Infinite induced-saturated graphs

Marthe Bonamy[1], Carla Groenland[*2], Tom Johnston[3], Natasha Morrison[†4], and Alex Scott[‡5]

[1]*CNRS, LaBRI, Université de Bordeaux, France.*
[2]*Delft Institute of Applied Mathematics, TU Delft, the Netherlands.*
[3]*University of Bristol, United Kingdom and Heilbronn Institute of Mathematical Research, United Kingdom.*
[4]*Department of Mathematics and Statistics, University of Victoria, Canada.*
[5]*Mathematical Institute, University of Oxford, United Kingdom.*

June 16, 2025

Abstract

A graph G is H-induced-saturated if G is H-free but deleting any edge or adding any edge creates an induced copy of H. There are various graphs H, such as P_4, for which no finite H-induced-saturated graph G exists. We show that for every finite graph H that is not a clique or stable set, there always exists a countable H-induced-saturated graph. In fact, a stronger property can be achieved: there is an H-free graph G such that G' contains a copy of H whenever $G' \neq G$ is obtained from G by deleting and adding edges so that only a finite number of changes have been made incident to any vertex.

1 Introduction

For a finite graph H, let P_H be the collection of graphs that do not contain H as an induced subgraph (also known as the class of H-free graphs). In this paper, we study how "unstable", "fragile" or "isolating" this property can be in the following sense:

Is there a graph $G \in P_H$ such that $G' \notin P_H$ for every graph $G' \neq G$ which is "close to" G?

One natural definition of "close" is to say that G is close to G' if G' can be obtained from G by *disturbing* a single pair of distinct vertices $u, v \in V(G)$: if uv is an edge in G, we delete it, and if it is not an edge, we add it. For our main result, we allow for even stronger disturbances in the structure of G. A *locally finite disturbance* of G is a graph obtained from G by disturbing arbitrarily many (at least one) pairs under the constraint that for any vertex v, the number of disturbed pairs involving v is finite. Our main result is the following.

Theorem 1. *Let H be a finite graph. Then there exists a countably infinite H-free graph G_H such that every locally finite disturbance of G_H has an induced copy of H if and only if H is not a clique or a stable set.*

[*]Research supported by the Dutch Research Council (NWO, VI.Veni.232.073)
[†]Research supported by NSERC Discovery Grant RGPIN-2021-02511
[‡]Research supported by EPSRC grant EP/W015404/

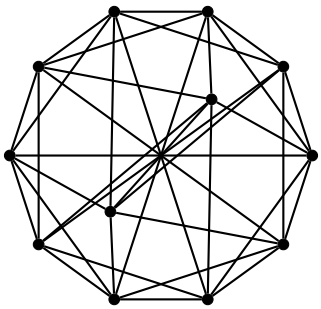


Figure 1: The depicted graph (the complement of the icosahedral graph) is induced-saturated for P_5.

The special case of individual disturbances has been previously studied. A graph G is called *H-induced-saturated* if G has no induced subgraph isomorphic to H but deleting any edge or adding any edge creates an induced copy of H. Our main result directly implies the following statement about induced-saturated graphs.

Corollary 2. *For any finite graph H which is not a clique or stable set, there is a countably infinite H-induced-saturated graph G_H.*

Finite induced-saturated graphs and variations on this have been studied extensively in recent years (see e.g. [1, 2, 6, 10, 12, 15, 18, 20]). It is immediately clear that there is no (finite or infinite) H-induced-saturated graph when H is a clique or a stable set (except on those on two vertices), and one of the first results on induced-saturated graphs showed that there is no finite P_4-induced-saturated graph [15]. This result can also be shown by noting that every finite P_4-free graph (on at least two vertices) contains two vertices that are twins[1] (where u and v may be adjacent or not). Disturbing the pair uv does not create a copy of P_4 in this case, and so the resulting graph is still P_4-free. After some partial progress [6, 18, 20] by various authors, Dvořák [10] showed P_t-induced-saturated graphs exist for all $t \geq 6$. It is straightforward to show there are P_t-induced-saturated graphs for $t = 2, 3$, so the only path remaining was P_5. The authors found examples of P_5-induced-saturated graphs shortly afterwards using a computer search [3], one of which is shown in Figure 1. This means that a finite P_t-induced-saturated graph exists if and only if $t \neq 1, 4$.

The motivation behind the study of induced-saturated graphs stems from a long history of research concerning "saturated" graphs, initiated by Erdős, Hajnal and Moon [11]. A graph is said to be *H-saturated* if it contains no subgraph isomorphic to H and the addition of any edge creates such a subgraph. Note that, in contrast to the induced setting, removing an edge can never create a subgraph. It is easy to construct an H-saturated graph: start from a finite graph without an H subgraph (for example, the empty graph) and keep adding edges as long as the chosen edge does not create a subgraph isomorphic to H. If there are no more edges that can added, the graph must be H-saturated. With the question of existence straightforward, the minimum and maximum number of edges in saturated structures is studied instead. See the comprehensive survey of Currie, Faudree, Faudree and Schmitt [7] for an extensive literature review on saturation in graphs and related variants.

Although not directly related, our work bears some similarity to the research of universal elements. Given a class $\mathcal{G}$ of countable graphs, a *universal element* is a graph $U \in \mathcal{G}$ that contains all graphs from $\mathcal{G}$ as subgraph. For the class of all countable graphs, such a universal

[1] Say vertices u and v are twins if every vertex $w \notin \{u, v\}$ is adjacent to u if and only if it is adjacent to v. They are called true twins if $uv \in E$ and false twins otherwise.

element exists (often named after Rado [17]). The existence of such objects has been studied for various families of countable graphs, including planar graphs and graph classes excluding a particular subgraph (see e.g. [8, 9, 13, 16]). The universal element U of the class of all countable graphs is also ultrahomogeneous: every isomorphism between finite substructures of U is a restriction of an automorphism of U. Lachlan and Woodrow [14] completely classified the countable ultrahomogeneous graphs. There have been various similar classifications of homogeneity in countable discrete structures [4, 5, 19].

In the remainder of the section, we illustrate the key ingredients in the proof of Theorem 1. We first describe how one might hope to deal with individual disturbances in the context of Corollary 2 before discussing how this can be adapted to handle locally finite disturbances. The formal proof of Theorem 1 is fully described in the later sections and does not rely on the informal discussion which makes up the remainder of this section.

1.1 P_4 and fixing operations

The case of P_4-free graphs is particularly interesting as there is no finite P_4-induced-saturated graph.

We find an infinite P_4-induced-saturated graph G by considering a sequence $(G_i)_{i\in\mathbb{N}}$ of graphs and taking its limit. The idea is to keep track of a list of pairs which are "bad", in the sense that disturbing them does not create a copy of P_4. We obtain G_{i+1} from G_i by attaching a gadget that "fixes" the next bad pair in the list without creating a copy of P_4. We then update the list by adding the bad pairs involving a new vertex to the end of the list. The construction for P_4 is as follows.

The fixing operation for a bad edge xy is to add a false twin to each of x and y, and the fixing operation for a bad non-edge xy is to add a true twin to each of x and y. Since P_4 has no twins, adding a twin can never create an induced copy of P_4. Moreover, in the resulting graph disturbing the pair $\{x, y\}$ indeed creates a P_4. The sequence $(G_i)_{i\in\mathbb{N}}$ of graphs is obtained by considering an arbitrary P_4-free graph, computing a list of its bad pairs, then obtaining G_{i+1} from G_i by fixing its first bad pair and updating the list of bad pairs.

We give an example below (see also Figure 2). Let G_1 be an edge on vertices u_1, u_2. Disturbing u_1u_2 results in a P_4-free graph, hence u_1u_2 is a bad edge in G_1. The list of bad pairs in G_1 is $\{\{u_1, u_2\}\}$. We define G_2 as the graph obtained from G_1 by adding a false twin v_1 to u_1 and a false twin v_2 to u_2. Disturbing the edge u_1u_2 in G_2 now creates an induced P_4, therefore the pair $\{u_1, u_2\}$ is fixed in G_2. However, now disturbing the non-edge u_1v_1 results in a P_4-free graph. The bad pairs in G_2 are precisely $\{\{u_1, v_1\}, \{u_2, v_2\}\}$. We define G_3 as the graph obtained from G_2 by adding a true twin w_1 to u_1 and a true twin x_1 to v_1. This fixes the pair $\{u_1, v_1\}$ and the bad pairs in G_3 are $\{\{u_2, v_2\}, \{u_1, w_1\}, \{x_1, v_1\}\}$.

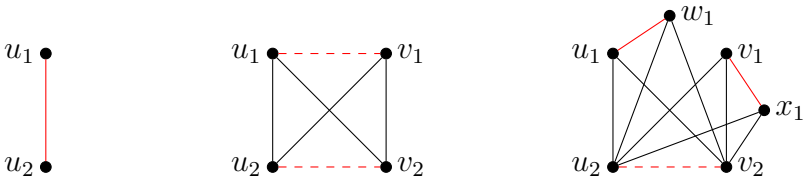


Figure 2: A sequence of graphs $G_1, G_2, G_3, \ldots$ is obtained by repeatedly applying a fixing operation. The red edges and red non-edges are not fixed.

We note that in the above construction, for any i, for any bad pair $\{x, y\}$ of a graph G_i, there is an index j such that disturbing $\{x, y\}$ in G_j (or any subsequent graph) yields an

induced P_4. In other words, any bad pair is ultimately fixed. Therefore, there is a countably infinite P_4-free graph G_{P_4} such that disturbing any pair in G_{P_4} results in an induced copy of P_4: Corollary 2 is true for $H = P_4$ and, more generally, for any graph H that admits fixing operations for bad pairs.

1.2 C_5 and gatekeepers

The fixing operations for P_4 defined above rely heavily on the strong structural properties of P_4-free graphs. Since we need to prove Corollary 2 for every graph H that is not a clique or stable set, we will need something much more general. The naïve way to define a fixing operation for H is by gluing a copy of $H - e$ with the missing edge aligned with the bad non-edge (or a copy of $H + e$ with the extra edge aligned with the bad edge). While this is a sure-fire way to guarantee that disturbing the bad pair will result in an induced copy of H, the pitfall is that the gluing might create a copy of H. One may hope to avoid this by choosing carefully which edge to add or delete in H, but this may not always be possible (consider $H = P_4$).

However, for $H = C_5$, this *is* a good approach (see Figure 3). To fix a bad non-edge xy, we may glue $C_5 - e$ by adding three vertices u_1, u_2, u_3 and four edges $xu_1, u_1u_2, u_2u_3, u_3y$. Note that the resulting graph is C_5-free, as every induced copy of C_5 involving some new edge would involve all of them. Similarly, to fix a bad edge xy, we may glue $C_5 + e$ by adding three vertices u_1, u_2, u_3 and five edges $xu_1, u_1y, xu_2, u_2u_3, u_3y$. It is not hard to see that the resulting graph is also C_5-free. This idea does not just work for C_5, but in fact works for many graphs,

Figure 3: The fixing operation for non-edges (left) and edges (right) is depicted. The blue vertices represent newly created vertices.

including every 3-connected graph H that is not a clique. Indeed, if we "fix" the pair $\{x, y\}$ in a similar fashion to Figure 3, then $\{x, y\}$ forms a 2-cut in the resulting graph between the "old" and "new" vertices. If H is 3-connected, then it cannot contain vertices of both types so no new copies of H can be created. This proves Corollary 2 for any 3-connected graph H.

We now formalise what makes C_5 and 3-connected graphs (but not P_4) behave well. An edge uv in a graph H is a *gatekeeper* if for any H-free graph G, gluing a copy of $H - uv$ on a non-edge of G results in an H-free graph. A similar definition holds for non-edges. We say H admits a gatekeeper *of each type* if it contains both an edge and a non-edge that are gatekeepers.

A graph H that admits a gatekeeper of each type admits a fixing operating for bad edges as well as for bad non-edges, using the naïve approach described above. Note that a path contains no edge that is a gatekeeper.

1.3 C_5 with a leaf and cores

Unfortunately, even small modifications to a graph can make a graph which is much harder to handle. For example, consider the graph obtained from C_5 by adding a pendant edge to one of the vertices. While we easily found fixing operations for C_5 using gatekeepers, this new graph does not admit gatekeepers of either type. Instead, we focus on the C_5 and work

within the class of C_5-free graphs. If we work within this class, we can use the gatekeepers for C_5 and not create a copy of C_5 or of our target graph. These gatekeepers are not quite enough to guarantee a disturbance creates a copy of C_5 plus a leaf, but this is easily solved by adding a leaf to every vertex we add. Note that the leaves cannot be in a C_5, so this is a valid modification. We now formalise and extend this idea using the notion of cores.

The 2-core H^* of a finite graph H is obtained by iteratively removing vertices of degree at most 1. Crucially, if we take a graph G which is H^*-free, then adding a leaf to each vertex of G does not create a copy of H^* let alone H, nor does adding an isolated vertex. Given a fixing operation for H^*, we can create a fixing operation for H as follows. Throughout the process, we maintain that the graph created so far is H^*-free. We interchange fixing operations for H^* with steps that either add a leaf to each vertex or add an isolated vertex. Since every bad pair is eventually fixed by the fixing operation for H^*, if we disturb a pair then we can embed a copy of H^*. We can extend our copy of H^* into a copy of H as we have added infinitely many leaves and isolated vertices.

Therefore, if a graph H^* admits fixing operations, any graph H whose 2-core is H^* admits fixing operations. We can generalise this notion by iteratively deleting any vertex with at most k neighbours or at most ℓ non-neighbours.

In particular, the 3-core H^* of a graph H is obtained by iteratively removing vertices of degree at most 2. By the discussion above, Corollary 2 holds for a graph H if its 3-core is 3-connected. We are also done when the complement $\overline{H}$ of H is 3-connected: the complement of an $\overline{H}$-induced-saturated graph is H-induced-saturated. When H has a 2-cut and both "sides" contain at least three vertices, then $\overline{H}$ is close to having a 3-connected 3-core. However, there is one "bad" 2-cut to take into account, depicted in Figure 4.

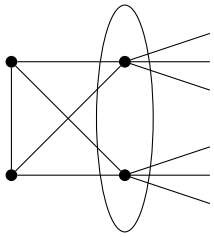


Figure 4: A "bad" 2-cut is shown. No such cuts are present in the 3^*-core.

We define the 3^*-*core* of a graph H as the graph obtained from H by iteratively removing vertices of degree at most 2, as well as true twins of degree 3 ("bad" 2-cuts). As before, fixing operations for a graph H^* can be extended to any graph H whose 3^*-core is H^*. If a graph H has a non-empty 3^*-core, then either the 3^*-core of H or the 3^*-core of $\overline{H}$ is 3-connected. There is a small catch: the 3^*-core could be a clique and this will give a few more cases to handle.

1.4 Outline of the proof

We can now give a broad overview of the steps in our proof. The proof will be given formally in Section 6.6.

- We give a fixing operation for all H whose 3^*-core is 3-connected and not a clique. The reader should already be able to verify this for the weaker Corollary 2 from the discussion above. We provide the stronger statements needed for Theorem 1 in Section 3.
- We give a fixing operation for all H whose 2-core is a $K_{1,1,p}$ or $K_{2,p}$ and a direct construction for all forests with a unique vertex of maximum degree in Section 5.

- We prove a structure theorem (Theorem 10) that shows that for all finite graphs H on at least 12 vertices, either H or its complement falls into one of the above categories. This is done in Section 4.
- We provide direct constructions for specific graphs H on at most 7 vertices in Section 6.
- We use a computer to check that for all graphs H on at most 11 vertices, either H or its complement falls into one of the above categories. This is detailed in Section 6.3.

1.5 From individual disturbances to locally finite disturbances

For Corollary 2, we only need to generate a copy of H when we make a single disturbance: we either turn an edge into a non-edge or vice versa. For Theorem 1, we allow many pairs to be disturbed at once, and the arguments above no longer directly apply.

We describe one common way in which we show that a pair has been fixed. After a locally finite disturbance, we show how to embed H vertex-by-vertex. Suppose that we already embedded $h_1, \dots, h_s$. If there are infinitely many options from which to choose the next vertex h_{s+1}, then there is always an option for which the adjacencies to $h_1, \dots, h_s$ have not been altered by the locally finite disturbance. Indeed, for each h_i, there are only finitely many v so that $\{h_i, v\}$ has been altered. One way to generate infinitely many options is to blow up vertices into infinite cliques. This does not always work, since it may create a copy of H. Moreover, it also creates new edges to be "fixed". Nevertheless, the difference between allowing a single disturbance or allowing any locally finite disturbance is reasonably small in most of our proofs. In some places, it does require additional insights, such as for 3-connected graphs (handled by Lemma 3).

There is one additional issue that comes up when we prove our stronger variant. Assume that a graph H admits fixing operations as in Section 1.1. For Corollary 2, we start from a finite graph, and fixing a pair adds finitely many vertices, hence finitely many new bad pairs. Each graph of the sequence is finite, and new bad pairs can simply be added to the end of the list of existing bad pairs. However, in the proof of Theorem 1, we add countably infinitely many vertices to fix a single bad pair, thus possibly infinitely many new bad pairs. Adding bad pairs to the "end" of an infinite list of existing bad pairs is not well-defined. In the proof of Lemma 17 we explain how to carefully schedule the fixes.

2 Notation and definitions

The word "graph" in this paper allows for infinite graphs, but the vertex set is always required to be countable. When we search for a (strongly) H-induced-saturated graph, H will always refer to a finite graph. We only consider simple graphs (without self-loops, directed edges or parallel edges).

A *clique* is a set of vertices that are pairwise adjacent and a *stable set* is a set of vertices that are pairwise non-adjacent. A connected graph $G = (V, E)$ is *k-connected* if the graph has at least $k + 1$ vertices and remains connected after removing any $k - 1$ vertices. For graphs G and H, we say that G is *H-free* if G does not contain H as induced subgraph.

Write $N(v)$ for the neighbourhood of v, that is, the set of vertices adjacent to v, and write $N[v] = N(v) \cup \{v\}$ for the closed neighbourhood. We say two vertices u, v are *true twins* if $N[u] = N[v]$ and *false twins* if $N(u) = N(v)$ (and so u and v are not adjacent). Write uv to denote a pair of vertices $\{u, v\}$ with $u \neq v$. Write P_t for the path on t vertices and K_t for the complete graph on t vertices.

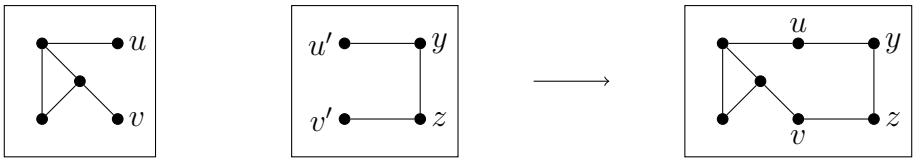


Figure 5: An example is given on how two graphs can be glued on non-edges $\{u, v\}$ and $\{u', v'\}$.

The definitions below are similar to those in the introduction, but now formally stated in full generality.

Locally finite disturbance For a pair of distinct vertices $u, v \in V(G)$, *disturbing* the pair uv is the operation which removes uv from $E(G)$ if it is present and adds it if it is not present. A *locally finite disturbance* of G is a graph obtained from G by disturbing arbitrarily many pairs (at least one, possibly infinitely many) under the constraint that for any vertex v, the number of disturbed pairs involving v is finite.

Strongly saturating For a finite graph H, say a graph G is *strongly H-induced-saturated* if G does not contain H as an induced subgraph, but any locally finite disturbance of G does. Note that the very notion of strongly H-induced-saturated requires G to be infinite. We say that G is *H-induced-saturated* if G does not contain H as an induced subgraph, but a copy of H is created when an edge is added to G or removed from G.

Fixed and unfixed edges Say a pair xy in a graph G is *unfixed* (for H) if there is a locally finite disturbance which disturbs the pair xy and does not result in an induced copy of H. Otherwise we call the pair *fixed* (for H). Note that a graph G is strongly H-induced-saturated if and only if all pairs of G are fixed for H.

Gluing Given two graphs G and G' on disjoint vertex sets with $A \subseteq V(G)$ and $A' \subseteq V(G')$, and a bijection $f : A' \to A$, the graph G'' obtained from *gluing* G' on G along f has $V(G'') = V(G) \cup V(G') \setminus A'$ and $uv \in E(G'')$ for distinct $u, v \in V(G'')$ if and only if

1. $uv \in E(G)$,
2. $uv \in E(G')$,
3. $u \in A$ and $f^{-1}(u)v \in E(G')$, or
4. $u, v \in A$ and $f^{-1}(u)f^{-1}(v) \in E(G')$.

We will most commonly apply this operation for $A = \{u, v\}$, $A' = \{u', v'\}$ and $f(u) = u', f(v) = v'$. An example of this is given in Figure 5.

Fixing operations We will sometimes need to maintain the stronger property that the graph is not just H-free but that it also does not contain a copy of some core of H. For this reason, we define a fixing operation relative to a class of graphs, as follows.

Given a class of graphs $\mathcal{F}$, we say the finite graph H admits an *edge fixing operation* (resp. *non-edge fixing operation*) for $\mathcal{F}$ if for every $G \in \mathcal{F}$ and every edge (resp. non-edge) xy of G, there exists a graph G' obtained from gluing a graph onto G such that

- $G' \in \mathcal{F}$, and

- any locally finite disturbance of G' which disturbs xy contains a copy of H.

Crucially, G is an induced subgraph of G'. We say H admits a *fixing operation* for $\mathcal{F}$ if it admits both an edge fixing operation and a non-edge fixing operation. We show in Lemma 17 that if H admits a fixing operation for a non-empty class $\mathcal{F}$ of H-free graphs, then there exists a strongly H-induced-saturated graph.

Gatekeepers An edge (resp. non-edge) uv in a graph H is a *gatekeeper* if for any H-free graph G, gluing a copy of $H - uv$ on a non-edge (resp. edge) of G results in an H-free graph.

Cores Given a graph H, the (k, ℓ)*-core* is the graph obtained by iteratively removing vertices which have fewer than k neighbours or less than ℓ non-neighbours. We write k*-core* to refer to the $(k, 0)$-core.

The 3^**-core* of a graph H is obtained by iteratively removing vertices which have at most 2 neighbours and removing pairs uv of vertices with $|N[u] \cup N[v]| \leq 4$.

3 Graphs with a 3-connected core

In this section we prove Theorem 1 in the case where H has a 3-connected 3^*-core which is not a clique.

In order to handle locally finite disturbances (rather than single edge disturbances), we will need the following result. Given a graph H with a pair of distinct vertices uv, we define the infinite graph $H_{uv\to c}$ (resp. $H_{uv\to s}$) as the graph obtained from H by disturbing the pair uv and then blowing up every vertex other than u and v into an infinite clique (resp. stable set). When we blow up a vertex v into an infinite clique (or stable set), we replace the vertex v by new vertices inducing an infinite clique (or stable set) and add edges from all of new vertices to all of the vertices that used to be adjacent to v.

Lemma 3. *Let H be a finite graph which is not a clique or stable set. Then H has an edge uv such that one of $H_{uv\to c}$ and $H_{uv\to s}$ contains no induced copy of H.*

Proof. First, suppose that uv is an edge in H such that neither u nor v has a true twin. We will show that $H_{uv\to c}$ contains no induced copy of H. Note that the property of being true twins is an equivalence relation and contracting the equivalence classes gives a graph H' which has no true twins. This leaves the vertices u and v unchanged as they have no true twins. Note that H' is an induced subgraph of H so if there is an induced copy of H in $H_{uv\to c}$, then there must be a copy of H' as well. Since H' has no true twins, every vertex of H' must be in the blow-up of a different vertex in $H_{uv\to c}$. But $H_{uv\to c}$ is the same as the graph obtained by blowing up each vertex of $H' - uv$ (instead of $H - uv$) except for u and v into an infinite clique. So a copy of H' in $H_{uv\to c}$ gives an induced copy of H' in $H' - uv$, a contradiction.

A similar argument shows that if neither u and v contain a false twin, the graph $H_{uv\to s}$ contains no induced copy of H.

Next, we argue that a vertex v cannot have both a true twin x and a false twin y. Indeed, x is adjacent to v, so y should also be adjacent to x (since v, y are false twins). But y is not adjacent to v so x should not be adjacent to y. This gives a contradiction.

So we may and will assume that for each edge ab of H, either a has a false twin and b a true twin, or vice versa (as otherwise we are done). This means we can partition the vertices of H into three sets, a set A of vertices with a true twin, a set B of vertices with a false twin B and a set C of vertices with neither a true twin nor a false twin. Since for each edge ab of

H, either a has a false twin and b a true twin, or vice versa, edges can only go between A and B. In particular, there are no edges in A. But any vertex in A must have an edge to its true twin in A so A is empty. But then there are no edges at all in our graph, contradicting our assumption that H is not a stable set. □

Since the complement of a stable set or clique is again a stable set or clique, we immediately get the following corollary by taking complements.

Corollary 4. *For every non-trivial finite graph H, there is a non-edge uv such that one of $H_{uv\to c}$ and $H_{uv\to s}$ contains no induced copy of H.*

We next repeat two simple, but important, observations.

Observation 5. *Let H be a finite 3-connected graph and let G and G' be two graphs which are H-free. Let G'' be obtained by gluing the graphs G and G' together along either an edge or a non-edge xy. Then G'' is H-free.*

Indeed, any copy of H would need to contain both a vertex $u \in V(G) \setminus \{x, y\}$ and a vertex $v \in V(G') \setminus \{x, y\}$, but we can disconnect u and v in G'' by removing the vertices x and y. This gives a vertex cut of H of size 2, contradicting the assumption that H is 3-connected. In the terminology introduced above, the observation says that any two vertices in H are gatekeepers.

Observation 6. *Let H be a finite graph with a pair of distinct vertices uv. Then, for $G' \in \{H_{uv\to c}, H_{uv\to s}\}$, any locally finite disturbance of G' which disturbs uv contains a copy of H.*

We repeat the argument here for completeness. The disturbance brings the pair uv to the same status it had in H; since the disturbance is locally finite, we can embed a copy of H vertex-by-vertex, as follows. We send u, v in H to u, v in G' respectively. We then iterate through the remaining vertices of H and map each vertex to one of the vertices in the blow-up of this vertex in G'. There are infinitely many options in each blow-up, so at least one of these vertices is not incident with the finite number of disturbances incident to the finite number of vertices we have embedded so far, and we map h to any such vertex.

We can now define our fixing operation and will then extend this to handle cores.

Lemma 7. *If H is a finite graph which is 3-connected and not a clique, then H admits a fixing operation for the class of H-free graphs.*

Proof. Suppose that H is a finite 3-connected graph and we are given an H-free graph G and an unfixed edge xy of G. By Lemma 3, we may assume that H has an edge uv such that there is a graph $G' \in \{H_{uv\to c}, H_{uv\to s}\}$ that is H-free. Let G'' be the graph formed by gluing together G and G' along the edge xy of G and the edge uv of G'. By Observation 5, the graph G'' is H-free. Any locally finite disturbance of G'' which disturbs xy contains a copy of H using vertices from G' by Observation 6. So indeed xy is fixed in G''.

The fixing operation for non-edges xy is analogous but relies on Corollary 4 instead of Lemma 3. □

Next, we provide a technical lemma which allows us to extend the fixing operations to cores. Operations 3 and 4 will only be needed for graphs H on at most 11 vertices in Section 6. We write $\delta(G)$ for the minimum degree of the graph G and $\deg_G(v)$ for the degree of vertex $v \in V(G)$ in G.

Lemma 8. *Let H and C be finite graphs such that H admits a fixing operation for the class of C-free graphs. Let H' be obtained from H via one of the following operations:*

1. *adding a new vertex with at most $k < \delta(C)$ neighbours, or*
2. *adding a new vertex with at most $\ell < \delta(\overline{C})$ non-neighbours, or*
3. *if $\delta(C) \geq 2$ and C has no vertex v with $\deg_C(v) = 2$ and two non-adjacent neighbours, adding a new vertex adjacent precisely to two non-adjacent vertices of H,*
4. *if $\delta(C) \geq 2$ and C has no vertex v with $\deg_C(v) = 2$ and two adjacent neighbours, adding a new vertex adjacent precisely to two adjacent vertices of H, or*
5. *if C is 3-connected and not a clique, adding a pair of adjacent new vertices onto H that are both adjacent to the same two vertices of H.*

Then H' also admits a fixing operation for the class of C-free graphs.

Proof. Throughout this proof, we start with a C-free graph G_0 and we assume that the pair xy is unfixed. Moreover, we assume H admits a fixing operation, so in particular there is a C-free graph G obtained by gluing a graph onto G_0, such that any locally finite disturbance of G which disturbs the pair xy contains a copy of H.

Suppose first that H' is obtained from H by adding a vertex w to H with neighbours $v_1, \ldots, v_k \in H$, with $k < \delta(C)$. For each choice of k vertices $(u_1, \ldots, u_k)$ from G, we add an infinite stable set fully connected to the vertices $(u_1, \ldots, u_k)$. Let G' be the resulting graph, which is by definition obtained by gluing a graph onto G, and so can also be obtained by gluing a graph onto G_0. Crucially, all vertices in $V(G') \setminus V(G)$ have degree $k < \delta(C)$ and so none of them can be present in a copy of C. Since G is C-free, G' must also be C-free. It remains to show that the pair xy is now fixed. Any locally finite disturbance in G' which disturbs xy must create a copy of H using solely vertices of G. Let $u_1, \ldots, u_k$ be the vertices in this copy that perform the roles of $v_1, \ldots, v_k$ in H. In G', there is an infinite stable set S which is adjacent to exactly $u_1, \ldots, u_k$, and for each vertex in the copy of H, only a finite number of edges to S have been modified. Hence there is a vertex $s \in S$ for which none of the edges to this copy have been modified and this provides us with a copy of H' in G'. Hence H' admits a fixing operation for the class of C-free graphs.

A similar argument applies when H' is obtained by adding $\ell < \delta(\overline{C})$ non-neighbours $v_1, \ldots, v_\ell$ to H: we repeat the "complement" of the argument above, adding an infinite clique which is adjacent to all vertices except for $u_1, \ldots, u_\ell$ (for each collection of ℓ distinct vertices).

If $\delta(C) \geq 2$ and every degree 2 vertex has adjacent neighbours, then we also follow a similar construction. Let H' be the graph obtained from H by adding a new vertex w adjacent to non-adjacent vertices v_1, v_2 in H. Let G' be obtained from G by adding, for each pair uu' of non-adjacent vertices in G, an infinite stable set adjacent to u and u'. Again, G' can be obtained by gluing a graph onto G_0. No copy of C can use the vertices from $V(G') \setminus V(G)$, since each vertex in C must have degree 2, and if it has degree 2, the neighbours must be adjacent. A copy of H' is created after a locally finite disturbance of the edge xy in G' for the same reason as above. For the fourth operation (when $\delta(C) \geq 2$ and the neighbours of degree 2 vertices are always are non-adjacent) we analogously glue an infinite stable set to pairs of adjacent vertices of G instead.

Finally, suppose that C is 3-connected and not a clique. Suppose moreover that H' is obtained by adding a pair u, v of adjacent new vertices onto H that are both adjacent to the same two vertices a, b of H, that is, $N'_H[u] = N'_H[v] = \{a, b, u, v\}$.

We first handle the case in which $ab \notin E(H)$ and C is a clique minus an edge, as this requires a different construction. Since C is 3-connected, we must have $|V(C)| \geq 5$. For each

choice of a', b' from G for which $a'b' \notin E(G)$, we add two infinite stable sets to G which are fully connected to each other and to a', b', and for $a'b' \in E(G)$, we add an infinite clique connected to a', b'. If C is any other 3-connected graph which is not a clique, then for each choice of a', b' from G, we add an infinite clique connected to a', b'. Let G' be the resulting graph.

We argue again that G' is C-free, which follows the "gatekeeper" idea from the introduction. Suppose towards a contradiction that G' contains a copy C' of C. We first argue that C' must be contained in $\{a', b'\}$ union the new vertices we just glued onto G. Since G is C-free and C is non-empty, C' needs to use at least one vertex $x' \in V(G') \setminus V(G)$. Let a', b' denote the pair of vertices in G that x' is adjacent to, and suppose that C' also contains $x \in V(G) \setminus \{a', b'\}$. Then $\{a', b'\}$ forms a 2-cut in G' separating x' from x, and hence separating x from x' in C'. This contradicts the assumption that C is 3-connected. Hence, $C' \cap V(G) \subseteq \{a', b'\}$.

When $a'b' \in E(G)$, then C' is contained in a clique, a contradiction as C is not a clique. When $a'b' \notin E(G)$ and C is a clique minus an edge, the graph C' is contained in a tri-partite graph, which is not possible either as C contains a copy of K_4. Hence, G' is C-free as desired.

The graph G' is again obtained by gluing a graph onto G_0 and we also repeat a similar argument to show the pair xy has been fixed. Any locally finite disturbance in G' which disturbs xy must create a copy of H using solely vertices of G. Let a', b' be the vertices in this copy that perform the role of a, b in H. Suppose first that in G', there is an infinite clique S which is adjacent to exactly a', b'. We first pick a vertex $x' \in S$ for which none of the edges to a', b' have been modified. Next, we pick another vertex $y' \in S$ for which none of the edges to a', b', x' have been modified. Together with x', y', the copy of H becomes a copy of H' in the locally finite disturbance. The other case is when C is a clique minus an edge and $a'b' \notin E(G')$, in which case we glued on two infinite stable sets fully adjacent to a', b' and each other instead. We choose x' from the first stable set for which none of the edges to a', b' have been modified and then y' from the second such that none of the edges to x', a', b' have been modified.

Hence H' admits a fixing operation for the class of C-free graphs. □

If the 3^*-core C of H is 3-connected and not a clique, then in particular it has minimum degree at least 3. This means that we can obtain H from C by repeatedly adding vertices of degree at most $2 < \delta(C)$ or adding pairs of adjacent vertices as in the statement of the lemma above. So starting from Lemma 7 and then repeatedly applying Lemma 8, we obtain the following corollary.

Corollary 9. *Let H be a finite graph with 3^*-core (or 3-core) H' which is 3-connected and not a clique. Then H admits a fixing operation for the class of H'-free graphs. In particular, there exists a strongly H-induced-saturated graph.*

We will also use Lemma 8 below in Section 5.

4 A characterisation of the remaining graphs

The goal of this section is to prove the following result.

Theorem 10. *For any finite graph H on at least 12 vertices, either the graph H or its complement $\overline{H}$ satisfies one of the following statements.*

- *H is a clique.*
- *H is a forest with a unique vertex of maximum degree.*

- *The 2-core of H is $K_{2,p}$ or $K_{1,1,p}$ for some $p \geq 3$.*
- *The 3^*-core of H is 3-connected and not a clique.*

Each of these cases will be handled separately in the proof of Theorem 1.

Before we present the proof of Theorem 10, we give a few auxiliary results. We say a graph H has a *butterfly cut* if we can partition its vertex set as $V(H) = A \sqcup U \sqcup B$ such that there are no edges between A and B, $|U| \leq 2$ and $|A|, |B| \geq 3$.

Observation 11. *If H has a butterfly cut, then $\overline{H}$ contains $K_{3,3}$ as a subgraph.*

Indeed, we can find the copy of $K_{3,3}$ with the two parts of the bipartition contained in A and B respectively.

Lemma 12. *If a finite graph H has at least 12 vertices, then either H or $\overline{H}$ has a 3-connected subgraph on at least 5 vertices.*

Proof. Note that $K_{3,3}$ is a 3-connected subgraph on at least 5 vertices. So if H has a butterfly cut, then we are done by Observation 11. We are also done if H is 3-connected.

So we may assume that there is a cut U_1 of size at most 2 in H and that we can partition $V(H) = A_1 \sqcup U_1 \sqcup B_1$ with $1 \leq |B_1| \leq 2$ such that there are no edges between A_1 and B_1. Since $|A_1| \geq 8$, we may assume $H[A_1]$ is not 3-connected. Again $H[A_1]$ must contain a cut U_2 of size at most 2 and we can assume it contains no butterfly cut. Hence, we can find a partition $A_1 = A_2 \sqcup U_2 \sqcup B_2$ with $1 \leq |B_2| \leq 2$ such that there are no edges between A_2 and B_2 in H. Then $|A_2| \geq 4$. If $|A_2| = 4$, then $|B_1| = |B_2| = 2$ and $\overline{H}[A_2 \cup B_1 \cup B_2]$ contains the complete 3-partite graph $K_{4,2,2}$, which is 3-connected. So we may assume $|A_2| \geq 5$. We are again done if $H[A_2]$ is 3-connected, so we can split once more into $A_2 = A_3 \sqcup U_3 \sqcup B_3$ with $|A_3| \geq 2$ and $|B_3| \geq 1$. Now $\overline{H}[A_3 \cup B_1 \cup B_2 \cup B_3]$ contains the 3-connected subgraph $K_{2,1,1,1}$ (the complete 4-partite graph with part sizes 2, 1, 1 and 1). □

Lemma 13. *If a finite graph H contains an induced subgraph on at least 5 vertices that is 3-connected and not a clique, then the 3^*-core of either H or $\overline{H}$ is 3-connected and not a clique.*

Proof. Let H be a finite graph with an induced subgraph S on at least 5 vertices that is 3-connected and not a clique. The 3^*-core H_{3^*} of H contains S, and is hence not a clique. Suppose that H_{3^*} is not 3-connected. Then there is a partition $V(H_{3^*}) = A \sqcup U \sqcup B$ with $|U| \leq 2$ and no edges between A and B. Since S is 3-connected, we may assume that $V(S) \subseteq A \cup U$. This implies that $\overline{H}[A]$ contains at least one non-edge and that $|A| \geq 3$. We will show that the 3^*-core of $\overline{H}$ is 3-connected.

Since B is part of the 3^*-core of H, it must be the case that $|B| \geq 3$. Thus $\overline{H_{3^*}}[A \cup B]$ is 3-connected and, due to the non-edge in $\overline{H}[A]$, it is not a clique. Consider the set of vertices $R = V(H) \setminus V(H_{3^*})$ that we removed from H to form the 3^*-core of H. We can iteratively add these back in (in clusters of size at most 2) to $\overline{H}[A \cup B]$. When a cluster is added, it has edges to all but at most two of the existing vertices (and in particular degree at least 4 since $|A| + |B| \geq 6$). Hence, $\overline{H}[A \cup B \cup R]$, which is not a clique, is 3-connected and contained in the 3^*-core of $\overline{H}$. Since $|U| \leq 2$, the last two vertices in U cannot form a separate component in the 3^*-core of $\overline{H}$. It follows that the 3^*-core of $\overline{H}$ is 3-connected and not a clique. □

Proof of Theorem 10. Let H be a finite graph on $n \geq 12$ vertices. Let H_0 be the largest 3-connected subgraph of H. By Lemma 12, we may assume the size of H_0 is at least 5 (switching from H to $\overline{H}$ if necessary). If H_0 is not a clique, then we are done by Lemma 13. So we assume that H_0 is a clique. By the maximality of H_0, any $u \notin H_0$ is adjacent to at most 2 vertices in H_0. We first handle the special case where H consists of at most two vertices plus the clique H_0, in which case $|H_0| \geq 10$.

- If there are no additional vertices, then H is a clique, which is one of our possible conclusions.
- Suppose there is one additional vertex u_1. Note that $d(u_1) \leq 2$, since $V(H) = \{u_1\} \sqcup V(H_0)$. Thus $\overline{H}$ is a star (centred at u_1) with at most two isolated vertices. Since the centre of the star is a vertex of unique maximum degree, this is one of the possible conclusions.
- Suppose that there are two additional vertices u_1 and u_2. Both have at most two edges to H_0. The 2-core of $\overline{H}$ equals $K_{2,p}$ or $K_{1,1,p}$ for some $p \geq |H_0| - 4 \geq 6$, depending on whether there is an edge between u_1 and u_2 or not.

We may now assume that $|V(H) \setminus V(H_0)| \geq 3$. Let $A = V(H_0)$ and $B = V(H) \setminus V(H_0)$, and note that $|A| \geq 5$ and $|B| \geq 3$. We will show that $\overline{H}$ has a 3-connected induced subgraph with at least 5 vertices which is not a clique so, by Lemma 13, the 3^*-core of either H or $\overline{H}$ is 3-connected and not a clique.

Since each $b \in B$ has at most 2 edges to A in H, it has at least $|A| - 2$ edges to A in $\overline{H}$. This means that $\overline{H}$ contains a (not necessarily induced) bipartite subgraph $H' = (A, B, E)$ with $|A| + |B| = 12$, with $|A| \geq 5$ and $|B| \geq 3$, and such that each element of B has exactly $|A| - 2$ neighbours in A. A computer search shows that there is indeed a 3-connected subgraph on at least 5 vertices in every such bipartite graph. Clearly, the corresponding vertices induce a 3-connected subgraph of $\overline{H}$ (as this only has more edges than the subgraph of the H') and the subgraph is not a clique as it must contain at least 3 vertices of A and these form an independent set in $\overline{H}$. □

5 The remaining large graphs

We now turn our attention to the graphs H satisfying the second or third property of Theorem 10, after which we will have proved Theorem 1 for all graphs H on at least 12 vertices.

5.1 Fixing operations for $K_{2,p}$

The fixing operation for $K_{2,p}$ is relatively simple.

Lemma 14. *Suppose the 2-core of H is a copy of $K_{2,p}$ where $p \geq 3$. Then H admits a fixing operation for the class of $K_{2,p}$-free graphs.*

Proof. By Lemma 8, we only need to show that there is a fixing operation for $K_{2,p}$: indeed, H can be obtained from $K_{2,p}$ by repeatedly adding vertices of degree at most $1 < \delta(K_{2,p})$.

Let G be a $K_{2,p}$-free graph. Suppose first that xy is an unfixed edge. To fix the edge xy, add an infinite stable set where every vertex is connected to both x and y. This is equivalent to gluing on a copy of $K_{1,1,\infty}$ and any locally finite disturbance which removes the edge xy gives a copy of $K_{2,p}$. It remains to check that we did not create a copy of $K_{2,p}$. Suppose that a newly added vertex v is in a copy of $K_{2,p}$. Any newly added vertex is only adjacent to x and y and so both x and y must be in the copy of $K_{2,p}$. However, then the copy of $K_{2,p}$ contains a triangle, giving a contradiction.

Suppose next that xy is an unfixed non-edge. Add a vertex w which is adjacent to x but not to y, and then add $p-1$ vertices $w_1, \dots, w_{p-1}$ adjacent to both w and y. Finally, we blow up the (new) vertices $w, w_1, \dots, w_{p-1}$ into infinite cliques $W, W_1, \dots, W_{p-1}$. All vertices in W and y are adjacent to all vertices in W_i for $i \in [p-1]$. It is not hard to see that any locally finite disturbance which adds the edge xy creates a copy of $K_{2,p}$. In fact, there is some $w' \in W$ and

$w'_i \in W_i$ for $i \in \{1, \dots, p-1\}$ such that $\{y, w'\} \cup \{x, w'_1, \dots, w'_{p-1}\}$ forms a copy of $K_{2,p}$. It remains to show that we have not introduced a copy of $K_{2,p}$. Let u and v be the two vertices of $K_{2,p}$ with p neighbours, and consider which vertices could be u in a copy of $K_{2,p}$. Since the p neighbours of u form a stable set, $u \notin W_i$ for every i as the neighbours of $w'_i \in W_i$ can be decomposed into 2 cliques and $p \geq 3$. A vertex $w' \in W$ only has enough neighbours that form a stable set if x is among the neighbours and there is a vertex from each W_i, but then we cannot choose a suitable v. The only other way a copy of $K_{2,p}$ could contain one of the added vertices is for u and v to be x and y, but we have not added any common neighbours of x and y. So we did not create any copies of $K_{2,p}$ and indeed we fixed the non-edge. □

5.2 Fixing operation for $K_{1,1,p}$

The edge fixing operation for $K_{1,1,p}$ is more involved and requires an additional assumption about the graphs it is applied to.

Lemma 15. *Suppose the 2-core of H is a copy of $K_{1,1,p}$ where $p \geq 2$. Then H admits a fixing operation for the class of $K_{1,1,p}$-free graphs where the common neighbourhood of any unfixed edge can be decomposed into at most $p-2$ cliques.*

Proof. We maintain throughout that the common neighbourhood of any unfixed edge can be decomposed into at most $p-2$ cliques. For $p = 2$, this means that if the edge uv is unfixed, then $N(u) \cap N(v) = \emptyset$. We discuss how to define a fixing operation for $K_{1,1,p}$ then explain how to extend to graphs with this as a 2-core.

We first handle the case $p = 2$, as it is simpler and serves as a warm-up to the case $p \geq 3$. First note that any edge in an infinite clique is fixed, as disturbing it will result in an induced copy of $K_{1,1,2}$ even if this is part of an arbitrary locally finite disturbance. Given an unfixed edge $uv \in E$, we fix it by adding a countably infinite clique completely adjacent to both u and v. Since u and v have no common neighbours before the fixing operation, this does not create an induced copy of $K_{1,1,2}$. Given an unfixed non-edge $uv \notin E$, we fix it by adding a countably infinite stable set completely adjacent to both u and v. Note that every new edge, though unfixed, is incident to a vertex of degree 2 which is not in a triangle, so the endpoints have no common neighbour.

Assume now $p \geq 3$. We first explain how to fix an edge $uv \in E$. We fix an infinite tree T in which a particular vertex r has degree 1 and all other vertices have degree $p-1$. We blow up all vertices except for r into an infinite clique. We blow up r into an edge and glue this onto uv.

We first show that this fix keeps the graph $K_{1,1,p}$-free. For an edge xy, when x and y correspond to different vertices of T, their common neighbourhood can in be decomposed into a single clique. In particular, they cannot be used as the pair of vertices of degree $p+1$ of any copy of $K_{1,1,p}$.

When x and y correspond to the same vertex of T, their common neighbourhood can be decomposed into $p-1$ cliques; this holds for $x = u, y = v$ using our assumption that the common neighbourhood of u, v could originally be covered by at most $p-2$ cliques, and for other pairs using the degree of T. This shows that we did not create any copy of $K_{1,1,p}$ since no pair among the new vertices, u and v can be used as the vertices of degree $p+1$.

Next, we show that all edges x and y with x, y corresponding to the same vertex of T have been fixed (this includes the edge uv). Consider a locally finite disturbance in which xy gets removed. Let s_p be the vertex in T that x, y correspond to and let $t \neq r$ be a neighbour of s_p in T. Let $s_1, \dots, s_{p-2}$ denote the other neighbours of t in T. There must be infinitely many

vertices in the blow-up of t that are still adjacent to x and y after the disturbance. We may pick any two of them, say a and b. Next, we select for each $i \in [p-2]$, a vertex w_i resulting from the blow-up of s_i, such that $\{a\} \cup \{b\} \cup \{w_1, \dots, w_{p-2}, x, y\}$ forms the desired copy of $K_{1,1,p}$. Note that we can indeed do so since each of a, b, x, y prohibits only finitely many options.

The new unfixed edges are the ones which correspond to different vertices of T, and indeed have the common neighbourhood property that we wish to maintain, since their common neighbourhood can be decomposed into a single clique (so at most $p-2$ since $p \geq 3$).

The fixing operation for a non-edge is far simpler. Indeed, let u, v be an unfixed non-edge. We add an infinite stable set with neighbourhood $\{u, v\}$. This ensures that any locally finite disturbance containing the pair u, v results in an induced $K_{1,1,p}$. Note that the graph resulting from the fixing operation is still $K_{1,1,p}$-free and the common neighbourhood of every edge is either unaffected by the fixing or is empty, so the fixing operation for non-edges maintains all the desired properties.

To extend the fixing operations above to graphs for which the 2-core is $K_{1,1,p}$ for $p \geq 2$, we unfortunately cannot directly apply Lemma 8 (due to the property about the common neighbourhoods). However, we may repeat the proof and note that by gluing infinite stable sets fully connected to at most one vertex, we not only maintain the property of being $K_{1,1,p}$-free, but also the property that unfixed edges have no stable set of size $p-1$ in their common neighbourhood. □

5.3 Forests with a vertex of unique maximum degree

Let F be a finite forest which has a unique vertex of maximum degree $d > 1$ that we denote by v. One easy way of preventing a graph G from containing a copy of F is to ensure it has maximum degree strictly less than d. Unfortunately, removing an edge from G is not going to increase the maximum degree and create a copy of F, so we need to be slightly smarter. Since F is a forest, the neighbourhood of any vertex is a stable set and, instead of simply restricting the maximum degree, we can ensure that no vertex in G has a neighbourhood containing a stable set of size d.

We define the graph K^p_∞ as the infinite graph with vertex set $\mathbb{Z}^p$ and an edge between a vertex $u = (u_1, \dots, u_p)$ and vertex $v = (v_1, \dots, v_p)$ if and only if they disagree in exactly one coordinate i.e. there is a unique i such that $u_i \neq v_i$. The maximum degree is unbounded, but the neighbourhood of any vertex can be decomposed into p cliques, one for each coordinate, and the largest stable set in the neighbourhood of any vertex is of size p. We claim that K^{d-1}_∞ is strongly F-induced-saturated.

Lemma 16. *Let F be a finite forest with a unique vertex of maximum degree d. The graph K^{d-1}_∞ is strongly F-induced-saturated.*

Proof. As argued above, K^{d-1}_∞ does not contain F since the largest stable set in any neighbourhood is smaller than the maximum degree in F. It remains to show that making any locally finite edit creates an induced copy of F.

We first consider the case in which F is a tree. Let v be the unique vertex of maximum degree and suppose we remove an edge. Without loss of generality, let the edge be from $(1, 0, \dots, 0)$ to $(2, 0, \dots, 0)$. There must exist an integer i_v such that $(i_v, 0, \dots, 0)$ is still connected to both $(1, 0, \dots, 0)$ and $(2, 0, \dots, 0)$. Here we will embed our vertex v. Similarly, there are integers $j_3, \dots, j_d$ such that after the locally finite edit, there are still edges from $(i_v, 0, \dots, 0)$ to

$$(i_v, j_3, 0, \dots, 0), (i_v, 0, j_4, \dots, 0), \dots, (i_v, 0, 0, \dots, j_d)$$

and the vertices displayed above together with $(1, 0, \ldots, 0), (2, 0, \ldots, 0)$ form a stable set. We embed the d neighbours of v into this stable set of size d.

We now explore the tree, heading out from v and assigning each vertex to a vertex in $\mathbb{Z}^{d-1}$ as we go. Suppose we have reached the vertex $u = (u_1, \ldots, u_{d-1})$ which disagrees with the previous vertex in coordinate i. There are at most $d' \leq d - 1$ neighbours of u, say $w_0, w_1, \ldots, w_{d'-1}$, and we have already seen one of them, say w_0. We must assign to each w_j with $j \in [d]$ a vertex in $\mathbb{Z}^{d-1}$. First choose $d' - 1$ distinct coordinates $p_1, \ldots, p_{d'-1}$, none of which are equal to i, and set w_j to be equal to u except in position p_j. For position p_j, we choose an integer which ensures that w_j is not adjacent to any of the vertices we have embedded except for u. Since at each step, there are only finitely many "bad integers" to choose out of infinitely many, we can always select an appropriate one.

Adding an edge can be handled similarly: we embed v into an endpoint x of the edge and note that the added edge again makes it possible to embed the d neighbours of v as a stable set in the neighbourhood of x.

For arbitrary forests F, let T be the connected component of F which contains the unique vertex of maximum degree. We showed above that any locally finite edit creates a copy of T. All other connected components of F have maximum degree at most $d - 1$. After embedding T, it is straightforward to embed the other components in K_∞^{d-1} in a way that there are no edges to the vertices used for T. □

5.4 Scheduling the fixes

The following lemma is straightforward but important. A key part of the proof is that any bad pair is fixed in a finite number of steps. This ensures every bad pair is fixed in the limit.

Lemma 17. *Suppose that a finite graph H admits a fixing operation for a non-empty class of H-free graphs. Then there exists a strongly H-induced-saturated graph.*

Proof. We define a sequence of graphs $G_1 \subseteq G_2 \subseteq \ldots$ in $\mathcal{F}$ and we define the graph G as $\cup_{i=1}^{\infty} G_i$.

Let $G_1 \in \mathcal{F}$. We keep track of a set F of pairs that need to be fixed with a priority order (described later in this proof), which is initialised as the (countable) set of unfixed pairs in G_1. Suppose we have defined $G_i \in \mathcal{F}$ for some integer $i \geq 1$. If all pairs $xy \in G_i$ are fixed, we may set $G_t = G_i$ for all $t \geq i$ and $G = G_i$ is our desired graph. Otherwise, let $xy \in G_i$ be an unfixed pair of highest priority in F. We let G_{i+1} be the graph obtained from applying the fixing operation for this pair.

We now describe how to define the priority order in such way that every bad pair is eventually fixed. For any step i, let B_i be the collection of bad pairs in G_i that do not exist in G_{i-1}. We fix a bijection $f_i : \mathbb{N} \to B_i$. We order the fixes $f_i(j)$ first by increasing $i + j$, and then by increasing j. That is, $f_i(j) < f_{i'}(j')$ if $i + j < i' + j'$, or if $i + j = i' + j'$ and $j < j'$. The first seven elements are given by

$$f_1(1), f_2(1), f_1(2), f_3(1), f_2(2), f_1(3), f_4(1).$$

This total order will indicate the priority of the fixes: at any point, we perform next the fix of highest priority among those that involve a pair of vertices both existing in the current graph. Note that if a high priority fix involves a vertex that is created late in the graph sequence, it is possible that "lower priority" fixes have been implemented first. In a sense, priority had just been pre-allocated for it. However, if a bad pair (u, v) is such that both u and v belong to $V(G_i)$, and $f_i(k) = uv$, then it will be fixed before step $(i + k)^2$.

By definition of a fixing operation for xy, if $G_i \in \mathcal{F}$, then $G_{i+1} \in \mathcal{F}$ and the pair xy is fixed in G_{i+1}. Moreover, any pair which was fixed in G_i remains fixed in G_{i+1} since G_i is an induced subgraph of G_{i+1}. By putting the priority order on F, we ensure that for any pair xy in G, there is a finite i such that xy is a fixed pair in G_i, so in particular xy is a fixed pair in G.

Since no copy of H can be created in G_i for all i, G is also H-free. This proves that G has the desired properties. □

6 The remaining small graphs

In this section we consider graphs on at most 11 vertices. While many of these graphs can be handled using the lemmas and theorems above, there are still several small graphs which we still need to consider. To this end, we introduce two new constructions to handle families of graphs (in Sections 6.1 and 6.2), and a method for checking for fixing operations with the help of a computer (in Section 6.3). This still leaves a total of eight graphs, four graphs and their complements, which we handle individually with two more constructions at the end of this section.

Before we give the new constructions and the method to find fixing operations, let us briefly summarise the steps in the computer search. For each graph G on at most 11 vertices, we check if any of the following hold.

1. The graph H is a non-empty forest with a unique vertex of maximum degree (Lemma 16).
2. The 2-core of H is a copy $K_{2,p}$ with $p \geq 3$ (Lemma 14).
3. The 2-core of H is a copy of $K_{1,1,p}$ where $p \geq 2$ (Lemma 15).
4. The 3-core of H is 3-connected and not a clique (Corollary 9).
5. The $(1,1)$-core of H is a copy of P_4 or the bull graph (Theorem 20).
6. The graph H is "close to" a permutation graph (Theorem 19).
7. The 2-core, 3-core, 2-edge-core or 2-non-edge-core have fixing operations (see Section 6.3).

If none of these hold for H, we check if any of them hold for its complement $\overline{H}$. As mentioned earlier, these checks leave just eight graphs, `E?qw`, `F?S|w`, `F?q|w` and `F?q w`, and their complements, which we resolve in Sections 6.4 and 6.5.

The code used for this computer check is attached to the arXiv submission.

6.1 The up and right graph

We start with a construction which we call the *up and right graph*, which handles all graphs which are "close to" a permutation graph. A graph G on the vertices $\{v_1, \ldots, v_n\}$ is a *permutation graph* if there is a permutation $\sigma \in S_n$ such that, for every $i < j$, the edge $v_i v_j$ is present if and only if $\sigma(i) < \sigma(j)$. A graph is *close to* a permutation graph if it is not a permutation graph, but adding or removing any edge creates a permutation graph.

The *up and right graph* is the graph on $\mathbb{Q} \times \mathbb{Q}$ where the vertex (p, q) is connected to (s, t) if and only if one of the following holds

- $(q + r\sqrt{2}) < (s + t\sqrt{2})$ and $(q - r\sqrt{2}) < (s - t\sqrt{2})$
- $(q + r\sqrt{2}) > (s + t\sqrt{2})$ and $(q - r\sqrt{2}) > (s - t\sqrt{2})$

That is, two vertices are connected if one of the vertices is up and right of the other after the linear transformation $(q, r) \mapsto (q + r\sqrt{2}, q - r\sqrt{2})$ has been applied. Our motivation for this is to ensure that no two vertices can differ in exactly one coordinate. Indeed, if

$$q \pm r\sqrt{2} = s \pm t\sqrt{2}$$

where $q, r, s, t \in \mathbb{Q}$, then we must have $(q, r) = (s, t)$. This means that given n distinct vertices $v_i = (q_i, r_i)$ for $i \in [n]$, we can order them such that

$$q_i + r_i\sqrt{2} < q_j + r_j\sqrt{2}$$

whenever $i < j$.

Similarly, there is also an order based on the other coordinate, and there is a unique permutation $\sigma \in S_n$ such that

$$q_i - r_i\sqrt{2} < q_j - r_j\sqrt{2}$$

whenever $\sigma(i) < \sigma(j)$. With this notation, there is an edge between v_i and v_j, where $i < j$, exactly when $\sigma(i) < \sigma(j)$. This means that any induced subgraph is a permutation graph.

The following lemma shows that every permutation graph can be found as an induced subgraph of the up and right graph, giving us a characterisation of exactly which finite graphs appear as induced subgraphs.

Lemma 18. *A graph H is an induced subgraph of the up and right graph if and only if it is a permutation graph.*

Proof. We already argued that any induced subgraph is a permutation graph, so it only remains to show how to find a copy of any given permutation graph. Let σ be a permutation that defines the permutation graph H and consider the n points

$$\tilde{v}_i = \left(\frac{i + \sigma(i)}{2}, \frac{i - \sigma(i)}{2\sqrt{2}} \right) \text{ for } i \in [n].$$

The coordinates are chosen so that

$$\frac{i + \sigma(i)}{2} + \frac{i - \sigma(i)}{2\sqrt{2}}\sqrt{2} = i$$

and

$$\frac{i + \sigma(i)}{2} - \frac{i - \sigma(i)}{2\sqrt{2}}\sqrt{2} = \sigma(i).$$

Ideally, we would take the $\tilde{v}_i$ as the vertices in the up and right graph, but the $\tilde{v}_i$ are not necessarily rational. However, the rational points are dense in $\mathbb{R}^2$ and we can choose vertices of the up and right graph which are sufficiently close to the points $\tilde{v}_i$ to keep the expected adjacencies. This provides a subgraph isomorphic to H. □

Given this lemma is it not hard to prove the following theorem classifying the graphs which are strongly saturating in the up and right graph.

Theorem 19. *The up and right graph is strongly H-induced-saturated if and only if the following hold.*

1. *H is not a permutation graph.*
2. *There exists an edge e such that $H - e$ is a permutation graph.*

3. *There exists a non-edge $\overline{e}$ such that $H + \overline{e}$ is a permutation graph.*

Proof. It is easy to see these conditions are necessary. Indeed, the up and right graph cannot contain a copy of H so, by Lemma 18, H is a not a permutation graph. Adding an edge e to the up and right graph must create a copy of H and e corresponds to some edge e' of this copy of H. This means there is an induced copy of $H - e'$ in the up and right graph and $H - e'$ is a permutation graph. Similarly, there must be a non-edge $\overline{e}$ of H such that $H + \overline{e}$ is a permutation graph.

We now show that the conditions are sufficient. Since H is not a permutation graph there is no copy of H in the up and right graph, and it remains to show that making any locally finite edit creates a copy of H. Let us first consider adding a single edge. By assumption there is an edge $e = v_i v_j$ of H such that $H - e$ is a permutation graph, and we claim that this is enough to guarantee the existence of a copy of H after adding an edge.

Suppose $H - e$ is a permutation graph and let f be a non-edge from (q, r) to (s, t) in the up and right graph. The up and right graph is vertex transitive so we can assume that $(q, r) = (0, 0)$. We now repeat the construction from Lemma 18, but with the addition of extra scaling factors. Define

$$k' = (k - i)\frac{s + t\sqrt{2}}{j - i} \text{ and } \sigma'(k) = (\sigma(k) - \sigma(i))\frac{s - t\sqrt{2}}{\sigma(j) - \sigma(i)},$$

and let

$$\tilde{v}_k = \left(\frac{k' + \sigma'(k)}{2}, \frac{k' - \sigma'(k)}{2\sqrt{2}}\right).$$

Note that by our choice of the scaling factors, we have $\tilde{v}_i = (0, 0)$ and $\tilde{v}_j = (s, t)$, so that e corresponds to f. Since e is not an edge in the permutation graph, $j - i$ and $\sigma(j) - \sigma(i)$ have opposite signs. Likewise, since f is not an edge in the up and right graph, $s + t\sqrt{2}$ and $s - t\sqrt{2}$ also have opposite signs, and the scaling factors both have the same sign. Hence, if the $\tilde{v}_k$ were vertices in the up and right graph, we would have found an induced subgraph isomorphic to $H - e$ and we would be done. Instead, we may need to perturb the $\tilde{v}_k$ by a small amount for $k \neq i, j$ to make sure that they are rational.

Using an almost identical argument one can show that removing an edge creates a copy of H provided that there is a non-edge $\overline{e}$ of H such that $H + \overline{e}$ is a permutation graph.

So far we have only shown that the graph is induced-saturated for H and contains a copy of H when a single disturbance is made, but it is not hard to adapt the proof to handle locally finite disturbances. We begin as before, setting v_i and v_j as $\tilde{v}_i$ and $\tilde{v}_j$ respectively. We then choose each v_k in turn. At each step, there are infinitely many choices for v_k that are sufficiently close to $\tilde{v}_k$. Since the edit is locally finite and we have only picked finitely many vertices so far, we can choose such a vertex (in fact, all but finitely many work) for which none of the adjacencies to the vertices we have already chosen have been altered. □

6.2 The torero graph

In this section, we give a construction which is strongly-H-induced-saturated whenever the $(1, 1)$-core is a copy of the bull graph (see Figure 6) or P_4.

The *torero graph* has vertex set $\mathbb{Q} \cap (0, 1)$ and there is an edge from x to y if and only if $x + y > 1$.

Theorem 20. *The torero graph is strongly H-induced-saturated whenever the $(1, 1)$-core of H is a copy of the bull graph or P_4.*

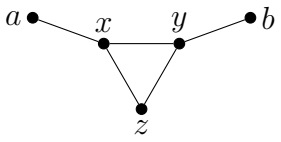
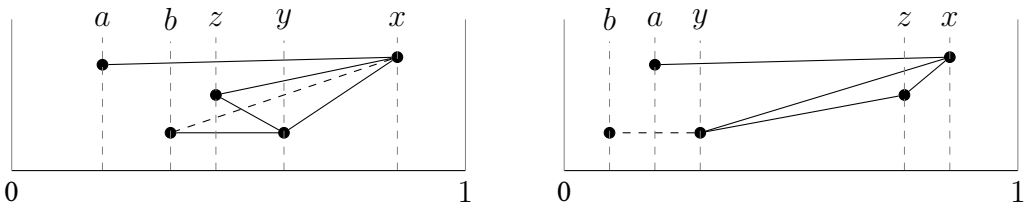


Figure 6: The bull graph.



(a) (b)

Figure 7: We show in (a) how to create a bull when the edge bx is removed and we show in (b) how to create a bull when by is added.

Proof. We prove this theorem in three parts. First, we show that there is no induced copy of P_4 in the torero graph (and hence no copy of the bull graph). Then we show that any locally finite modification of the torero graph contains an induced copy of the bull graph (and hence an induced copy of P_4). Finally, we observe that the torero graph is strongly-H-induced-saturated when the $(1,1)$-core is a copy of the bull graph or P_4.

Label the vertices of P_4 by $v_1, \dots, v_4$ so that the edges are v_1v_2, v_2v_3 and v_3v_4. Since v_1v_2 is an edge and v_1v_3 is not an edge, $v_1 + v_2 > 1 > v_1 + v_3$. In particular, $v_2 > v_3$. However, since v_3v_4 is an edge and v_2v_4 is not an edge, we have $v_2 < v_3$, a contradiction.

Suppose that we remove an edge bx from the torero graph where $b < x$. Since this is normally an edge, we have that $b + x > 1$. There is therefore some $y \in \mathbb{Q}$ such that $b < y < x$ and $b + y > 1$. Now choose a and z such that $a + y < 1 < a + x$ and $z + b < 1 < z + y$, which is possible since $b < y < x$. This gives an induced copy of the bull when we remove a single edge from the torero graph. Since at each step we are choosing any rational in an interval, of which there are infinitely many, we can always choose a vertex such that no edge/non-edge to any of the preceding vertices has been disturbed. This means that we can also manage all locally finite edits. Adding an edge follows a similar argument where the new edge forms the edge by in the bull. Diagrams of both cases can be seen in Figure 7.

Finally, we explain how to handle any graph H whose (1,1)-core is a copy of the bull graph or P_4. Since the torero graph has no induced copy of P_4, it also has no copy of H. To embed a copy of H after a locally finite disturbance, we first embed a copy of the bull graph or P_4 as appropriate. The graph H may be obtained from either the bull or P_4 by iteratively adding vertices connected to none of the preceding vertices, or to all of them. Given any set of vertices, there are infinitely many vertices which are connected to every vertex in the set (those sufficiently close to 1) and infinitely many vertices which are connected to none of the vertices (those sufficiently close to 0), and we can find a copy of H by iteratively choosing such vertices. □

6.3 Computer check for gatekeepers

Let us say that a 2-cut $\{u, v\}$ of G is an *edge-2-cut* if $\{u, v\}$ is an edge in G, and otherwise we say it is a *non-edge-2-cut*.

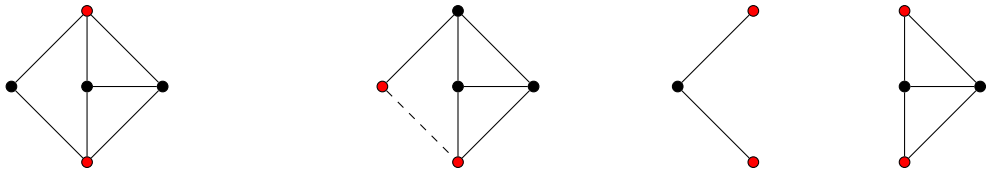
Let H be a 2-connected graph and let $e = xy$ be an edge of H. We show that if there is no non-edge-2-cut $\{u, v\}$ of H such that a component of $H - \{u, v\}$ is an induced subgraph of $H - e$, then e is a gatekeeper. Suppose towards a contradiction that the edge $e = xy$ is not a gatekeeper. Then there is some graph G which does not contain a copy of H such that gluing $H - e$ to G by identifying x and y with the endpoints of a non-edge of G creates a copy H' of H. The vertices x and y are a 2-cut of H' and removing them splits H' into components, say, $C_1, \dots, C_r$. At least one of these components is disjoint from G, else H' would be contained entirely in G. Let u and v be the vertices from H that correspond to x and y in H'. Then the component contained in $H' - \{x, y\}$ corresponds to a component of $H - \{u, v\}$ which is an induced subgraph of $H - e$.

In fact, we can strengthen this condition slightly by including how a component of $H - \{u, v\}$ connects to the vertices $\{u, v\}$, and this is what we use in practice. That is, for each non-edge-2-cut $\{u, v\}$ which splits H into components $C_1, \dots, C_r$ (say), we check if there is a copy of $H[V(C_i) \cup \{u, v\}]$ in $H - e$ where the vertices $\{u, v\}$ correspond to $\{x, y\}$ (either way round). An example is given in Figure 8.

For the constructed graph to be *strongly-H-induced-saturated* it is not sufficient to simply glue on a copy of $H - e$, even when e is a gatekeeper. Instead, we will replace each vertex in $H - e$ except for x and y by either infinite cliques or stable sets. As seen in the proof of Lemma 3, this can be done if neither vertex is a true twin or if neither vertex is a false twin, and we check for this condition before we check if a particular edge or non-edge is a gatekeeper using the above method.

Let us summarise by describing the implementation. First, the code checks that the graph is 2-connected and not a complete graph. Then all of the 2-cuts are enumerated and split into edge-2-cuts and non-edge-2-cuts. For each edge-2-cut $e = xy$, we check that x and y do not have twins of differing types, and we move onto the next edge-2-cut if they do. We then replace e by a "red" edge, which serves to colour the two vertices x and y. We loop over the non-edge-2-cuts and, for each non-edge-2-cut $\{u, v\}$, we find the connected components of $H - \{u, v\}$. For each connected component, C, we form the subgraph $H[V(C) \cup \{u, v\}]$ and add in a red edge between u and v. If this subgraph is isomorphic (including edge colours) to a subgraph of H, then we move onto another edge-2-cut. If we did not find such a subgraph for any choice of non-edge-2-cut and connected component, then we have found the suitable fixing operation and we move onto non-edge-2-cuts, which are handled similarly.

This gives us a method to look for fixing operations in a given 2-connected graph, but we can combine this with Lemma 8 to cover many more graphs. Indeed, suppose that the k-core of H, which we denote H', has a fixing operation (potentially found by the method above) for the class of H'-free graphs. Then, by repeatedly applying Lemma 8, the graph H also admits a fixing operation for the class of H'-free graphs and, in particular, is strongly saturating. Using the third and fourth parts of Lemma 8, we can also generalise this to what we call the 2-edge-core and the 2-non-edge-core. The *2-edge-core* (resp. *2-non-edge-core*) of a graph is formed by repeatedly removing vertices of degree less than 2 and vertices of degree 2 whose neighbours are adjacent (resp. non-adjacent). The 2-edge-core of a graph has minimum degree at least 2 and no vertex v with degree 2 whose neighbours are adjacent. Using Lemma 8 repeatedly, it follows that the graph H admits a fixing operation if its 2-edge-core or 2-non-edge-core admits a fixing operation, as required for check 7 above.

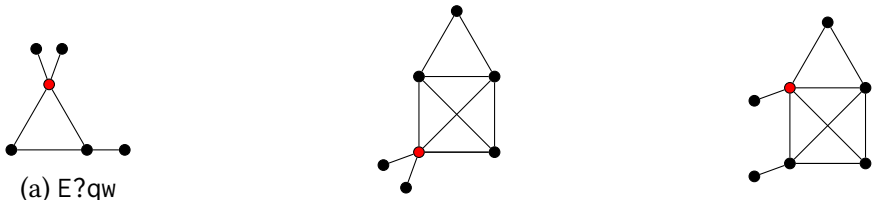


(a) (b) (c)

Figure 8: (a) The graph Dr[with its only 2-cut highlighted in red. There are no 2-cuts which are edges, so every non-edge is a gatekeeper. (b) The graph with the edge xy removed, with x and y highlighted in red. (c) The two fragments of the graph created by taking the components of the only 2-cut and adding back in the 2-cut. Observe that there is no copy of either of the fragments in (b) with matching vertex colours, which means that xy is a gatekeeper.

6.4 The rational geometric graph

The *rational geometric graph* is the graph on $\mathbb{Q}$ where there is an edge between q and r if and only if $|q - r| < \pi$. Our main interest in this construction is to handle the three problematic small graphs shown in Figure 9. First, we make the easy observation that the neighbourhood of any vertex v in the rational geometric graph can be partitioned into two cliques (where there may be edges between the cliques). In particular, this immediately shows that none of the graphs in Figure 9 are induced subgraphs of the rational geometric graph.



(a) E?qw (b) F?rLw (c) F?S|w

Figure 9: The three problematic graphs which we handle using the rational geometric graph. The vertices highlighted in red have neighbourhoods which cannot be split into two cliques.

To handle these graphs, what remains to show is that making any locally finite edit to the rational geometric graph creates an induced copy. Let us first consider making just a single change, either swapping an edge to a non-edge or a non-edge to an edge. By the symmetry of the rational geometric graph, we can assume the edge/non-edge is from 0 to some $r > 0$, where $r < \pi$ if it is an edge and $r > \pi$ if it is a non-edge. It is straightforward to find a sequence of intervals from which we can choose the vertices of our small graphs, but in the interest of conciseness we only sketch the constructions in Figure 10.

Again, since we may choose from infinitely many vertices at any stage, we can always choose one such that none of the edges/non-edges to the proceeding vertices have been altered.

6.5 The final graph

There is one final graph we need to handle, shown in Figure 11.

We first give an auxiliary construction and then blow this up to allow for locally finite disturbances. Let G be the graph with vertex set $\mathbb{Z}^3$. We join the vertex (i, j, k) to (a, b, c) in G if they agree in at least one coordinate, i.e. $i = a$, $j = b$ or $k = c$. There are two different kinds of edges uv in G: those where u and v agree in exactly one coordinate and those where u and v agree in exactly two coordinates. There is one type of non-edge.

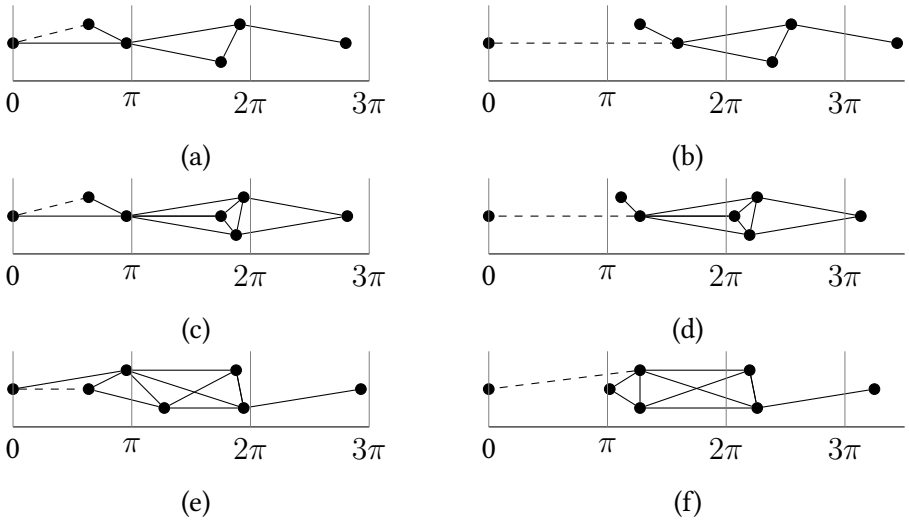


Figure 10: The left-hand side shows the constructions when removing an edge on the dotted line and the right-hand side shows the constructions when adding the edge indicated by the dotted line.

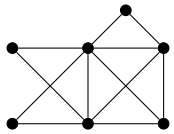


Figure 11: The final graph `F?q~w`.

The graph G' that we are interested in is obtained by blowing up each vertex of G into an infinite clique (replacing edges by complete bipartite graphs). We can split the neighbours of any vertex of G' into three cliques based on which coordinate they agree on in G, which shows that G' does not contain a copy of H.

Next, we show that making any locally finite disturbance creates a copy of H. We again first show that a single disturbance in G results in a copy of H in Figure 12.

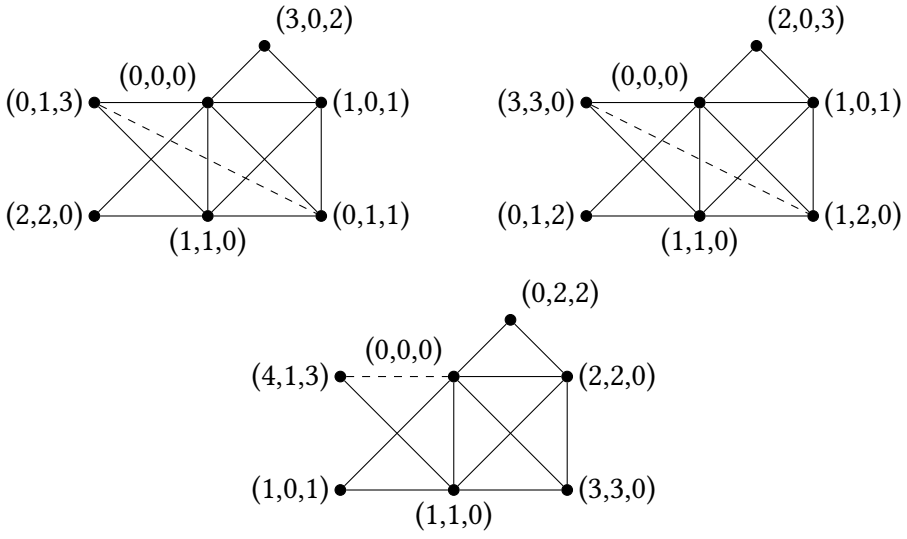


Figure 12: Examples of how to embed H into a copy of G when an edge of G has been disturbed. The dotted lines represent the location of the edge which was removed (top) or added (bottom).

Suppose now that $v_1'v_2'$ is disturbed in a locally finite disturbance of G' where v_1' and v_2' correspond to different vertices v_1 and v_2 in G. Then a copy of H will still be created. The argument for this is similar to previous constructions: if $v_1, \ldots, v_k$ are the vertices in G on which a copy of H is created after disturbing v_1v_2 in G, then we may iteratively choose v_i' corresponding to v_i such that the edges/non-edges from v_i' to $v_1', \ldots, v_{i-1}'$ have not been adjusted.

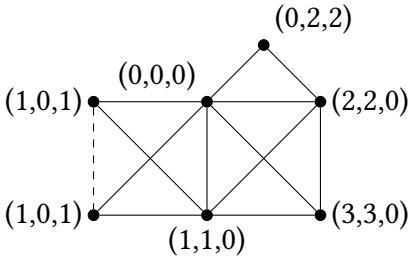


Figure 13: The construction shows how to obtain a copy of H when an edge is removed between two vertices which come from the same vertex of G (in this case, $(1, 0, 1)$).

We have also introduced a new type of edge in G': an edge $u'u''$ where both u' and u'' come from the infinite clique that corresponds to a single vertex u of G. We indicate in Figure 13 how to obtain a copy of H in a locally finite disturbance that only affects edges of this type.

6.6 Proof of Theorem 1

We now have all the components required to prove Theorem 1.

Proof of Theorem 1. Suppose that H is a finite graph on at least 12 vertices which is not a clique or stable set. By Theorem 10, either the graph H or its complement $\overline{H}$ satisfies one of the following statements:

1. H is a forest with a unique vertex of maximum degree,
2. the 2-core of H is $K_{2,p}$ for $p \geq 3$,
3. the 2-core of H is $K_{1,1,p}$ for $p \geq 3$, or
4. the 3^*-core of H is 3-connected and not a clique.

In the first and fourth cases, there exists a strongly H-induced-saturated graph by Lemma 16 and Corollary 9, respectively. By Lemma 14, any graph in the second case admits a fixing operation for the class of $K_{2,p}$-free graphs. By Lemma 15, any graph in the third case admits a fixing operation for a particular class of $K_{1,1,p}$-free graphs. Applying Lemma 17 in each of these cases, and noting that the complement of any strongly H-induced-saturated graph is a strongly $\overline{H}$-induced-saturated graph, completes the proof for any H on at least 12 vertices.

For graphs on at most 11 vertices, our computer search identifies whether each graph H or its complement satisfies any of the following conditions.

1. The graph H is a non-empty forest with a unique vertex of maximum degree.
2. The 2-core of H is a copy $K_{2,p}$ with $p \geq 3$.
3. The 2-core of H is a copy of $K_{1,1,p}$ where $p \geq 2$.
4. The 3-core of H is 3-connected and not a clique.
5. The $(1, 1)$-core of H is a copy of P_4 or the bull graph.

6. The graph H is close to a permutation graph.
7. The 2-core, 3-core, 2-edge-core or 2-non-edge-core have fixing operations.

The first four cases follow for the same reasons they did for graphs on at least 12 vertices. The fifth and sixth cases are resolved by Theorems 20 and 19, respectively. The last case is considered in Section 6.3. There are 8 graphs (`E?qw`, `F?S|w`, `F?q|w` and `F?q w`, and their complements) which do not fall into one of the above cases. The first six of these are resolved in Section 6.4, and the final pair is resolved in Section 6.5. □

7 Conclusion

Even though it is still widely open which finite graphs H admit a finite H-induced-saturated graph, we obtained a full characterisation in this paper when allowing countable H-induced-saturated graph instead. We showed that a finite graph H admits a countable H-induced-saturated graph if and only if H is not a clique or stable set. We also showed the characterisation remains the same when asking for a much stronger notion of saturation.

Many interesting directions remain open. For example, what happens when H is infinite? What about other structures than graphs, for example, can a similar (or partial) characterisation be obtained for k-uniform hypergraphs, coloured graphs, matroids, posets or directed graphs? In some of these cases, an appropriate notion of "disturbance" has to be chosen. As an explicit example, for tournaments it seems natural to "flip the direction of edges".

Problem 21. *For which finite tournaments H, does there exist a countable tournament G that does not contain H as subtournament, yet any tournament obtained from G by changing the direction of a single arc does contain H?*

For example, when H is a finite transitive tournament, then it can never be avoided, but when H is a directed triangle then there are even finite examples for G. It would also be interesting to see if the answer is the same for the "locally finite disturbance" variant of this question.

Another direction is to show the existence (or non-existence) of graphs that are simultaneously (strongly) H-induced-saturated for all graphs H in a specific class of graphs. We gave such a result for the class of forests which have a unique vertex of maximum degree, and also characterised in Theorem 19 for which graphs H our "up-and-right graph" is strongly H-induced-saturated. What about the class of finite co-graphs, or finite graphs of bounded twinwidth?

References

[1] M. Axenovich and M. Csikós. Induced saturation of graphs. *Discrete Mathematics*, **342**(4):1195–1212, 2019.

[2] S. Behrens, C. Erbes, M. Santana, D. Yager and E. Yeager. Graphs with induced saturation number zero. *Electronic Journal of Combinatorics*, **23**(1):#P1.54, 2016.

[3] M. Bonamy, C. Groenland, T. Johnston, N. Morrison and A. Scott. Induced saturation for P_5. https://tomjohnston.co.uk/blog/2020-05-22-induced-saturation-for-paths.html, 2020.

[4] P. J. Cameron and J. Nešetřil. Homomorphism-homogeneous relational structures. *Combinatorics, Probability and Computing*, **15**(1-2):91–103, 2006.

[5] G. L. Cherlin. *The Classification of Countable Homogeneous Directed Graphs and Countable Homogeneous n-tournaments*, volume 621. American Mathematical Soc., 1998.

[6] E.-K. Cho, I. Choi and B. Park. On induced saturation for paths. *European Journal of Combinatorics*, **91**:103204, 2021.

[7] B. L. Currie, J. R. Faudree, R. J. Faudree and J. R. Schmitt. A survey of minimum saturated graphs. *Electronic Journal of Combinatorics*, **DS19**:36, 2021.

[8] R. Diestel, R. Hahn and W. Vogler. Some remarks on universal graphs. *Combinatorica*, **5**:283–293, 1985.

[9] R. Diestel and D. Kühn. A universal planar graph under the minor relation. *Journal of Graph Theory*, **32**(2):191–206, 1999.

[10] V. Dvořák. P_n-induced-saturated graphs exist for all $n \geq 6$. *Electronic Journal of Combinatorics*, **27**(4):#P4.43, 2020.

[11] P. Erdős, A. Hajnal and J. W. Moon. A problem in graph theory. *American Mathematical Monthly*, **71**:1107–1110, 1964.

[12] X. Fan, S. Hajebi, S. Hajebi and S. Spirkl. Halfway to induced saturation for even cycles. *arXiv preprint arXiv:2505.24100*, 2025.

[13] P. Komjáth, A. H. Mekler and J. Pach. Some universal graphs. *Israel Journal of Mathematics*, **64**(2):158–168, 1988.

[14] A. H. Lachlan and R. E. Woodrow. Countable ultrahomogeneous undirected graphs. *Transactions of the American Mathematical Society*, **262**(1):51–94, 1980.

[15] R. R. Martin and J. J. Smith. Induced saturation number. *Discrete Mathematics*, **312**(21):3096–3106, 2012.

[16] J. Pach. A problem of Ulam on planar graphs. *European Journal of Combinatorics*, **2**(4):357–361, 1981.

[17] R. Rado. Universal graphs and universal functions. *Acta Arithmetica*, **9**:331–340, 1964.

[18] E. Räty. Induced saturation of P_6. *Discrete Mathematics*, **343**(1):111641, 2020.

[19] J. H. Schmerl. Countable homogeneous partially ordered sets. *Algebra universalis*, **9**(1):317–321, 1979.

[20] C. M. Tennenhouse. Induced subgraph saturated graphs. *Theory and Applications of Graphs*, **3**(2), 2016.