Exhaustive generation of k-critical $\mathcal{H}$-free graphs

Jan Goedgebeur* Oliver Schaudt†

Abstract

We describe an algorithm for generating all k-critical $\mathcal{H}$-free graphs, based on a method of Hoàng et al. Using this algorithm, we prove that there are only finitely many 4-critical (P_7, C_k)-free graphs, for both $k = 4$ and $k = 5$. We also show that there are only finitely many 4-critical (P_8, C_4)-free graphs. For each case of these cases we also give the complete lists of critical graphs and vertex-critical graphs. These results generalize previous work by Hell and Huang, and yield certifying algorithms for the 3-colorability problem in the respective classes.

Moreover, we prove that for every t, the class of 4-critical planar P_t-free graphs is finite. We also determine all 52 4-critical planar P_7-free graphs.

We also prove that every P_{11}-free graph of girth at least five is 3-colorable, and show that this is best possible by determining the smallest 4-chromatic P_{12}-free graph of girth at least five. Moreover, we show that every P_{14}-free graph of girth at least six and every P_{17}-free graph of girth at least seven is 3-colorable. This strengthens results of Golovach et al.

1 Introduction

Given a graph G, a *k-coloring* is a mapping $c : V(G) \to \{1, \ldots, k\}$ with $c(u) \neq c(v)$ for all edges uv of G. If a k-coloring exists for G, we call G *k-colorable*. Moreover G is called *k-chromatic* if it is k-colorable, but not $(k-1)$-colorable.

The graph G graph is called *k-critical* if it is k-chromatic, but every proper subgraph of G is $(k-1)$-colorable. For example, the class of 3-critical graphs equals the family of odd cycles. The characterization of k-critical graphs is a notorious problem in graph theory.

To get a grip on this problem, it is common to consider graphs with restricted structure, as follows. Let a graph H and a number k be given. An *H-free* graph is a graph that does not contain H as an induced subgraph. We say that a graph G is *k-critical H-free* if G is H-free, k-chromatic, and every H-free proper subgraph of G is $(k-1)$-colorable. If $\mathcal{H}$ is a set of graphs, then we say that a graph G is $\mathcal{H}$-free if G is H-free for each $H \in \mathcal{H}$. The definition of a *k-critical $\mathcal{H}$-free* graph is analogous.

A notion similar to critical graphs is that of *k-vertex-critical* graphs: k-chromatic graphs for which every proper induced subgraph is $(k-1)$-colorable. We define k-vertex-critical H-free and k-vertex-critical $\mathcal{H}$-free graphs accordingly. Note that, unlike for critical graphs, the set of k-vertex-critical $\mathcal{H}$-free graphs equals the set of $\mathcal{H}$-free k-vertex-critical graphs.

We remark that every k-critical graph is k-vertex-critical. Moreover, as noted by Hoàng et al. [12], there are finitely many k-critical $\mathcal{H}$-free graphs if and only if there are finitely many k-vertex-critical $\mathcal{H}$-free graphs, for any family of graphs $\mathcal{H}$.

The study of k-critical graphs in a particular graph class received a significant amount of interest in the past decade, which is partly due to the interest in the design of certifying algorithms. Given a decision problem, a solution algorithm is called *certifying* if it provides, together with the yes/no decision, a polynomial time verifiable certificate for this decision. In case of k-colorability for $\mathcal{H}$-free graphs, a canonical certificate would be either a k-coloring or an induced subgraph of the input graph which is (a) not k-colorable and (b) of constant size. However, assertion (b) can only be realized if there is a finite list of $(k+1)$-critical $\mathcal{H}$-free graphs.

Let us now mention some results in this line of research. From Lozin and Kamiński [13] and Kràl et al. [14] we know that the k-colorability problem remains NP-complete for H-free graphs, unless H is

*Jan Goedgebeur, jan.goedgebeur@ugent.be, Department of Applied Mathematics, Computer Science & Statistics, Ghent University, Krijgslaan 281-S9, 9000 Ghent, Belgium

†Oliver Schaudt, schaudto@uni-koeln.de, Institut für Informatik, Universität zu Köln, Köln, Germany

the disjoint union of paths. This motivates the study of graph classes in which some path is forbidden as induced subgraph.

Bruce et al. [5] proved that there are exactly six 4-critical P_5-free graphs, where P_t denotes the path on t vertices. Later, Maffray and Morel [15], by characterizing the 4-vertex-critical P_5-free graphs, designed a linear time algorithm to decide 3-colorability of P_5-free graphs. Randerath et al. [19] have shown that the only 4-critical (P_6, C_3)-free graph is the Grötzsch graph. More recently, Hell and Huang [11] proved that there are four 4-critical (P_6, C_4)-free graphs. They also proved that in general, there are only finitely many k-critical (P_6, C_4)-free graphs.

In a companion paper, we proved the following dichotomy theorem together with Chudnovsky and Zhong for the case of a single forbidden induced subgraph, answering questions of Golovach et al. [9] and Seymour [20].

Theorem 1 (Chudnovsky et al. [6])**.** *Let H be a connected graph. There are finitely many 4-critical H-free graphs if and only if H is a subgraph of P_6.*

The main difficulty in the proof of the above theorem is to show that there are only finitely many 4-critical P_6-free graphs, namely 24. A substantial step in this proof, in turn, is to show that there only finitely many 4-critical $(P_6, \text{diamond})$-free graphs. After unsuccessfully trying to prove this by hand, we developed an algorithm to automatize the huge amount of case distinctions, based on a method recently proposed by Hoàng et al. [12].

In the present paper, we thoroughly extend this algorithm in order to derive more characterizations of 4-critical $\mathcal{H}$-free graphs. As a demonstration of the power of this algorithm, we prove the following results.

- There are exactly 6 4-critical (P_7, C_4)-free graphs.
- There are exactly 17 4-critical (P_7, C_5)-free graphs.
- There are exactly 94 4-critical (P_8, C_4)-free graphs.
- There are exactly 52 4-critical planar P_7-free graphs. In addition, using a result of Böhme et al. [1] we derive that for every t there are only finitely many 4-critical planar P_t-free graphs.
- Every P_{11}-free graph of girth at least five is 3-colorable and there is a 4-chromatic P_{12}-free graph of girth 5.
- Every P_{14}-free graph of girth at least six is 3-colorable.
- Every P_{17}-free graph of girth at least seven is 3-colorable.

Our results extend and/or strengthen previous results of Hell and Huang [11] and Golovach et al. [10].

Besides these results, some of which we think are infeasible to prove by hand, we see the algorithm as the main contribution of our paper. Its modular design allows to easily implement new expansion rules, which makes it adaptable to closely related problems in this line of research. To this end, we also mention a case that was out of reach with the current algorithm, but where we have a good feeling that there is only a finite set of obstructions.

In the next section we propose a number of lemmas that give necessary conditions for k-critical graphs. Our generation algorithm, which we also present in the next section, is built on these lemmas.

In Section 3 we prove the above mentioned results, and also discuss some results from the literature that we verified.

2 A generic algorithm to find all k-critical $\mathcal{H}$-free graphs

We build upon a method recently proposed by Hoàng et al. [12]. With this method they have shown that there is a finite number of 5-critical (P_5, C_5)-free graphs.

The idea is to use necessary conditions for a graph to be critical to generate all critical graphs. The algorithm then performs all remaining case distinctions automatically. In order to deal with more advanced cases, we need to thoroughly alter the approach of Hoàng et al. [12]. We remark that the algorithm presented below is, moreover, a substantial strenghtening of that used in the proof of Theorem 1, and this strenghtening is necessary in order to derive the results of the present paper.

2.1 Preparation

In this section we prove a number of lemmas which will later be used as expansion rules for our generation algorithm. Although each of them is straightforward, they turn out to be very useful for our purposes.

We use the following notation. The set $N_G(v)$ denotes the neighborhood of a vertex v in G. If it is clear from the context which graph is meant, we abbreviate this to $N(v)$. The graph $G|U$ denotes the subgraph of G induced by the vertex subset $U \subseteq V(G)$. Moreover, for a vertex subset $U \subseteq V(G)$ we denote by $G - U$ the induced subgraph $G|(V(G) \setminus U)$ of G.

Let G be a k-colorable graph. The *k-hull* of G, which we denote G_k, is the graph obtained from G by making two vertices u and v adjacent if and only if there is no k-coloring of G under which u and v receive the same color. Clearly G_k is a supergraph of G without loops, and G_k is k-colorable.

It is a folklore fact that a k-critical graph cannot contain two distinct vertices u and v with $N(u) \subseteq N(v)$. The following observation is a proper generalization of this fact, and we proved it together with Chudnovsky and Zhong in [6].

Lemma 2 (Chudnovsky et al. [6])**.** *Let $G = (V, E)$ be a k-vertex-critical graph and let U, W be two non-empty disjoint vertex subsets of G. Let $H := (G - U)_{k-1}$. If there exists a homomorphism $\phi : G|U \mapsto H|W$, then $N_G(u) \setminus U \not\subseteq N_H(\phi(u))$ for some $u \in U$.*

We make use of Lemma 2 in the following way. Assume that G' is a k-vertex-critical graph and G is a $(k-1)$-colorable induced subgraph of G'. Then pick two disjoint vertex subsets U and W of G, both non-empty, and let $H := (G - U)_{k-1}$. Assume that there is a homomorphism $\phi : G|U \mapsto H|W$ with $N_G(u) \setminus U \subseteq N_H(\phi(u))$ for each $u \in U$. Then we know that there is a vertex $x \in V(G') \setminus V(G)$ adjacent to some $u \in U$ but non-adjacent to $\phi(u)$, in the graph G'. Also x is non-adjacent to $\phi(u)$ in $(G' - U)_{k-1}$.

Recall that any k-critical graph G must have minimum degree at least $k - 1$. As otherwise, any $(k-1)$-coloring of $G - u$ can be extended to a $(k-1)$-coloring of G, where u is some vertex in G of degree at most $k - 2$. The following is an immediate strengthening.

Lemma 3. *Let G be a k-vertex-critical graph and $u \in V(G)$. Then in any $(k-1)$-coloring of $G - u$, the set $N_G(u)$ receives $k - 1$ distinct colors.*

We make use of Lemma 3 in the following way. Assume that G is a $(k-1)$-colorable graph that is an induced subgraph of some k-vertex-critical graph G'. Suppose that there is a vertex u such that there is no $(k-1)$-coloring of $G - u$ in which the set $N_G(u)$ receives $k - 1$ distinct colors. Let us say that ℓ is the maximum number of distinct colors that the set $N_G(u)$ can recieve in a $(k-1)$-coloring of $G - u$. Then there must be some vertex $v \in N_{G'}(u) \setminus V(G)$ such that there is a $(k-1)$-coloring of $G'|(V(G - u) \cup \{v\})$ in which the set $N_G(u) \cup \{v\}$ receives $\ell + 1$ distinct colors.

We also need the following fact which is folklore: every cutset of a critical graph contains at least two non-adjacent vertices, where a *cutset* is a vertex subset whose removal increases the number of connected components of the graph.

Lemma 4. *Let G be a k-critical graph, and let X be a clique of G. Then $G - X$ is a connected graph.*

In fact, we only need that a k-critical graph does not have a cutvertex, that is, a cutset of size 1. We use Lemma 4 as follows. Given a $(k-1)$-colorable graph G that is an induced subgraph of some k-vertex-critical graph G'. Suppose that there is a cutvertex u in G. Then there must be some vertex $v \in V(G') \setminus V(G)$ with at least one neighbor in $V(G) \setminus \{u\}$.

The next lemma is custom-made for the case of 4-critical graphs.

Lemma 5. *Let G be a 4-vertex-critical graph. Suppose that there is an induced cycle C where all vertices are of degree three. Then C has odd length, and there is a 3-coloring of $G - V(C)$ for which every member of the set $(\bigcup_{c \in V(C)} N(c)) \setminus V(C)$ receives the same color.*

Proof. Consider the graph C_k, for some $k \geq 3$, where each vertex v is equipped with a two-element list $L(v) \subseteq \{1, 2, 3\}$. It is an easy exercise to show that this graph has a valid list coloring, unless k is odd and all lists are identical.

Consequently, given any 3-coloring c of $G - V(C)$, we may extend this coloring to a 3-coloring of G unless C is of odd length and c is constant on the set $(\bigcup_{c \in V(C)} N(c)) \setminus V(C)$. □

We use Lemma 5 as follows. Given a 3-colorable graph G that is an induced subgraph of some 4-vertex-critical graph G'. Suppose that there is an induced cycle C in G where all vertices are of degree three. Moreover, suppose that C has even length, or C has odd length and in every 3-coloring of $G-V(C)$ the set $(\bigcup_{c\in V(C)} N(c)) \setminus V(C)$ is not monochromatic. Then there must be some vertex $v \in V(G') \setminus V(G)$ with at least one neighbor in $V(C)$.

The next lemma tells us from which graphs we have to start our enumeration algorithm.

Lemma 6. *Every k-critical P_t-free graph different from K_k contains one of the following graphs as induced subgraph:*

- *an odd hole C_{2s+1}, for $2 \leq s \leq \lfloor (t-1)/2 \rfloor$, or*
- *an odd antihole $\overline{C_{2s+1}}$, for $3 \leq s \leq k-1$.*

Proof. It follows from the Strong Perfect Graph Theorem [7] that every k-critical graph different from K_k must contain an odd hole or anti-hole as an induced subgraph. The statement of the lemma thus follows from the following facts:

(a) the cycle C_{2s+1} contains an induced P_{2s},

(b) the antihole $\overline{C_{2k+1}}$ has clique number k, and

(c) $\overline{C_5}$ is isomorphic to C_5.

□

2.2 The enumeration algorithm

We use Algorithm 1 below to enumerate all $\mathcal{H}$-free k-critical graphs. In order to keep things short, we use the following conventions for a $(k-1)$-colorable graph G.

We call a pair (u, v) of distinct vertices for which $N_G(u) \subseteq N_{(G-u)_{k-1}}(v)$ *similar vertices*. Similarly, we call a 4-tuple (u, v, u', v') of distinct vertices with $uv, u'v' \in E(G)$ such that $N_G(u) \setminus \{v\} \subseteq N_{(G-\{u,v\})_{k-1}}(u')$ and $N_G(v) \setminus \{u\} \subseteq N_{(G-\{u,v\})_{k-1}}(v')$ *similar edges*. Finally, we define *similar triangles* in an analogous fashion.

Recall that a diamond is the graph obtained from K_4 by removing one edge. We define *similar diamonds* in complete analogy to similar triangles.

Let u be a vertex of G for which, in every $(k-1)$-coloring of $G-u$, the set $N_G(u)$ receives at most $k-2$ distinct colors. Then we call u a *poor vertex*.

Let C be an induced cycle in G such that every vertex of C has degree three. We say that C is a *weak cycle* if C is of even length or, if C has odd length, there is no 3-coloring of $G - V(C)$ for which every member of the set $(\bigcup_{c\in V(C)} N(c)) \setminus V(C)$ receives the same color.

Algorithm 1 Generate $\mathcal{H}$-free k-critical graphs

```
1: let F be an empty list
2: Construct(K_k)  // i.e. perform Algorithm 2
3: for all graphs G mentioned in Lemma 6 do
4:    Construct(G)  // i.e. perform Algorithm 2
5: end for
6: Output F
```

We now prove that Algorithm 1 is correct.

Theorem 7. *Assume that Algorithm 1 terminates, and outputs the list of graphs $\mathcal{F}$. Then $\mathcal{F}$ is the list of all k-critical $\mathcal{H}$-free graphs.*

Proof. In view of lines 1 and 3 of Algorithm 2, it is clear that all graphs of $\mathcal{F}$ are k-critical H-free. So, it remains to prove that $\mathcal{F}$ contains all k-critical $\mathcal{H}$-free graphs. To see this, we first prove the following claim.

Algorithm 2 Construct(Graph G)

if G is $\mathcal{H}$-free AND not generated before **then**
  if G is not $(k-1)$-colorable **then**
    if G is k-critical $\mathcal{H}$-free **then**
      add G to the list $\mathcal{F}$
    end if
  else
    if G contains similar vertices (u, v) **then**
      for every graph H obtained from G by attaching a new vertex x and incident edges in all possible ways, such that $ux \in E(H)$, but $vx \notin E((H-u)_{k-1})$ **do**
        Construct(H)
      end for
    else if G contains a poor vertex u **then**
      for every graph H obtained from G by attaching a new vertex x and incident edges in all possible ways, such that $ux \in E(H)$ and the maximum number of distinct colors the set $N_H(u)$ recieves in some $(k-1)$-coloring of $H-u$ properly increased **do**
        Construct(H)
      end for
    else if G contains similar edges (u, v, u', v') **then**
      for every graph H obtained from G by attaching a new vertex x and incident edges in all possible ways, such that $rx \in E(H)$ and $r'x \notin E((H-\{u,v\})_{k-1})$ for some $r \in \{u, v\}$ **do**
        Construct(H)
      end for
    else if G contains similar triangles (u, v, w, u', v', w') **then**
      for every graph H obtained from G by attaching a new vertex x and incident edges in all possible ways, such that $rx \in E(H)$ and $r'x \notin E((H-\{u,v,w\})_{k-1})$ for some $r \in \{u, v, w\}$ **do**
        Construct(H)
      end for
    else if G contains similar diamonds $(u, v, w, x, u', v', w', x')$ **then**
      for every graph H obtained from G by attaching a new vertex y and incident edges in all possible ways, such that $ry \in E(H)$ and $r'y \notin E((H-\{u,v,w,x\})_{k-1})$ for some $r \in \{u, v, w, x\}$ **do**
        Construct(H)
      end for
    else if $k = 4$ and G contains a weak cycle C **then**
      for every graph H obtained from G by attaching a new vertex x with a neighbor in C and incident edges in all possible ways **do**
        Construct(H)
      end for
    else if G contains a cutvertex u **then**
      for every graph H obtained from G by attaching a new vertex x adjacent to some member of $V(G) \setminus \{u\}$ and incident edges in all possible ways **do**
        Construct(H)
      end for
    else
      for every graph H obtained from G by attaching a new vertex x and incident edges in all possible ways **do**
        Construct(H)
      end for
    end if
  end if
end if

Claim 1. *For every k-critical $\mathcal{H}$-free graph F other than K_k, Algorithm 2 applied to some graph F' from Lemma 6 generates an isomorphic copy of an induced subgraph of F with i vertices for every $|V(F')| \leq i \leq |V(F)|$.*

We prove this by induction, as an invariant of our algorithm. Due to Lemma 6, we know that F contains some F' as an induced subgraph, so the claim holds for $i = |V(F')|$.

So assume that the claim is true for some i with $|V(F')| \leq i \leq |V(F)| - 1$. Let G be an induced subgraph of F with $|V(G)| = i$ such that an isomorphic copy of G, say G' is generated by the algorithm. We may choose G' such that it is not pruned by the isomorphism check on line 1. In order to save notation, let us assume that $G' = G$.

First assume that G contains similar vertices, and that the similar vertices (u, v) are considered by the algorithm. Then, by Lemma 2, $N_F(u) \not\subseteq N_{(F-u)_{k-1}}(v)$. Hence, there is some vertex $x \in V(F) \setminus V(G)$ which is adjacent to u in F, but not to v in $(F - u)_{k-1}$. Following the statement of line 9, Construct$(F|(V(G) \cup \{x\}))$ is called. We omit the discussion of the lines 15, 19, and 23 as they are analogous.

So assume that none of the criteria checked earlier apply to G, but the algorithm finds a poor vertex u in G. Then, by Lemma 3, there is some vertex $x \in V(F) \setminus V(G)$ adjacent to u such that the maximum number of distinct colors the set $N_H(u)$ recieves in some $(k-1)$-coloring of $H - u$ increased compared to $G - u$, where $H := F|(V(G) \cup \{x\})$. Following the statement of line 11, Construct(H) is called.

Now assume that none of the criteria checked earlier apply to G, but we have $k = 4$ and G contains a weak cycle. Say C is the weak cycle considered by the algorithm. By Lemma 5, F does not contain a weak cycle, and so there must be some vertex $x \in V(F) \setminus V(G)$ with a neighbor in C. Hence, Construct$(F|(V(G) \cup \{x\}))$ is called in line 29.

Otherwise if the algorithm finds a cutvertex u of G, there must be some vertex $x \in V(F) \setminus V(G)$ which is adjacent to some vertex in $V(G) \setminus \{u\}$. This follows from Lemma 4. Following the statement of line 33, Construct$(F|(V(G) \cup \{x\}))$ is called.

Finally, if none of the above criteria apply to G, the algorithm attaches a new vertex to G in all possible ways, and calls Construct for all of these new graphs. Since $|V(F)| > |V(G)|$, among these graphs there is some induced subgraph of F, and of course this graph has $i + 1$ vertices. This completes the proof of Claim 1.

Given that the algorithm terminates, Claim 1 implies that $\mathcal{F}$ must contain all k-critical $\mathcal{H}$-free graphs. □

We implemented this algorithm in C with some further optimizations (see Section 2.3). We used the program `nauty` [16, 18] to make sure that no isomorphic graphs are accepted. More specifically, we use `nauty` to compute a canonical form of the graphs. We maintain a list of the canonical forms of all non-isomorphic graphs which were generated so far and only accept a graph if it was not generated before (in which case its canonical form is added to the list). More sophisticated isomorphism rejection techniques are known (such as the canonical construction path method [17]), but these methods are not compatible with the destruction of similar *elements* in Algorithm 2. Furthermore isomorphism rejection is not a bottleneck in our program.

Our program does indeed terminate in several cases. More details about this can be found in Section 3. The source code of the program can be downloaded from [8] and in the Appendix we describe how we extensively tested the correctness of our implementation.

2.3 Optimizations and implementation details

In this section we describe implementation details and additional optimizations which significantly speed up the program.

2.3.1 Priority of operations

The order or priority in which the next expansions are determined in Algorithm 2 are vital for the termination of the program. For most cases, the optimal order is as described in Algorithm 2: i.e. first test if the graph contains similar vertices, if this is not the case, then test if it contains small vertices, if this is not the case then test if it contains similar edges, etc.

Even within the same category of similar *elements*, the intermediate graphs generated by Algorithm 2 typically contain multiple similar elements. Choosing the right similar element is also important for the efficiency and termination of the algorithm. In most cases it is best to destroy the similar element with the smallest degree.

We also tried to extend every similar element once without iterating any further and then choosing the similar element with the least number of generated children and extending it recursively. But this was a lot slower and did not significantly reduce the number of graphs generated in the cases we investigated, except when generating critical graphs with a given minimal girth (see Section 3.3).

2.3.2 Testing of properties

It is important to test $\mathcal{H}$-freeness, colorability and isomorphism in the right order. In our case, most generated graphs are k-colorable but not $\mathcal{H}$-free, and not too many isomorphic copies are generated. In view of this, it is best to test these properties in the following order:

1. $\mathcal{H}$-freeness
2. k-colorability
3. Isomorphism

In most cases the routine which tests if a graph contains an induced P_t is a bottleneck. As nearly all generated graphs contain an induced P_t (for the values of t which are within reach of the program), the following heuristic helps a lot. Whenever an induced P_t was found in a graph, we store it. When we test the next graph, we first test if one of the previously stored P_t's from a previous graph (with the same number of vertices) is still an induced path in the current graph and only if this is not the case, we exhaustively search for an induced P_t.

Experiments show that is optimal to store approximately 20 such previous P_t's and in about 90% of the cases this heuristic allows to reject the generated graph since an induced P_t was found.

2.3.3 Limiting expansions

The algorithm connects the new vertex v_n to a set of vertices $S \subseteq \{v_0, ..., v_{n-1}\}$. When connecting v_n to S yields a graph which is not k-colorable, connecting v_n to $S' \supseteq S$ will yield a graph which is also not k-colorable and not $(k+1)$-critical, so these expansions can be skipped. Note that we cannot apply this optimization if we are looking for vertex-critical graphs.

The same optimization can also be applied if connecting v_n to S yields a graph with a cycle of length smaller than g or a non-planar graph when searching for graphs with girth at least g or planar graphs, respectively.

3 Results

This section describes the main results obtained with our implementation of Algorithm 1. In the Appendix we describe how we tested the correctness of our implementation.

The adjacency lists of all new critical graphs from this section can be found in Appendix 2 and these graphs can also be downloaded from the *House of Graphs* [3] at `http://hog.grinvin.org/Critical`

3.1 Verification of previously known results

As a correctness test and to demonstrate the strength of the approach we verified the following characterizations which were known before in the literature.

- There are six 4-critical P_5-free graphs [5].
- There are eight 5-critical (P_5, C_5)-free graphs [12].
- The Grötzsch graph is the only 4-critical (P_6, C_3)-free graph [19].
- There are four 4-critical (P_6, C_4)-free graphs [11].

In each of the above cases our program terminates in a few seconds.

3.2 4-critical (P_r, C_s)-free graphs

It is known [6] that there is an infinite family of 4-critical P_7-free graphs. However, a careful observation shows that all members of this family contain a C_k for all $k = 3, 4, 5$, and are C_ℓ-free for $\ell = 6, 7$. This motivates the study of 4-critical (P_7, C_k)-free graphs when $k = 3, 4, 5$, since there might be only finitely many of these. We can solve the cases of $k = 4$ and $k = 5$ using Algorithm 1. Moreover, we can also solve the (P_8, C_4)-free case.

Theorem 8. *The following assertions hold.*

(a) There are exactly 17 4-critical (P_7, C_4)-free graphs.

(b) There are exactly 94 4-critical (P_8, C_4)-free graphs.

(c) There are exactly 6 4-critical (P_7, C_5)-free graphs.

Proof of Theorem 8. Algorithm 1 terminates in the (P_7, C_4)-free case and outputs 17 4-critical graphs. It also terminates in the (P_8, C_4)-free case and outputs 94 4-critical graphs. Hence, by Theorem 7, (a) and (b) both hold.

In the (P_7, C_5)-free case we need to use a slightly modified version of Algorithms 1 and 2. Instead of just graphs, the algorithms now consider pairs $P = (G, \text{triples}(P))$ where G is a graph and for each $(u, v, x) \in \text{triples}(P)$, u, v, and x are distinct vertices of G. In Algorithm 1, we call Construct(P) for all pairs of the form $(G, \emptyset)$ where G is a graph from Lemma 6.

At the beginning of Algorithm 2 where we consider some pair $P = (G, \text{triples}(P))$ we check whether, for each triple $(u, v, x) \in \text{triples}(P)$, there is some $(k-1)$-coloring of $G-u$ where v receives the same color as x. If this is not the case, then we discard the pair P and return. Since we need to seperately consider distinct pairs P, P' even if their graphs G, G' are isomorphic, we do not perform the isomorphism test in line 1 anymore.

In all subsequent lines where usually Construct(H) is called, we instead call Construct(P'), where $P' = (H, \text{triples}(P))$. Note that triples($P'$) equals triples($P$) here.

The only exception is the call of Construct(H) in line 9. Let us say the algorithm considers the pair (u, v) of similar vertices, and the vertex x is the new vertex added to the graph G. Then the algorithm calls Construct(P') in line 9, where $P' = (H, \text{triples}(P) \cup \{(u, v, x)\})$.

To see that this is correct, it suffices to prove the following altered version of Claim 1.

Claim 2. *Let F be a k-critical $\mathcal{H}$-free graph other than K_k. The modified version of Algorithm 2 applied to the pair $(F', \emptyset)$, where F' is some graph from Lemma 6, generates a pair $P^i = (G^i, \text{triples}(P^i))$ with $|V(G^i)| = i$ for every $|V(F')| \leq i \leq |V(F)|$ such that following assertions hold.*

(a) There is an injective homomorphism ϕ^i from G^i to F such that, for each triple $(u, v, x) \in \text{triples}(P^i)$, there is some $(k-1)$-coloring of $F - \phi^i(u)$ where $\phi^i(v)$ and $\phi^i(x)$ receive the same color.

(b) The graph G^i is an induced subgraph of G^{i+1}, $\text{triples}(P^i) \subseteq \text{triples}(P^{i+1})$, and $\phi^{i+1}|_{V(G^i)} \equiv \phi^i$ for all $|V(F')| \leq i \leq |V(F)| - 1$.

We prove this by induction, as an invariant of our algorithm. Due to Lemma 6, we know that F contains some F' as an induced subgraph, so the claim holds for $i = |V(F')|$.

So assume that the claim is true for some i with $|V(F')| \leq i \leq |V(F)| - 1$. Let $P^i = (G^i, \text{triples}(P^i))$ and ϕ^i be as desired. First assume that G^i contains similar vertices (u, v). By Lemma 2, $N_F(\phi^i(u)) \not\subseteq N_{(F-\phi^i(u))_{k-1}}(\phi^i(v))$. Hence, there is some vertex $x \in V(F) \setminus \text{im } \phi^i$ which is adjacent to $\phi^i(u)$ in F, but not to $\phi^i(v)$ in $(F - \phi^i(u))_{k-1}$. Here, im ϕ^i denotes the image of ϕ^i. Thus, we may safely define ϕ^{i+1} and the pair $P^{i+1} = (G^{i+1}, \text{triples}(P^{i+1}))$ as follows.

(a) $V(G^{i+1}) = V(G^i) \cup \{y\}$ and $G^{i+1}|V(G^i) = G^i$, where y is the new vertex added by the algorithm in line 9,

(b) $\phi^{i+1}|_{V(G^i)} \equiv \phi^i$, $\phi^{i+1}(y) = x$,

(c) ϕ^{i+1} is an injective homomorphism from G^{i+1} to F, and

(d) $P^{i+1} = P^i \cup (u, v, y)$.

By construction, for each triple $(u, v, x) \in \text{triples}(P^{i+1})$ there is some $(k-1)$-coloring of $F - \phi^{i+1}(u)$ where $\phi^{i+1}(v)$ and $\phi^{i+1}(x)$ receive the same color. Moreover, Construct(P^{i+1}) is called in line 9.

If G^i does not contain similar vertices, we proceed in complete analogy. The only difference is that we have to set $P^{i+1} = P^i$.

The remainder of the proof of Claim 2 is literally the same as that of Claim 1. □

These 4-critical (P_7, C_4)-free and (P_7, C_5)-free graphs are shown in Figure 1 and 2, respectively. The adjacency lists of these graphs can be found in Appendix 2.

We also determined that there are exactly 35 4-vertex-critical (P_7, C_4)-free graphs, 164 4-vertex-critical (P_8, C_4)-free graphs, and 27 4-vertex-critical (P_7, C_5)-free graphs (details on how we obtained these graphs can be found in the Appendix).

The Tables 1, 2, and 3 give an overview of the counts of the 4-critical and 4-vertex-critical graphs mentioned in Theorem 8.

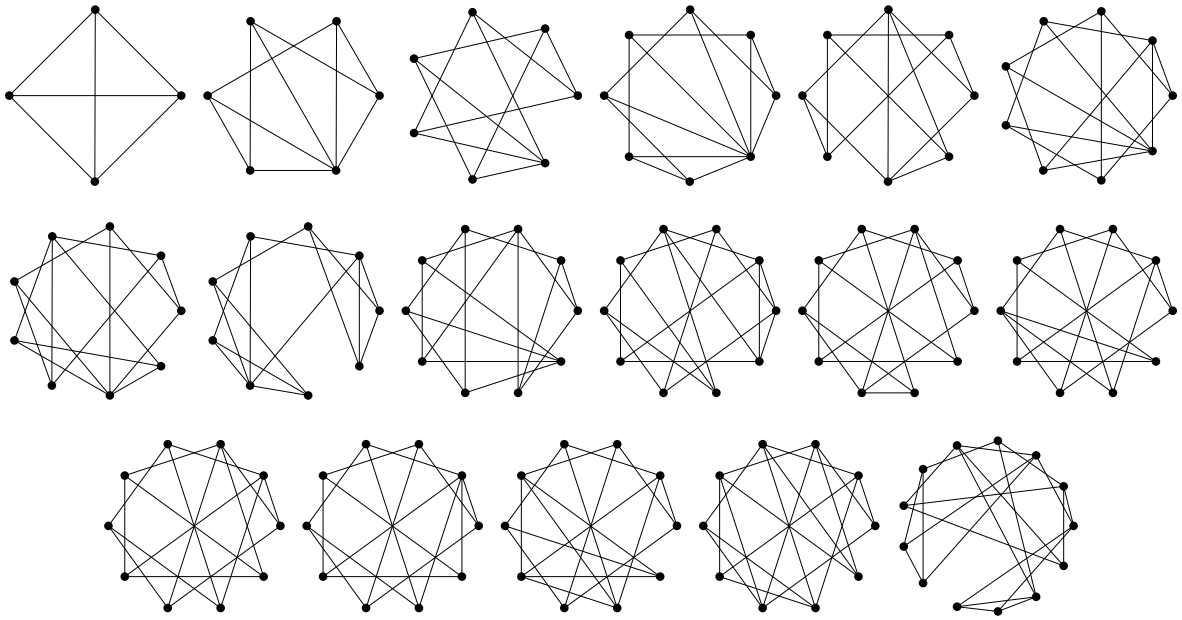

Figure 1: All 17 4-critical (P_7, C_4)-free graphs.

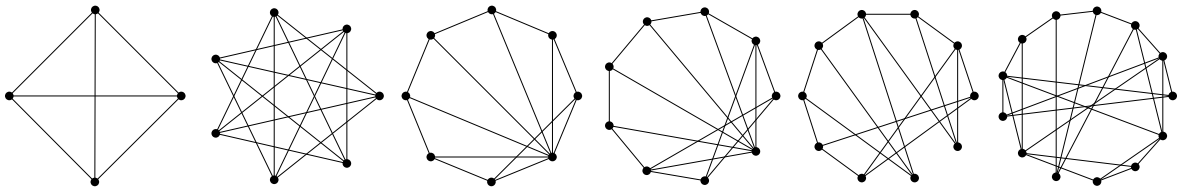

Figure 2: All 6 4-critical (P_7, C_5)-free graphs.

Vertices	4	6	7	8	9	10	13	total
Critical graphs	1	1	1	2	3	8	1	17
Vertex-critical graphs	1	1	1	2	4	24	2	35

Table 1: Counts of all 4-critical and 4-vertex-critical (P_7, C_4)-free graphs.

We did not succeed in solving the (P_7, C_3)-free case. However, we were able to determine all 4-critical (P_7, C_3)-free graphs with at most 35 vertices. These are shown in Figure 3, the counts are shown in Table 4 and their adjacency lists can be found in Appendix 2. Since the largest 4-critical (P_7, C_3)-free graph up to 35 vertices has only 16 vertices, we conjecture the following.

Vertices	4	6	7	8	9	10	11	12	13	14	total
Critical graphs	1	1	1	2	3	15	28	34	8	1	94
Vertex-critical graphs	1	1	1	2	4	33	54	53	14	1	164

Table 2: Counts of all 4-critical and 4-vertex-critical (P_8, C_4)-free graphs.

Vertices	4	7	8	9	10	13	total
Critical graphs	1	1	1	1	1	1	6
Vertex-critical graphs	1	1	1	6	17	1	27

Table 3: Counts of all 4-critical and 4-vertex-critical (P_7, C_5)-free graphs.

Conjecture 9. *The seven graphs from Figure 3 are the only 4-critical (P_7, C_3)-free graphs.*

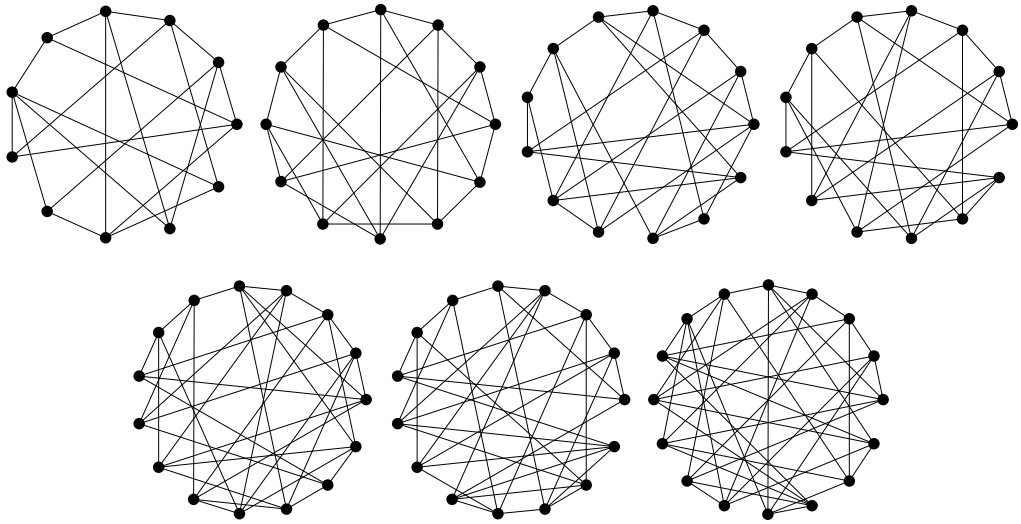

Figure 3: All seven 4-critical (P_7, C_3)-free graphs with at most 35 vertices.

Vertices	11	12	13	15	16	total
Critical graphs	1	1	2	2	1	7

Table 4: Counts of all 7 4-critical (P_7, C_3)-free graphs with at most 35 vertices. The number of 4-vertex-critical graphs is the same.

3.3 4-critical graphs with a given minimal girth

In the 1980s, Sumner [22] proved that every (P_5, C_3)-free graph is 3-colorable. This result has been extended in various directions, e.g., Golovach et al. [10] proved that every P_7-free graph of girth at least five is 3-colorable. They mention the Brinkmann graph, constructed by Brinkmann and Meringer in [4], as an example of a P_{10}-free of girth five which is not 3-colorable. However, this claim is not entirely correct since the Brinkmann graph contains a P_{12} as an induced subgraph. Using Algorithm 1 we can in fact show that every P_{11}-free graph with girth at least five is 3-colorable. We also improve upon results of Golovach et al. [10] in the cases of girth at least six, and at least seven. These results are also summarized in Table 5.

Theorem 10. *The following assertions hold.*

(a) Every P_{11}-free graph of girth at least five is 3-colorable.

(b) There is a 4-chromatic P_{12}-free graph of girth five, and the smallest such graph has 21 vertices.

(c) Every P_{14}-free graph of girth at least six is 3-colorable.

(d) Every P_{17}-free graph of girth at least seven is 3-colorable.

Proof. Algorithm 1 terminates in the cases of (a), (c), and (d). It outputs an empty list of 4-critical graphs. Thus, by Theorem 7, there are no 4-critical graphs in the respective graph classes, which means that all graphs therein are 3-colorable.

To prove (b), we modified Algorithm 1 such that it discards any graph with more than 30 vertices. Indeed no other 4-critical graphs than one graph on 21 vertices were found. By Theorem 7, (b) follows. □

We remark that the 4-critical P_{12}-free graph with girth five mentioned in Theorem 10.(b) is the only such graph up to at least 30 vertices. This graph is shown in Figure 4 and its adjacency list can be found in Appendix 2. It can also be obtained from the *House of Graphs* [3] by searching for the keywords "4-critical P12-free * girth 5".

Note that a graph with girth at least five cannot contain similar vertices (u, v) which both have degree at least two. Experiments showed that when searching for critical P_t-free graphs with a given minimal girth, it is best only to try to apply the following expansion rules in Algorithm 2: poor vertices (line 11), weak cycles (line 29) and cutvertices (line 33). This saves a significant amount of CPU time since then one does not have to search for similar *elements* or compute *k-hulls*.

Girth	Old lower bound [10]	New lower bound
4	P_5-free (exact)	P_5-free (exact)
5	P_7-free	P_{11}-free (exact)
6	P_{10}-free	P_{14}-free
7	P_{12}-free	P_{17}-free

Table 5: Old and new lower bounds such that every P_k-free graph with girth at least g is 3-colorable.

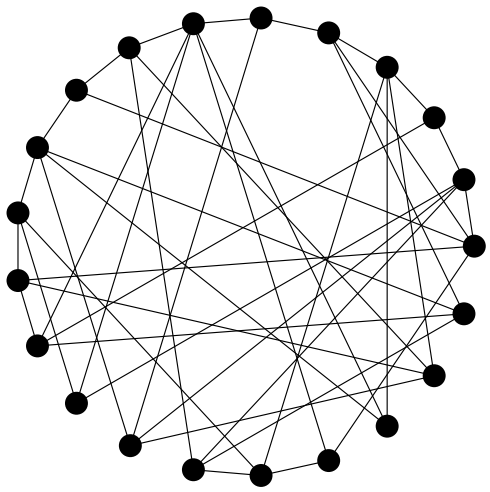

Figure 4: The smallest 4-critical P_{12}-free graph with girth 5. It has 21 vertices and it is the only 4-critical P_{12}-free graph with girth at least five up to at least 30 vertices.

3.4 4-critical planar graphs

In this section we turn to planar critical graphs. Due to the four-color theorem, we may restrict our attention to 4-critical graphs. Let us first note that if an induced path is forbidden, there are only finitely many vertex-critical planar graphs. This implies that there are only finitely many 4-critical planar P_t-free graphs, for all $t \in \mathbb{N}$.

Theorem 11. *For any integer t there are only finitely many vertex-critical graphs that are both planar and P_t-free.*

For the proof of Theorem 11, we need the following result of Böhme et al. [1], which we state in a slightly different fashion than in the original paper.

Theorem 12 (Böhme et al. [1])**.** *For every $k, s, t \in \mathbb{N}$ there is a number $n = n(k, s, t)$ such that every k-connected graph on at least n vertices contains an induced path on t vertices or a subdivision of $K_{2,s}$.*

From this result our theorem follows readily.

Proof of Theorem 11. Let $n = n(2, 2t, t)$ be the number promised by Theorem 12. Suppose for a contradiction that there is a planar P_t-free vertex-critical graph $G = (V, E)$ on at least n vertices. Clearly G is connected. We consider G to be embedded into the plane.

By Theorem 12 and since G is P_t-free, G contains a subdivision H of $K_{2,2t}$ as a subgraph. We consider H as an embedded subgraph of G. Let u and v be the two vertices of degree $2t$ in H, and let $P^1, \ldots, P^{2t}$ be the mutually vertex disjoint paths from u to v in H. W.l.o.g. $P^i \cup P^{i+1}$ is the border of an inner face of H, for each $i = 1, \ldots, 2t-1$, and $P^1 \cup P^{2t}$ is the border of the outer face.

First we assume that $G - \{u, v\}$ is connected. We pick two vertices $x \in V(P^1) \setminus \{u, v\}$ and $y \in V(P^t) \setminus \{u, v\}$. Let Q be a shortest path from x to y in $G - \{u, v\}$. Due to the planarity of G, necessarily Q contains a vertex of each P^i, $i = 1, \ldots, t$, or of each P^j, $j = 1, t, t+1, \ldots, 2t$. So, Q is an induced path on at least t vertices, in contradiction to the fact that G is P_t-free.

So we know that $\{u, v\}$ is a cutset of G. Thus $G - \{u, v\}$ has exactly two components, say G_1 and G_2, a folklore fact about vertex-critical graphs. We may assume that the paths $P^1 - \{u, v\}, \ldots, P^t - \{u, v\}$ are contained in G_1. We again pick vertices $x \in V(P^1) \setminus \{u, v\}$ and $y \in V(P^t) \setminus \{u, v\}$. Let Q be a shortest path from x to y in G_1. Since G_2 is not empty and G is planar, Q must contain a vertex of each P^i, $i = 1, \ldots, t$. But this is again a contradiction to the fact that G is P_t-free. □

Unfortunately, Theorem 11 does not provide a list of 4-critical graphs, nor yield a useful bound on their order. Using a slight modification of our algorithm, we obtained the following result.

Theorem 13. *There are exactly 52 4-critical planar P_7-free graphs.*

Proof. We modified Algorithm 2 so it only searches for *planar* k-critical $\mathcal{H}$-free graphs by adding a test for planarity on line 1. We used the algorithm of Boyer and Myrvold [2] to test if a graph is planar. Moreover, we included a search for similar P_4's and P_6's in Algorithm 2, which are defined in complete analogy to similar vertices, edges, and so forth. The test for similar P_4's we implemented right after the test for similar triangles in line 19 of Algorithm 2, while the test for similar P_6's is included right after the test for a cutvertex in line 33. This modified version of the algorithm terminates and does indeed output 52 4-critical planar P_7-free graphs. □

Note that, as mentioned earlier, if the planarity condition is dropped there are infinitely many 4-critical P_7-free graphs. The counts of the 4-critical and 4-vertex-critical planar P_7-free graphs can be found in Table 6. The graphs themselves can be downloaded from `http://hog.grinvin.org/Critical`.

Vertices	4	6	7	8	9	10	11	12	13	total
Critical graphs	1	1	2	2	14	19	4	6	3	52
Vertex-critical graphs	1	1	6	2	65	347	6	19	15	462

Table 6: Counts of all planar 4-critical P_7-free graphs.

Acknowledgements

Several of the computations for this work were carried out using the Stevin Supercomputer Infrastructure at Ghent University. Jan Goedgebeur is supported by a Postdoctoral Fellowship of the Research Foundation Flanders (FWO).

References

[1] T. Böhme, B. Mohar, R. Škrekovski, and M. Stiebitz, *Subdivisions of large complete bipartite graphs and long induced paths in k-connected graphs*, Journal of Graph Theory **45** (2004), 270–274.

[2] J.M. Boyer and W.J. Myrvold, *On the Cutting Edge: Simplified $O(n)$ Planarity by Edge Addition*, Journal of Graph Algorithms and Applications **8** (2004), no. 2, 241–273.

[3] G. Brinkmann, K. Coolsaet, J. Goedgebeur, and H. Mélot, *House of Graphs: a database of interesting graphs*, Discrete Applied Mathematics **161** (2013), no. 1-2, 311–314, Available at `http://hog.grinvin.org/`.

[4] G. Brinkmann and M. Meringer, *The smallest 4-regular 4-chromatic graphs with girth 5*, Graph Theory Notes of New York **32** (1997), 40–41.

[5] D. Bruce, C.T. Hoàng, and J. Sawada, *A certifying algorithm for 3-colorability of P_5-free graphs*, Proceedings of the 20th International Symposium on Algorithms and Computation, Springer-Verlag, 2009, pp. 594–604.

[6] M. Chudnovsky, J. Goedgebeur, O. Schaudt, and M. Zhong, *Obstructions for three-coloring graphs with one forbidden induced subgraph*, Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016, 2016, pp. 1774–1783.

[7] M. Chudnovsky, N. Robertson, P. Seymour, and R. Thomas, *The Strong Perfect Graph Theorem*, Annals of Mathematics **164** (2006), no. 1, 51–229.

[8] J. Goedgebeur, Homepage of generator for 4-critical P_t-free graphs: `http://caagt.ugent.be/criticalpfree/`.

[9] P.A. Golovach, M. Johnson, D. Paulusma, and J. Song, *A survey on the computational complexity of colouring graphs with forbidden subgraphs*, 2014, arXiv:1407.1482v4 [cs.CC].

[10] P.A. Golovach, D. Paulusma, and J. Song, *Coloring graphs without short cycles and long induced paths*, Discrete Applied Mathematics **167** (2014), 107–120.

[11] P. Hell and S. Huang, *Complexity of coloring graphs without paths and cycles*, LATIN 2014: Theoretical Informatics, Springer, 2014, pp. 538–549.

[12] C.T. Hoàng, B. Moore, D. Recoskie, J. Sawada, and M. Vatshelle, *Constructions of k-critical P_5-free graphs*, Discrete Applied Mathematics **182** (2015), 91–98.

[13] M. Kamiński and V.V. Lozin, *Coloring edges and vertices of graphs without short or long cycles*, Contributions to Discrete Mathematics **2** (2007), 61–66.

[14] D. Král, J. Kratochvíl, Zs. Tuza, and G.J. Woeginger, *Complexity of coloring graphs without forbidden induced subgraphs*, Proceedings of the International Workshop on Graph-Theoretic Concepts in Computer Science 2001 (M.C. Golumbic, M. Stern, A. Levy, and G. Morgenstern, eds.), Lecture Notes in Computer Science, vol. 2204, 2001, pp. 254–262.

[15] F. Maffray and G. Morel, *On 3-colorable P_5-free graphs*, SIAM Journal on Discrete Mathematics **26** (2012), 1682–1708.

[16] B.D. McKay, *nauty User's Guide (Version 2.5)*, Technical Report TR-CS-90-02, Department of Computer Science, Australian National University. The latest version of the software is available at `http://cs.anu.edu.au/~bdm/nauty`.

[17] ———, *Isomorph-free exhaustive generation*, Journal of Algorithms **26** (1998), no. 2, 306–324.

[18] B.D. McKay and A. Piperno, *Practical graph isomorphism, II*, Journal of Symbolic Computation **60** (2014), 94–112.

[19] B. Randerath and I. Schiermeyer, *3-Colorability ∈ P for P_6-free graphs*, Discrete Applied Mathematics **136** (2004), no. 2, 299–313.

[20] P. Seymour, *private communication*.

[21] N. Sloane, The on-line encyclopedia of integer sequences: `http://oeis.org/`.

[22] D.P. Sumner, *Subtrees of a graph and the chromatic number*, Theory and Applications of Graphs (G. Chartrand, ed.), John Wiley, New York, 1981, pp. 557–576.

Appendix 1: Correctness testing

The correctness of our algorithm is proven in Theorem 7, but it is also very important to thoroughly verify the correctness of our implementation to minimize the chance of programming errors. Therefore this section is dedicated to the description of the extensive correctness tests which we performed on our implementation.

Since all of our consistency tests passed, we believe that this is strong evidence for the correctness of our implementation.

More specifically, we performed the following consistency tests to verify the correctness of our generator for k-critical $\mathcal{H}$-free graphs (i.e. Algorithm 1). The source code of this program can be downloaded from [8].

- As already mentioned in Section 3.1, we verified that our program indeed yields the same results when we apply it to the following cases which were already known before in the literature (by hand or by computer):
  - There are six 4-critical P_5-free graphs [5].
  - There are eight 5-critical (P_5, C_5)-free graphs [12].
  - The Grötzsch graph is the only 4-critical (P_6, C_3)-free graph [19].
  - There are four 4-critical (P_6, C_4)-free graphs [11].
- We developed a completely independent generator for k-critical P_t-free graphs. Here we started from a generator for all graphs (i.e. the program `geng` [16, 18]) and added routines to it to prune graphs which are not $(k-1)$-colorable or which are not P_t-free. The number of graphs generated by this program keeps growing, so it cannot terminate. But it still allowed us to independently generate all k-critical P_t-free graphs up to a given number of vertices:
  - We executed this program to generate all 4-critical P_6-free graphs up to 16 vertices and it indeed yielded the same 24 graphs from [6].
  - We executed this program to generate all 4-critical (P_7, C_3)-free graphs up to 16 vertices and it indeed yielded the same 7 critical graphs from Conjecture 9.
  - We executed this program to generate all 4-critical and 4-vertex-critical P_7-free graphs up to 13 vertices and it indeed yielded the same graphs from [6].
  - We executed this program to generate all 5-critical P_5-free graphs up to 15 vertices and in both cases it yielded the same 90 critical graphs.
- We modified our program to generate all P_t-free graphs and compared it with the known counts of P_t-free graphs for $t = 4, 5$ on the On-Line Encyclopedia of Integer Sequences [21] (i.e. sequences A000669 and A078564).
- We modified our program to generate all C_u-free graphs and compared it with the known counts of C_u-free graphs for $u = 4, 5$ on the On-Line Encyclopedia of Integer Sequences [21] (i.e. sequences A079566 and A078566).
- We modified our program to generate all k-colorable graphs and compared it with the known counts of k-colorable graphs for $k = 3, 4$ on the On-Line Encyclopedia of Integer Sequences [21] (i.e. sequences A076322 and A076323).

- We tested k-criticality in two independent ways and both methods yielded exactly the same results:
  1. By removing every vertex v once and verifying that $G-v$ is $(k-1)$-colorable and by removing all possible non-empty sets of edges E and verifying that every $G-E$ which is $\mathcal{H}$-free is $(k-1)$-colorable.
  2. By collecting all non-$(k-1)$-colorable graphs generated by the program and only outputting the graphs which do not contain any other graph from the list as a subgraph.
- We determined all k-vertex-critical graphs in two independent ways which both yielded exactly the same results:
  1. By modifying line 3 of Algorithm 2 so it tests for k-vertex-criticality instead of k-criticality.
  2. By recursively adding edges in all possible ways to the set of k-critical graphs (as long as the graphs remain k-vertex-critical) and testing if the resulting graphs are $\mathcal{H}$-free.

Appendix 2: Adjacency lists

This section contains the adjacency lists of various critical $\mathcal{H}$-free graphs mentioned in this article. These graphs and the lists of the vertex-critical graphs can also be found on the *House of Graphs* [3] at `http://hog.grinvin.org/Critical`.

Adjacency lists of 4-critical (P_7, C_3)-free graphs

Below are the adjacency lists of the seven 4-critical (P_7, C_3)-free graphs with at most 35 vertices from Conjecture 9.

1. {0 :1 4 6 8; 1 : 0 2 7 9; 2 : 1 3 6 10; 3 : 2 4 8 9; 4 : 0 3 5; 5 : 4 6 7 9 10; 6 : 0 2 5; 7 : 1 5 8; 8 : 0 3 7 10; 9 : 1 3 5; 10 : 2 5 8}
2. {0 :1 4 7 11; 1 : 0 2 8 9; 2 : 1 3 7 10; 3 : 2 4 9 11; 4 : 0 3 5 8; 5 : 4 6 9 10; 6 : 5 7 8 11; 7 : 0 2 6 9; 8 : 1 4 6 10; 9 : 1 3 5 7; 10 : 2 5 8 11; 11 : 0 3 6 10}
3. {0 :1 4 7 9 11; 1 : 0 2 8 10; 2 : 1 3 7 9; 3 : 2 4 8 11; 4 : 0 3 5 12; 5 : 4 6 9 10; 6 : 5 7 8; 7 : 0 2 6 12; 8 : 1 3 6 9 12; 9 : 0 2 5 8; 10 : 1 5 11 12; 11 : 0 3 10; 12 : 4 7 8 10}
4. {0 :1 4 7 9; 1 : 0 2 8 10; 2 : 1 3 7 11; 3 : 2 4 8 9; 4 : 0 3 5 10; 5 : 4 6 8 11; 6 : 5 7 9 10; 7 : 0 2 6 12; 8 : 1 3 5 12; 9 : 0 3 6 11; 10 : 1 4 6 12; 11 : 2 5 9 12; 12 : 7 8 10 11}
5. {0 :1 4 7 9 10; 1 : 0 2 8 11 12; 2 : 1 3 7 10 14; 3 : 2 4 8 9 11; 4 : 0 3 5 12 14; 5 : 4 6 8 10; 6 : 5 7 9 11; 7 : 0 2 6 13; 8 : 1 3 5 13; 9 : 0 3 6 12 14; 10 : 0 2 5 11 12; 11 : 1 3 6 10 14; 12 : 1 4 9 10 13; 13 : 7 8 12 14; 14 : 2 4 9 11 13}
6. {0 :1 4 7 10; 1 : 0 2 8 9 12; 2 : 1 3 7 11 13; 3 : 2 4 8 9 10; 4 : 0 3 5 12; 5 : 4 6 8 11; 6 : 5 7 9 13; 7 : 0 2 6 14; 8 : 1 3 5 13 14; 9 : 1 3 6 11 14; 10 : 0 3 11 13 14; 11 : 2 5 9 10 12; 12 : 1 4 11 13 14; 13 : 2 6 8 10 12; 14 : 7 8 9 10 12}
7. {0 :1 4 7 9 10; 1 : 0 2 8 11 12; 2 : 1 3 7 10 14; 3 : 2 4 8 9 11; 4 : 0 3 5 12 15; 5 : 4 6 8 10 14; 6 : 5 7 9 11 12; 7 : 0 2 6 13 15; 8 : 1 3 5 13 15; 9 : 0 3 6 13 14; 10 : 0 2 5 11 13; 11 : 1 3 6 10 15; 12 : 1 4 6 13 14; 13 : 7 8 9 10 12; 14 : 2 5 9 12 15; 15 : 4 7 8 11 14}

Adjacency lists of 4-critical (P_7, C_4)-free graphs

Below are the adjacency lists of the seventeen 4-critical (P_7, C_4)-free graphs from Theorem 8.

1. {0 : 1 2 3; 1 : 0 2 3; 2 : 0 1 3; 3 : 0 1 2}
2. {0 : 1 2 5; 1 : 0 3 5; 2 : 0 4 5; 3 : 1 4 5; 4 : 2 3 5; 5 : 0 1 2 3 4}
3. {0 : 1 2 4; 1 : 0 3 5; 2 : 0 4 6; 3 : 1 5 6; 4 : 0 2 6; 5 : 1 3 6; 6 : 2 3 4 5}

4. {0 : 1 2 7; 1 : 0 3 7; 2 : 0 4 7; 3 : 1 5 7; 4 : 2 6 7; 5 : 3 6 7; 6 : 4 5 7; 7 : 0 1 2 3 4 5 6}
5. {0 : 1 2 6; 1 : 0 3 5; 2 : 0 4 6 7; 3 : 1 5 7; 4 : 2 5 6; 5 : 1 3 4; 6 : 0 2 4 7; 7 : 2 3 6}
6. {0 : 1 2 7; 1 : 0 3 6 8; 2 : 0 4 7; 3 : 1 5 8; 4 : 2 6 8; 5 : 3 7 8; 6 : 1 4 8; 7 : 0 2 5; 8 : 1 3 4 5 6}
7. {0 : 1 2 7; 1 : 0 3 6; 2 : 0 4 7; 3 : 1 5 6 8; 4 : 2 6 7; 5 : 3 7 8; 6 : 1 3 4; 7 : 0 2 4 5 8; 8 : 3 5 7}
8. {0 : 1 2 8; 1 : 0 3 6 8; 2 : 0 4 8; 3 : 1 5 6; 4 : 2 6 7; 5 : 3 6 7; 6 : 1 3 4 5 7; 7 : 4 5 6; 8 : 0 1 2}
9. {0 : 1 2 8; 1 : 0 3 8; 2 : 0 4 6 8; 3 : 1 5 7; 4 : 2 6 9; 5 : 3 7 9; 6 : 2 4 9; 7 : 3 5 9; 8 : 0 1 2; 9 : 4 5 6 7}
10. {0 : 1 2 7 9; 1 : 0 3 6 9; 2 : 0 4 7; 3 : 1 5 8 9; 4 : 2 6 8; 5 : 3 7 8; 6 : 1 4 9; 7 : 0 2 5; 8 : 3 4 5; 9 : 0 1 3 6}
11. {0 : 1 2 7; 1 : 0 3 6; 2 : 0 4 7 9; 3 : 1 5 8; 4 : 2 6 9; 5 : 3 7 8; 6 : 1 4 9; 7 : 0 2 5 8; 8 : 3 5 7; 9 : 2 4 6}
12. {0 : 1 2 7; 1 : 0 3 6 8; 2 : 0 4 7; 3 : 1 5 8; 4 : 2 6 9; 5 : 3 7 8 9; 6 : 1 4 9; 7 : 0 2 5; 8 : 1 3 5; 9 : 4 5 6}
13. {0 : 1 2 7; 1 : 0 3 6 8; 2 : 0 4 7 9; 3 : 1 5 8; 4 : 2 6 9; 5 : 3 7 8; 6 : 1 4 9; 7 : 0 2 5; 8 : 1 3 5; 9 : 2 4 6}
14. {0 : 1 2 7; 1 : 0 3 6 8 9; 2 : 0 4 7; 3 : 1 5 8; 4 : 2 6 9; 5 : 3 7 8; 6 : 1 4 9; 7 : 0 2 5; 8 : 1 3 5; 9 : 1 4 6}
15. {0 : 1 2 7; 1 : 0 3 6 8; 2 : 0 4 7; 3 : 1 5 8; 4 : 2 6 8 9; 5 : 3 7 9; 6 : 1 4 8 9; 7 : 0 2 5; 8 : 1 3 4 6; 9 : 4 5 6}
16. {0 : 1 2 7; 1 : 0 3 6 8; 2 : 0 4 7 9; 3 : 1 5 8 9; 4 : 2 6 7 9; 5 : 3 7 8; 6 : 1 4 8; 7 : 0 2 4 5; 8 : 1 3 5 6; 9 : 2 3 4}
17. {0 : 1 2 9 10; 1 : 0 3 6 12; 2 : 0 4 7 8; 3 : 1 5 11; 4 : 2 6 11 12; 5 : 3 7 8; 6 : 1 4 12; 7 : 2 5 8; 8 : 2 5 7; 9 : 0 10 11; 10 : 0 9 11; 11 : 3 4 9 10; 12 : 1 4 6}

Adjacency lists of 4-critical (P_7, C_5)-free graphs

Below are the adjacency lists of the six 4-critical (P_7, C_5)-free graphs from Theorem 8.

1. {0 : 1 2 3; 1 : 0 2 3; 2 : 0 1 3; 3 : 0 1 2}
2. {0 : 2 3 4 5; 1 : 3 4 5 6; 2 : 0 4 5 6; 3 : 0 1 5 6; 4 : 0 1 2 6; 5 : 0 1 2 3; 6 : 1 2 3 4}
3. {0 : 1 6 7; 1 : 0 2 7; 2 : 1 3 7; 3 : 2 4 7; 4 : 3 5 7; 5 : 4 6 7; 6 : 0 5 7; 7 : 0 1 2 3 4 5 6}
4. {0 : 1 6 7; 1 : 0 2 7 8; 2 : 1 3 8; 3 : 2 4 8; 4 : 3 5 8; 5 : 4 6 8; 6 : 0 5 7 8; 7 : 0 1 6; 8 : 1 2 3 4 5 6}
5. {0 : 1 6 7; 1 : 0 2 7 9; 2 : 1 3 9; 3 : 2 4 8 9; 4 : 3 5 8; 5 : 4 6 8; 6 : 0 5 7; 7 : 0 1 6; 8 : 3 4 5; 9 : 1 2 3}
6. {0 : 1 6 7; 1 : 0 2 7 8 12; 2 : 1 3 9 12; 3 : 2 4 9; 4 : 3 5 9; 5 : 4 6 8; 6 : 0 5 7 8 12; 7 : 0 1 6; 8 : 1 5 6 10 11; 9 : 2 3 4; 10 : 8 11 12; 11 : 8 10 12; 12 : 1 2 6 10 11}

Adjacency list of smallest 4-critical P_{12}-free graph with girth five

Below is the adjacency list of the smallest 4-critical P_{12}-free graph with girth 5 from Theorem 10.(b).

- {0 : 1 4 8 11 17; 1 : 0 2 13 14 15; 2 : 1 3 12; 3 : 2 4 16 18 19; 4 : 0 3 5 20; 5 : 4 6 14; 6 : 5 7 12 13 17 18; 7 : 6 8 15 19; 8 : 0 7 9; 9 : 8 10 14 18 20; 10 : 9 11 13 16; 11 : 0 10 12 19; 12 : 2 6 11 20; 13 : 1 6 10; 14 : 1 5 9 19; 15 : 1 7 16 20; 16 : 3 10 15 17; 17 : 0 6 16; 18 : 3 6 9; 19 : 3 7 11 14; 20 : 4 9 12 15}