our two FPT algorithms of Section 4 concerns the existence of polynomial kernels. In particular, we even do not know whether the problem admits a kernel for very simple cases, such as when $H = K_5 \setminus K_3$ or $H = K_5 \setminus K_{2,2}$.

A more anecdotal conclusion is the fact that the parameterized complexity of the problem on H-free graphs is now complete for every graph H on four vertices, including concerning the polynomial kernel question (see Figure 9). Observe that the FPT/$W[1]$-hard dichotomy was recently settled for all graphs on five vertices [5], using tools from this paper.

Acknowledgements N. B. and P. C. are supported by the ANR Project DISTANCIA (ANR-17-CE40-0015) operated by the French National Research Agency (ANR).

We would like to thank two anonymous reviewers for their valuable comments which greatly improved the presentation of the paper.

References

1. Alekseev, V.E.: The effect of local constraints on the complexity of determination of the graph independence number. Combinatorial-Algebraic Methods in Applied Mathematics pp. 3–13 (1982). In Russian
2. Alekseev, V.E.: On the number of maximal independent sets in graphs from hereditary classes. Combinatorial-Algebraic Methods in Discrete Optimization pp. 5–8 (1991). (In Russian)
3. Bacsó, G., Lokshtanov, D., Marx, D., Pilipczuk, M., Tuza, Z., van Leeuwen, E.J.: Subexponential-time algorithms for maximum independent set in P_t-free and broom-free graphs. Algorithmica **81**(2), 421–438 (2019). DOI 10.1007/s00453-018-0479-5. URL `https://doi.org/10.1007/s00453-018-0479-5`

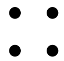
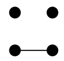
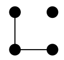
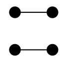
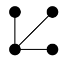
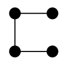
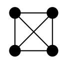
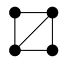
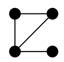
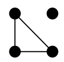
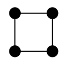
Graph	P	PK	PTK	FPT
	Obvious			
	Obvious			
	Obvious			
	[2]			
	[22]			
	[8]			
	Thm. 1	Ramsey		
	Thm. 1	Cor. 3		
	Thm. 1	Thm. 18		
		Cor. 4	Cor. 2	
				Thm. 2

Fig. 9 Status of the problem for graphs H on four vertices. A green cell represents a positive answer while a red cell represents a negative answer under classical complexity assumptions. P, PK, PTK respectively stand for *Polynomial*, *NP-hard but admits a polynomial kernel*, and *no polynomial kernel unless $NP \subseteq coNP/poly$ but admits a polynomial Turing kernel.*

4. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. **28**(1), 277–305 (2014)
5. Bonnet, É., Bousquet, N., Thomassé, S., Watrigant, R.: When maximum stable set can be solved in FPT time. In: P. Lu, G. Zhang (eds.) 30th International Symposium on Algorithms and Computation, ISAAC 2019, *LIPIcs*, vol. 149, pp. 49:1–49:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019). URL `https://doi.org/10.4230/LIPIcs.ISAAC.2019.49`
6. Cai, L., Chan, S.M., Chan, S.O.: Random separation: A new method for solving fixed-cardinality optimization problems. In: Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC), pp. 239–250 (2006)
7. Chen, J., Liu, Y., Lu, S., Sze, S., Zhang, F.: Iterative expansion and color coding: An improved algorithm for 3d-matching. ACM Trans. Algorithms

8(1), 6:1–6:22 (2012). DOI 10.1145/2071379.2071385. URL `https://doi.org/10.1145/2071379.2071385`
8. Corneil, D.G., Perl, Y., Stewart, L.K.: A linear recognition algorithm for cographs. SIAM J. Comput. **14**(4), 926–934 (1985)
9. du Cray, H.P., Sau, I.: Improved FPT algorithms for weighted independent set in bull-free graphs. Discrete Mathematics **341**(2), 451–462 (2018)
10. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015)
11. Dabrowski, K.: Structural solutions to maximum independent set and related problems. Ph.D. thesis, University of Warwick (2012)
12. Dabrowski, K., Lozin, V.V., Müller, H., Rautenbach, D.: Parameterized complexity of the weighted independent set problem beyond graphs of bounded clique number. J. Discrete Algorithms **14**, 207–213 (2012)
13. Diestel, R.: Graph Theory, 4th Edition, *Graduate texts in mathematics*, vol. 173. Springer (2012)
14. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer (2013)
15. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
16. Grzesik, A., Klimošová, T., Pilipczuk, M., Pilipczuk, M.: Polynomial-time algorithm for maximum weight independent set on p6-free graphs. In: Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019, pp. 1257–1271 (2019). DOI 10.1137/1.9781611975482.77. URL `https://doi.org/10.1137/1.9781611975482.77`
17. Hermelin, D., Mnich, M., van Leeuwen, E.J.: Parameterized complexity of induced graph matching on claw-free graphs. Algorithmica **70**(3), 513–560 (2014)
18. Karthick, T.: Independent sets in some classes of $\mathrm{S}_{i,j,k}$-free graphs. J. Comb. Optim. **34**(2), 612–630 (2017)
19. Karthick, T., Maffray, F.: Maximum weight independent sets in classes related to claw-free graphs. Discrete Applied Mathematics **216**, 233 – 239 (2017)
20. Lokshtanov, D., Vatshelle, M., Villanger, Y.: Independent set in P_5-free graphs in polynomial time. In: Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, pp. 570–581 (2014)
21. Marx, D.: Parameterized complexity of independence and domination on geometric graphs. In: Parameterized and Exact Computation, Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006, Proceedings, pp. 154–165 (2006). DOI 10.1007/11847250_14. URL `https://doi.org/10.1007/11847250_14`
22. Minty, G.J.: On maximal independent sets of vertices in claw-free graphs. Journal of Combinatorial Theory, Series B **28**(3), 284 – 304 (1980)
23. Naor, M., Schulman, L.J., Srinivasan, A.: Splitters and near-optimal derandomization. In: Proceedings of IEEE 36th Annual Foundations of Com-

puter Science, pp. 182–191 (1995). DOI 10.1109/SFCS.1995.492475
24. Poljak, S.: A note on stable sets and colorings of graphs. Commentationes Mathematicae Universitatis Carolinae pp. 307–309 (1974)
25. Reed, B., Smith, K., Vetta, A.: Finding odd cycle transversals. Operations Research Letters **32**(4), 299 – 301 (2004)
26. Sloper, C., Telle, J.A.: An overview of techniques for designing parameterized algorithms. Comput. J. **51**(1), 122–136 (2008). DOI 10.1093/comjnl/bxm038. URL `https://doi.org/10.1093/comjnl/bxm038`
27. Thomassé, S., Trotignon, N., Vuskovic, K.: A Polynomial Turing-Kernel for Weighted Independent Set in Bull-Free Graphs. Algorithmica **77**(3), 619–641 (2017)
28. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. Theory of Computing **3**(1), 103–128 (2007)