Flood-It with Jewelry – Characterizing the Game Complexity for Cograph Generalizations

Martin Darmüntzel
University of Rostock, Germany

Christian Rosenke
University of Rostock, Germany

Mark Scheibner
Independent Researcher, Rostock, Germany

Abstract

Flood-It is a single-player game played on a precolored graph G, where the objective is to make G monochromatic using as few *flooding* moves as possible. In each move, a color c is selected and all vertices reachable from a fixed pivot vertex via a monochromatic path are recolored with c. In the *free* variant, the pivot may be chosen anew in every move.

Deciding whether a graph can be made monochromatic in at most k moves is NP-complete for both variants, fixed and free. This hardness persists even under strong structural restrictions such as split graphs and trees. The *Free Flood-It* variant is generally considered more difficult than its fixed-pivot counterpart, as it remains hard on several graph classes where the latter becomes tractable, including co-comparability and AT-free graphs.

Cographs, that is, P_4-free graphs, are among the few classes on which even Free Flood-It is solvable in polynomial time and therefore serve as our starting point. We consider the ten natural one-vertex extensions of P_4 – referred to as *jewels* – and study the complexity of both flooding games on the 1024 graph classes obtained by forbidding subsets of these graphs as induced subgraphs.

Our main contribution is a polynomial-time algorithm for Free Flood-It on graphs that are free of the three jewels bull, gem, and P_5, covering 128 of the 1024 classes. In addition, we prove that both variants remain NP-complete on thin-spider graphs, which exclude the eight jewels banner, co-banner, chair, gem, house, kite, P_5, and C_5, thereby establishing hardness for 256 additional classes.

Combined with known algorithms and hardness results, our work determines the complexity of both Flood-It variants for 896 of the 1024 considered graph classes.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Discrete optimization; Mathematics of computing → Graph algorithms; Mathematics of computing → Graph coloring

Keywords and phrases Flood-It, Free Flood-It, cograph generalizations, polynomial time algorithms, NP-completeness

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.40

Related Version *Full Version*: `https://arxiv.org/abs/2606.23837` [4]

Acknowledgements The authors thank the anonymous reviewers for their helpful comments and suggestions.

Declaration on GenAI/LLM use We note that the AI system ChatGPT, in particular the models *GPT-5 mini* and *GPT-5.3*, has been used in this work for the only purpose of improving the presentation and readability of the text, as the authors are not native English speakers. No other use of ChatGPT or its subroutines has been made in this paper. In particular, no AI framework has contributed to any creative decisions, the development of original results, or the construction of rigorous mathematical proofs, all of which were carried out entirely by the authors.

1 Introduction

Flood-It and several of its variants have been studied extensively between 2010 and 2020. This substantial body of work has shown that computing optimal strategies remains computationally hard, even under severe restrictions on the input. For comprehensive overviews of known hardness results and the comparatively few polynomial-time solvable cases, we refer the interested reader to the excellent surveys by Fellows et al. [7] and Fleischer and Woeginger [8].

While the original flood-filling game is played on precolored grids, the notion of a playfield has been generalized in the literature to arbitrary graphs to better capture and analyze the underlying algorithmic challenges. In this paper, we focus on single-player variants, where each game is described by a *move order*, that is, a sequence of moves, each of which floods a chosen color c onto all vertices w that are reachable from a pivot vertex v via a monochromatic path, that is, a path from v to w whose vertices all have the same color at the time of the move. The goal of the game is to make the playfield graph G monochromatic; accordingly, any move order that achieves this is called a *strategy* for G. In the variant called *Free Flood-It* the player may *freely* choose the pivot vertex in every move. Naturally, if the pivot is a *fixed* vertex specified as part of the input, we obtain the *(fixed) Flood-It* game. In both variants, the objective is to find a strategy of minimum length. There are corresponding decision variants of these problems that additionally take an integer k as input and ask whether there exists a strategy using at most k moves. Since the relevant variant, optimization or decision, will always be clear from context, this paper does not introduce separate notations.

It is known that the computational problems underlying these flood-filling games, Flood-It and Free Flood-It, remain intractable even under severe input restrictions such as *split graphs* [8, 10]. In particular, both problems are NP-hard when the input playfield is a *tree* [6] or a *grid* [2], even if it is initially colored with only three colors.

Only very few non-trivial graph classes are known on which Flood-It (FIt, for short) is tractable, including *co-comparability* [8] and *AT-free* graphs [10]. In contrast, Free Flood-It (FFIt, for short) remains hard even on these classes. For many years, polynomial-time solvability of the free variant has been known only for very restricted grids [12], as well as for simple paths and cycles [13]. However, for the second powers of paths and cycles [5], FFIt becomes NP-hard again.

As part of the algorithmic techniques developed in this paper, we show that deciding whether a graph G can be made monochromatic using exactly $k-1$ free moves – one for each of the k colors used in G except the last – can be done in polynomial time, independent of the presence of certain structures in the graph and the initial coloring. Indeed, if the playfield initially uses k distinct colors, any strategy requires at least $k-1$ moves, since each move can eliminate at most one color. Given the apparent resistance of FFIt to efficient algorithms, it is somewhat surprising that deciding the existence of a $k-1$-move strategy is always in polynomial time.

The main contribution of this paper, however, is motivated by the recent result [15] that FFIt is polynomial-time solvable on *cographs* – the graphs without induced P_4. This naturally raises the question of how the restriction of forbidding induced P_4 can be relaxed without losing tractability. In [15], this question is addressed by considering the extension of P_4 by an independent vertex and deriving a significantly more involved polynomial-time algorithm for graphs that are free of induced co-gems; see Figure 1 (f).

It is well known that the co-gem is not the only graph generalizing P_4, and that there are exactly ten non-isomorphic one-vertex extensions of P_4; see, for example, [1]. As illustrated in Figure 1, these graphs can be viewed as interpolating between the co-gem and its complement, the gem, by selecting a subset of the four edges that connect the additional vertex to the underlying P_4. Motivated by this interpretation, we refer to these graphs collectively as *jewels*.

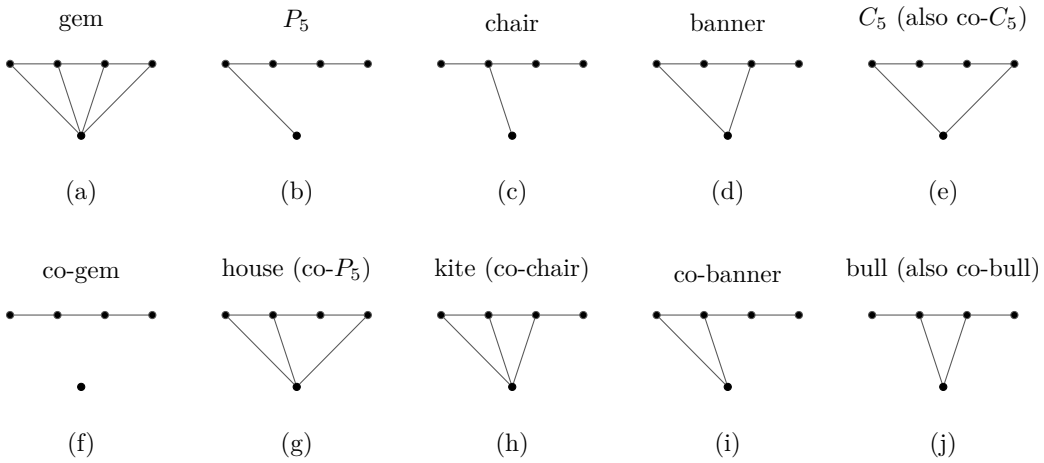

Figure 1 The jewels, one-vertex extensions of the P_4, given with common name and complement graph relation.

Now, if we consider arbitrary subsets of these small graphs as forbidden induced subgraphs, we obtain a large collection of natural generalizations of cographs. Based on this, we continue the line of inquiry initiated in [15] in a natural way and ask, for every possible subset $\mathcal{G}$ of the jewels, what the complexity of (Free) Flood-It is when the input is restricted to the graph class defined by forbidding all graphs in $\mathcal{G}$. Since there are ten jewels, this framework gives rise to a family of $2^{10} = 1024$ graph classes. The empty set of forbidden graphs yields the class of all graphs, whereas forbidding all ten jewels results in the class of cographs (together with the single graph P_4).

Given the size of this family, it is natural to revisit existing results in order to preemptively resolve as many cases as possible. Starting with the tractable cases, and noting that adding further forbidden induced subgraphs cannot destroy tractability, the recent result [15] already implies that Free Flood-It is solvable in polynomial time on the 512 classes that include co-gem in the set of forbidden graphs.

▶ **Corollary 1.** Free Flood-It *can be solved in polynomial time on every graph class whose set of forbidden induced jewels contains co-gem.*

A closer look at [15] reveals that the paper studies the related Miniature Painting problem and polynomial-time solvability of Free Flood-It follows only as a corollary. This implication does not readily extend to the fixed-pivot variant. Nevertheless, no graph class is currently known on which the free variant is tractable while the fixed variant remains hard. This motivates the conjecture that Flood-It is also polynomial-time solvable on co-gem-free graphs.

The main result of the present paper follows the general direction of the previous work in [15] and develops new algorithmic techniques for graph classes defined by forbidding P_5, bull, and gem. Specifically, Section 3 analyzes the structure of $(P_5, \text{bull}, \text{gem})$-free graphs and

derives properties that enable efficient algorithms. These insights are then used in Sections 4 and 5 to obtain a polynomial-time algorithm for Free Flood-It on this class. This leads to the following theorem.

▶ **Theorem 2.** Free Flood-It *can be solved in polynomial time on every graph class whose forbidden induced jewels include* P_5*, bull, and gem.*

Our result is stated only for FFIt, as the polynomial-time algorithm for AT-free graphs from [11] already implies tractability of the fixed-pivot variant on $(P_5, \text{bull}, \text{gem})$-free graphs, which are AT-free. FFIt, however, remains hard on AT-free graphs [10]; thus, this paper establishes a boundary between tractability and hardness within this graph class. The main result covers 128 of the 1024 classes of cograph generalizations. However, beyond the 512 classes already addressed, only 64 are new.

To identify classes with intractable flooding strategies, we take a closer look at the NP-completeness proofs by Fleischer and Woeginger [8] and by Fukui et al. [10]. These reveal that their reductions for split graphs satisfy strong structural constraints. In general, split graphs are free of induced P_5; see Figure 1 (b). However, the constructed gadget graphs are, in fact, also free of house, banner, co-banner, C_5, kite, and gem. Consequently, both flooding games remain NP-complete on every graph class defined by forbidding an arbitrary subset of these jewels.

As a second contribution of this paper, Section 6 refines these proof techniques by employing *thin spiders* as reduction gadgets. These split graphs are, in addition to the seven jewels listed above, also free of induced diamond and claw; see Figure 2. Since chair is a natural generalization of claw, it can be added to the list of forbidden graphs. As a result, Section 6 establishes NP-completeness for both variants of Flood-It on a broad range of classes, summarized in the following theorem.

▶ **Theorem 3.** Flood-It *and* Free Flood-It *are* NP*-complete on the class of* X*-free graphs for every* $X \subseteq \{P_5, \textit{house}, \textit{chair}, \textit{kite}, \textit{banner}, \textit{co-banner}, C_5, \textit{gem}\}$*.*

Since this covers all subsets of eight of the ten jewels, the theorem resolves the complexity question for another 256 classes in our family. Implicitly, 128 of them have been known before from the results of [8, 10].

Another source of hardness is provided by a result of Fellows et al. [6], who show that both game variants are NP-complete when restricted to trees. Because none of the cyclic jewels can appear as induced subgraphs of trees, forbidding them does not restrict this class. Hence, their result directly yields the following corollary.

▶ **Corollary 4.** Flood-It *and* Free Flood-It *are* NP*-complete on the class of* X*-free graphs for every* $X \subseteq \{\textit{banner}, \textit{co-banner}, \textit{bull}, C_5, \textit{kite}, \textit{house}, \textit{gem}\}$*.*

Among the 128 classes covered by this corollary, those whose forbidden set contains bull are not addressed by Theorem 3. This accounts for 64 additional resolved cases.

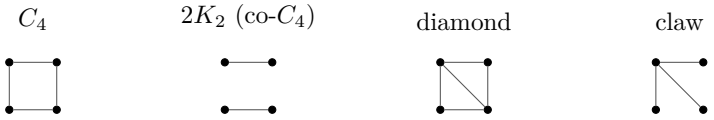

Figure 2 Additional graphs appearing in this paper.

At this stage, 896 of the 1024 graph classes can be considered settled. The results in this paper account for 192 previously unresolved classes, representing substantial progress. The remaining 128 graph classes are yet to be classified. We suspect that resolving them will require fundamentally different approaches, both in terms of reduction gadgets for NP-completeness proofs and structural insights for polynomial-time algorithms.

After the next section introduces basic concepts and notation, the remainder of the paper is organized as follows. Section 3 develops structural properties of $(P_5, \text{bull}, \text{gem})$-free graphs, which are subsequently exploited in Sections 4 and 5 to obtain a polynomial-time algorithm for FFIt on this graph class. In Section 6, we prove that both flood-filling games are NP-complete on thin spiders and derive Theorem 3 as a direct consequence. Finally, Section 7 concludes the paper. Due to space constraints, some proofs are omitted.

2 Preliminaries

Before presenting our results, we fix the notation used throughout the paper. We mostly rely on standard mathematical and graph-theoretic notation. For sets A and B, we denote the *union*, *difference*, and *intersection* by $A + B$, $A - B$, and $A \cap B$, respectively. Singleton sets $\{x\}$ are identified with their only element x, whenever this is clear from the context.

Graph Notation

All graphs considered in this paper are finite, undirected, and without multiple edges or loops. Figure 1 depict the *jewels*, that is, all graphs that can be obtained by extending a P_4 with one additional vertex, called the *tip*. As explained in the introduction, these graphs form the basis for a spectrum of graph classes, each representing a certain generalization of cographs. Moreover, Figure 2 shows some small graphs that are also referenced in this paper.

For every vertex $x \in V(G)$, we define the *open neighborhood* $N_G(x) = \{y \mid xy \in E(G)\}$ and the *closed neighborhood* $N_G[x] = N_G(x) + x$. Since, throughout the paper, the graph G is always clear from the context, we omit it as a subscript and simply write $N(x)$ and $N[x]$. We say that a vertex x *sees* a vertex y, if $y \in N(x)$. If $|N(x)| = 1$ then x is called a *leaf* and the only vertex in $N(x)$ is the *parent* of x. For a subset $V \subseteq V(G)$, we define $N[V] = \bigcup_{v \in V} N[v]$. A *dominating set* D of a graph G is a subset of $V(G)$ such that $N[D] = V(G)$.

If G is a graph, every subgraph $H = (V, E)$ of G is specified by a vertex subset $V \subseteq V(G)$ and an edge subset $E \subseteq E(G)$ that contains only edges with both endpoints in V. A subgraph H is *induced*, if E contains exactly all edges $vw \in E(G)$ with $v, w \in V$. For any vertex subset $V \subseteq V(G)$, we write $G[V]$ for the induced subgraph of G on V. We use $G - V$ as a shorthand for $G[V(G) - V]$. Graphs G and H are *isomorphic*, if there exists a bijection $\sigma \colon V(G) \to V(H)$ such that $xy \in E(G)$ if and only if $\sigma(x)\sigma(y) \in E(H)$. If no induced subgraph of G is isomorphic to a graph H, then G is called *H-free*. A subgraph H of G, induced or not, is said to *dominate* G, if $V(H)$ is a dominating set of G.

For graphs G and H with disjoint vertex sets and a vertex $v \in V(G)$, the *substitution* $G[v \leftarrow H]$ of v with H is the graph with vertex set $V(H) + V(G) - v$ and edge set $E(H) + E(G) - \{vw \mid w \in N_G(v)\}$, together with all edges uw for $u \in V(H)$ and $w \in N_G(v)$. In this way, the graph H replaces the single vertex v in G and becomes a *module* of $G[v \leftarrow H]$, that is, a subgraph whose vertices are either all seen or all unseen by every vertex in $V(G) - v$. Since the order of substituting graphs $H_1, \ldots, H_k$ for pairwise distinct vertices $v_1, \ldots, v_k$ of a graph G is irrelevant, we write $G[v_1, \ldots, v_k \leftarrow H_1, \ldots, H_k]$ to denote the result of these substitutions in arbitrary order.

Flood Filling for Graphs

For clarity, we refer to any surjective function $f\colon V(G) \to C$ with color set C as a *flooding* of a graph G, rather than using the more common term *coloring*. For every color $c \in C$, we write $f^{-1}(c) = \{v \in V(G) \mid f(v) = c\}$ for the set of vertices with color c. Given a flooding f of G, a vertex subset $M \subseteq V$ is *monochromatic*, if $M \subseteq f^{-1}(c)$ for some color c. A monochromatic set M is called a *color component* of G, if M is connected in G and no proper superset of M is both monochromatic and connected. A color $c \in C$ is said to be *connected* with respect to f, if $f^{-1}(c)$ forms a single color component. Otherwise, it is *disconnected*. Moreover, for every vertex $v \in V(G)$, we denote by $f^{-1}(v)$ the color component containing v, which is a subset of $f^{-1}(c)$ for the color $c = f(v)$. We also call $f^{-1}(v)$ the *flooded area* of v.

With these notions in place, we formally define the flood filling problems introduced earlier. Let G be a graph with initial flooding $f_0\colon V(G) \to C$. A (possibly empty) *free move order* $(v_1, c_1), \ldots, (v_s, c_s)$ consists of moves (v_i, c_i), where each $v_i \in V(G)$ is a *pivot vertex* and each $c_i \in C$ is a *flooding color*. Since free move orders are the main subject of this paper, we often just call them *move order*. The result of a move (v_i, c_i) is a new flooding $f_i\colon V(G) \to C$ defined by

$$f_i(v) = \begin{cases} c_i, & \text{if } v \in f_{i-1}^{-1}(v_i), \\ f_{i-1}(v), & \text{otherwise.} \end{cases}$$

In contrast to free move orders, a *fixed move order* with pivot p satisfies $p = v_1 = \cdots = v_s$. Because the pivot does not change, fixed move orders can be specified simply by the sequence of colors $c_1, \ldots, c_s$.

The purpose of a move (v_i, c_i) is typically to change the color of all vertices in the color component $f_{i-1}^{-1}(v_i)$ to c_i so that, with respect to f_i, this component merges with parts of $f_{i-1}^{-1}(c_i)$, that is, with vertices that are already colored c_i. To formalize this growth in *territory*, we say that the move (v_i, c_i) *conquers* the vertices $f_i^{-1}(v_i) - f_{i-1}^{-1}(v_i)$ from the pivot v_i. This notion is particularly intuitive in the fixed flooding setting, where all moves conquer territory from the same pivot.

A move order, fixed or free, is called a *strategy* for G, f_0, and, in the fixed case, p, if the resulting flooding f_s is constant. The Flood-It problem asks, given G, f_0, and a pivot p, for a shortest fixed strategy, that is, one using the minimum possible number of moves. Analogously, Free Flood-It asks for a shortest free strategy. An example instance of Flood-It is shown Figure 3.

3 Restrictions for Graphs free of the Jewels P_5, bull and gem

This section prepares the main result of the paper, which concerns the classification of a large family of cograph generalizations that admit polynomial-time algorithms. As outlined in the introduction, a substantial portion of these classes has already been identified in the preceding work [15] on the closely related Miniature Painting problem. A central insight behind the previous approach is that the considered co-gem-free graphs cannot expand arbitrarily and that their vertices are, in a certain sense, restricted to the space around small subgraphs. More precisely, if π is an induced P_4 in a co-gem-free graph G, then π forms a dominating set and thus serves as a kind of *hub*. From the perspective of Free Flood-It, this means that, since all vertices of G lie in the neighborhood of π, a meaningful strategy may first conquer π and then flood each remaining color once in order to make G monochromatic. In fact, we essentially show that there exists a shortest strategy for G that conquers an induced P_4, say π, within at most twelve moves and subsequently uses π as a hub for all remaining moves. Such strategies can then be brute-forced, yielding a polynomial-time algorithm.

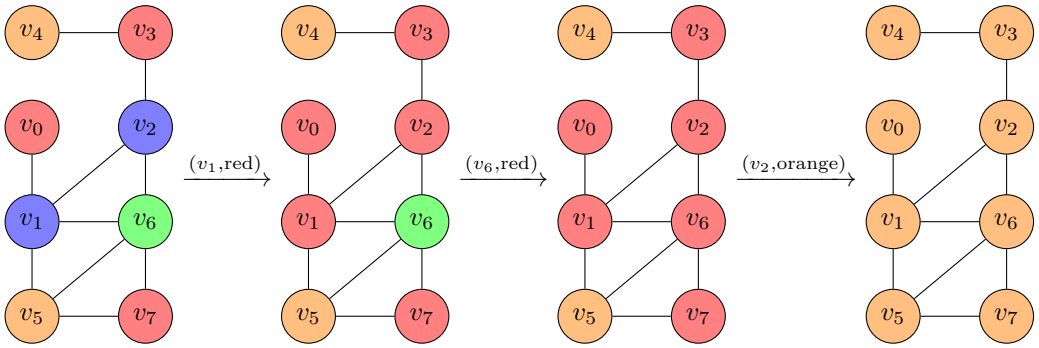

Figure 3 A game of Free Flood-It on a graph G with an initial flooding f_0 assigning the colors red, blue, orange, green. Applying the move (v_1, red) results in the flooding f_1 with $f_1(v_1) = f_1(v_2) = \text{red}$ and $f_1(v) = f_0(v)$ for all other vertices, eliminating blue in the process. The second move eliminates green, and the last red. The depicted move order $(v_1, \text{red}), (v_6, \text{red}), (v_2, \text{orange})$ is a strategy for G, f_0. In fact, it is even a shortest strategy.

However, while the existence of small, connected dominating sets in co-gem-free graphs provides a powerful starting point, it is by no means a turnkey solution for flood filling problems. Considerable effort is still required to exploit these structural properties algorithmically. Accordingly, the aim of the present section is to take a first step toward a similar understanding for a different collection of jewel-free graphs. Specifically, we establish analogous conditions for $(P_5, \text{bull}, \text{gem})$-free graphs, identifying a new type of hub: a localized graph structure that, in a suitable sense, restricts all vertices of the graph to the space around it.

As every $(P_5, \text{bull}, \text{gem})$-free graph G can be viewed as a generalized cograph, our overarching objective is to recover the remnants of cograph structure that persist within G. In fact, we will show that, in most cases, G decomposes into a cyclic arrangement of cographs. To stay within our notional scheme and since cographs are, in the strict sense, free of jewels, we refer to such a structure as a *bead bracelet* with the individual cographs being just *beads*. More precisely, for a cycle C_k with $k \geq 3$, vertices $v_0, v_1, \ldots, v_{k-1}$, and non-empty cographs $H_0, H_1, \ldots, H_{k-1}$, the graph $C_k[v_0, v_1, \ldots, v_{k-1} \leftarrow H_0, H_1, \ldots, H_{k-1}]$ that substitutes the cycle vertices with the beads $H_0, H_1, \ldots, H_{k-1}$ is called a *k-bead bracelet*. The following lemma formalizes the connection between this concept and $(P_5, \text{bull}, \text{gem})$-free graphs.

▶ **Lemma 5.** *If G is a connected $(P_5, \text{bull}, \text{gem})$-free graph that contains an induced C_5, then G is a 5-bead bracelet.*

Proof. Consider any induced C_5 in G and let v_0, v_1, v_2, v_3, v_4 be its vertices in cyclic order. For every $i \in \{0, \ldots, 4\}$, we define $N_i = N(v_{i-1}) \cap N(v_{i+1})$, where, throughout the proof, indices are taken modulo five. We prove that $H_0 = G[N_0], \ldots, H_4 = G[N_4]$ are cographs, that $N_0, \ldots, N_4$ partition $V(G)$, and that the edges between these sets follow the C_5. That is, for $x \in N_i$ and $y \in N_j$ with $0 \leq i < j < 5$, we have $xy \in E(G)$ if and only if $j - i \in \{1, 4\}$.

We first show that the N-sets are mutually disjoint. Suppose, for a contradiction, that there is a vertex $x \in N_i \cap N_j$ with $0 \leq i < j < 5$. By symmetry, it is enough to consider the cases $j = i+1$ and $j = i+2$. If $j = i+1$, then x sees v_{i-1}, v_i, v_{i+1}, and v_{i+2} as in Figure 4 (a). Hence, x is the tip of an induced gem with P_4 on $v_{i-1}, v_i, v_{i+1}, v_{i+2}$, a contradiction. So let $j = i + 2$. Then x sees v_{i-1}, v_{i+1}, and v_{i-2} as shown in Figure 4 (b). That $xv_i \notin E(G)$ and $xv_{i+2} \notin E(G)$ follows from the previous case. But then there is an induced bull with P_4 on $v_i, v_{i-1}, v_{i-2}, v_{i+2}$ and tip x, again a contradiction. Hence, $N_0, \ldots, N_4$ are mutually disjoint.

Before proving that the N-sets cover $V(G)$, we record one useful consequence. For every $i \in \{0, \ldots, 4\}$ and every $x \in N_i$, the vertex x is adjacent to v_{i-1} and v_{i+1} by definition, but not to v_{i-2} or v_{i+2}. Indeed, if $xv_{i-2} \in E(G)$, then $x \in N(v_{i-2}) \cap N(v_{i+1}) = N_{i+2}$, contradicting the disjointness of the N-sets. Similarly, if $xv_{i+2} \in E(G)$, then $x \in N(v_{i+2}) \cap N(v_{i-1}) = N_{i-2}$, again a contradiction.

Now it remains to show that the N-sets cover $V(G)$. Suppose, for a contradiction, that there is a vertex $v \in V(G) - (N_0 + N_1 + N_2 + N_3 + N_4)$. Since G is connected, there is a shortest path from v to $(N_0 + N_1 + N_2 + N_3 + N_4)$. Let w be the endpoint of this path adjacent to the C_5, and let x be the neighbor of w on this path. Possibly, $x = v$. By the choice of the path, $x \notin N_i$ for any $i \in \{0, \ldots, 4\}$, and in particular $x \notin \{v_0, \ldots, v_4\}$. We show that this is impossible.

First, x cannot see two or more vertices of the C_5. If x sees two non-consecutive vertices of the C_5, then x belongs to some N_i, contrary to the choice of x. If x sees exactly two consecutive vertices v_i and v_{i+1} of the C_5, then, as Figure 4 (c) demonstrates, there is an induced bull with P_4 on $v_{i-1}, v_i, v_{i+1}, v_{i+2}$ and tip x. Hence, x sees at most one vertex of the C_5. But Figure 4 (d) shows that, if x sees exactly one vertex v_i of the C_5, then $x, v_i, v_{i+1}, v_{i+2}, v_{i-2}$ induce a P_5, a contradiction. Thus, x sees no vertex of the C_5.

However, x is adjacent to w, and $w \in N_i$ for some $i \in \{0, \ldots, 4\}$. Hence, w sees v_{i-1} and v_{i+1}, but not v_{i-2} or v_{i+2}. This case is displayed in Figure 4 (e), where it is easy to see that $x, w, v_{i+1}, v_{i+2}, v_{i-2}$ induce a P_5, again a contradiction. Therefore, no such vertex v exists, and $N_0, \ldots, N_4$ partition $V(G)$.

We next prove that the edges between the N-sets follow the C_5. Let $x \in N_i$ and $y \in N_j$ with $0 \leq i < j < 5$. First assume that $j - i \in \{1, 4\}$. By symmetry, we may suppose that $j = i + 1$. We show that $xy \in E(G)$. If $x = v_i$, then $y \in N_{i+1} = N(v_i) \cap N(v_{i+2})$, so $xy \in E(G)$. Symmetrically, if $y = v_{i+1}$, then $xy \in E(G)$. Hence, we may assume that $x \in N_i - v_i$ and $y \in N_{i+1} - v_{i+1}$. Suppose, for a contradiction, that $xy \notin E(G)$. By definition and by the observation above, x sees v_{i-1} and v_{i+1}, but not v_{i-2} or v_{i+2}. Similarly, y sees v_i and v_{i+2}, but not v_{i-1} or v_{i+3}. See Figure 4 (f) for a visualization. Since $v_{i+3} = v_{i-2}$, the vertices $y, v_{i+2}, v_{i+3}, v_{i-1}, x$ induce a P_5, a contradiction. Thus, $xy \in E(G)$.

Now assume that $j - i \in \{2, 3\}$. By symmetry, we may suppose that $j = i + 2$. We show that $xy \notin E(G)$. Suppose, for a contradiction, that $xy \in E(G)$. If $x = v_i$, then y sees v_i, v_{i+1}, and v_{i+3}. Thus, $y \in N_{i+2} \cap N_{i-1}$, contradicting the disjointness of the N-sets. Symmetrically, $y \neq v_{i+2}$. Hence, we may assume that $x \in N_i - v_i$ and $y \in N_{i+2} - v_{i+2}$. By the observation above, x sees v_{i-1} and v_{i+1}, but not v_{i-2} or v_{i+2}. Similarly, y sees v_{i+1} and v_{i-2}, but not v_i or v_{i-1}. If $xv_i \in E(G)$, as in Figure 4 (g), then x is the tip of an induced gem with P_4 on v_{i-1}, v_i, v_{i+1}, y. If $xv_i \notin E(G)$, then there is an induced bull with P_4 on v_i, v_{i+1}, y, v_{i-2} and tip x, like Figure 4 (h) demonstrates. Both cases are impossible. Hence, $xy \notin E(G)$.

It remains to prove that $H_0, \ldots, H_4$ are cographs. We show this by proving the equivalent claim that they are P_4-free. Any induced P_4 in H_i would induce a gem with tip v_{i+1}. Therefore, each H_i is a cograph, and the proof is complete. ◀

The underlying C_5 keeps the surrounding cographs in proximity and therefore serves as an effective flooding hub. Building on this observation, Section 5 develops a new polynomial-time approach that exploits this hub structure. On the other hand, there also exist $(P_5, \text{bull}, \text{gem}, C_5)$-free graphs G. Such graphs behave even more like cographs. In particular, connected cographs are typically rich in dominating edges. This property carries over to the present generalization and, consequently, every such graph contains at least one dominating edge.

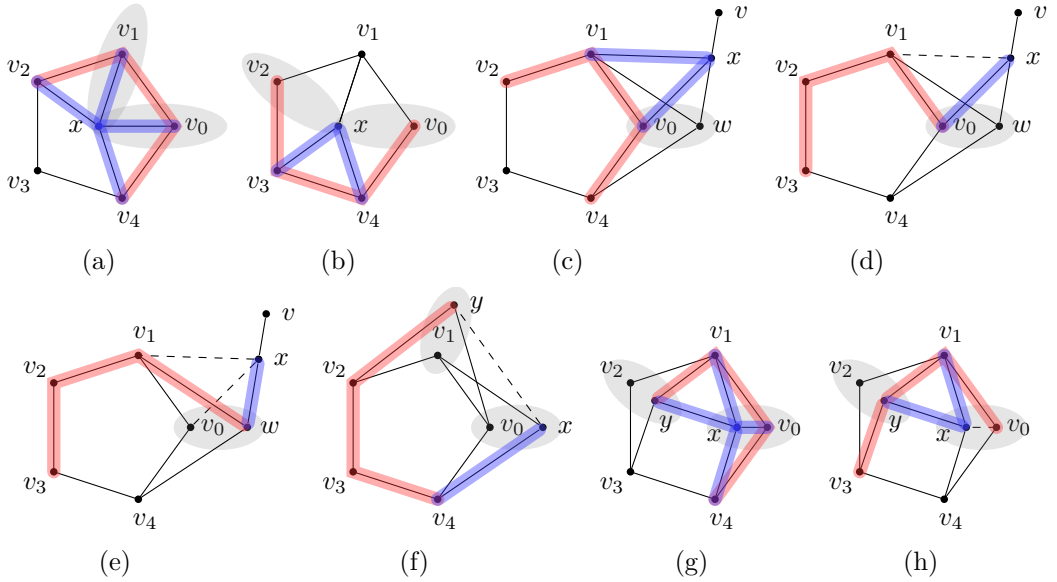

Figure 4 Illustrations for the proof of Lemma 5. For visual clarity we set $i = 0$ and $j = i + 1$ whenever $j - i \in \{1, 4\}$ and $j = i + 2$ whenever $j - i \in \{2, 3\}$. Edges highlighted in red form the P_4 and blue edges connect it to the tip of the counterexample jewel. Relevant non-edges are depicted as dashed lines – showing all of them would clutter the illustration. For clarity, some selected N-sets are indicated as gray ellipses.

▶ **Lemma 6** (Proof omitted). *If G is a connected $(P_5, bull, gem, C_5)$-free graph of at least two vertices then G contains a dominating edge.*

The proof of the lemma is trivial for P_4-free graphs and, otherwise, works in two steps. Firstly, a dominating P_4 is derived using that G is $(P_5, \text{bull}, \text{gem})$-free. The idea for this is to select an arbitrary P_4 and, in case it is not dominating, establish certain rigid conditions of its neighborhood that reveal another P_4 that is dominating. Secondly, the proof shows that in the absence of induced C_5, the middle edge of a dominating P_4 alone is dominating.

The situation described by the lemma is, quite literally, an *edge case* and therefore calls for a separate treatment, which is carried out in the following section.

4 The Edge Case

Trivially, free flooding in the *edge*-case has a lower bound of $|C| - 1$ moves. Recall that C is the image of the flooding f_0, or, more intuitively, the initial set of colors in G. Moreover, an upper bound of $|C|$ moves follows immediately, since one can always conquer a dominating edge e in a single move and then, using a pivot in e, flood all colors of C but one to make the graph monochromatic. As these are the only two possible cases, it suffices to decide whether a given graph G under an initial flooding f_0 admits a free strategy of $|C| - 1$ moves. If this is not the case, we can compute a dominating edge e in polynomial time and fall back on the $|C|$-move strategy that exploits e.

Accordingly, this section focuses on a polynomial-time approach to determine a free strategy S with $|C| - 1$ moves, if one exists. This approach, surprisingly, requires no restrictions on the given graph. To achieve this minimum number of moves, the strategy S must *eliminate* exactly one color c in each move. More precisely, this means that while

Algorithm 1 Polynomial-time algorithm that returns a free strategy of $|C| - 1$ moves, if one exists, and otherwise $\perp$.

```
1: procedure MINIMAL_STRATEGY(G, f_0)
2:     S ← empty move order
3:     C' ← C                                   ▷ Colors that have not been eliminated
4:     Γ ← {c ∈ C | c is connected}             ▷ Colors that have successfully been connected
5:     while Γ ≠ C do
6:         Δ ← {c ∈ C | Γ connects c} − Γ        ▷ Colors that can be connected optimally
7:         if Δ = ∅ then
8:             return ⊥
9:         end if
10:        for c ∈ Δ do
11:            P ← a color region of c
12:            S ← S followed by a move (v_R, c) for each R ∈ P with arbitrary v_R ∈ R
13:            Γ ← Γ + c                        ▷ c is now connected
14:            C' ← C' − {c | P contains a c-colored region}   ▷ Remove eliminated colors
15:        end for
16:    end while
17:    c ← an arbitrary value from C'           ▷ This will be the last remaining color
18:    return S + a move (v_R, c) for each remaining non-c-colored color component
19: end procedure
```

the color c appears on some vertices before a move i, no vertex is colored with c after that move. Clearly, eliminating a color c requires that the c-colored vertices form a single color component $f_{i-1}^{-1}(c)$ before move i, and that move i uses a pivot vertex $v_i \in f_{i-1}^{-1}(c)$.

Consequently, every move of the strategy S floods a color c from a pivot in the color component $f_{i-1}^{-1}(c')$ of a connected color $c' \neq c$ and, in doing so, may create the now connected color c. Our key observation is that, as long as this principle is respected, the precise ordering of moves is of secondary importance. More specifically, we argue that a free strategy of this type can be obtained by repeatedly selecting a suitable disconnected color c, determining which connected colors c' must be eliminated to make c connected, and then performing the corresponding moves in arbitrary order. Furthermore, this selection process can be done greedily. Any disconnected color whose vertices are only separated by connected colors can be made connected through these moves, immediately.

To formalize this idea in Algorithm 1, we introduce the following notions used there. A set $\Gamma \subseteq C$ of colors is said to *connect* a color $c \in C$, if all pairs $v, w \in f_0^{-1}(c)$ are connected in the graph $G\big[f_0^{-1}(c) + \bigcup_{\gamma\in\Gamma} f_0^{-1}(\gamma)\big]$. Note that this induced subgraph does not need to be connected. If it is, however, and $c \notin \Gamma$, we call the set $\{f_0^{-1}(\gamma) \mid \gamma \in \Gamma\}$ a *color region of* c. As an example, in the initial flooding in Figure 3 the set blue, green connects the color red. Since $G[\{v_0, v_1, v_2, v_3, v_6, v_7\}]$ is connected the set $\{\{v_1, v_2\}, \{v_6\}\}$ is also a color region of red.

A move that connects a previously disconnected color c is called the *connecting move* for c. Accordingly, a move order that contains a connecting move for c is also said to *connect* c. To expand on the example, the second move in Figure 3 is a connecting move for red, and accordingly the depicted move order thus, clearly, connects red. We state the main result of this section in the following theorem.

▶ **Theorem 7.** *For every graph G with initial flooding f_0, Algorithm 1 returns in polynomial time a free strategy of $|C| - 1$ moves for G, if one exists, and $\perp$, otherwise.*

Proof. We first show that Algorithm 1 always terminates. If all colors are connected in f_0, this is obviously the case. The algorithm also clearly terminates if $\Delta = \emptyset$ in any iteration of the while-loop. For all other cases, notice that Γ is always extended by all elements in $\Delta \subseteq C$ and $\Delta \cap \Gamma = \emptyset$. In other words, each iteration of the while-loop (that does not return $\perp$) assigns to Γ a larger subset of C. Since C is finite, we must have $\Gamma = C$ after a finite number of iterations, exiting the while-loop.

Now, assume there is a free strategy S of $|C| - 1$ moves for the input G and f_0 and, for the sake of contradiction, assume that Algorithm 1 returns $\perp$, anyway. Then f_0 contains at least one color that is not connected, since otherwise the algorithm would skip the while-loop and return a strategy. Furthermore, there has to be some iteration of the while-loop where the set $\{c \in C \mid \Gamma \text{ connects } c\}$ is empty.

Let I be the free move order computed by our algorithm up to this point, and let c be the first color that S connects, but I does not. Note that c exists, because S finally connects all colors. In fact, if S would not connect some color c', it could neither eliminate c' nor could it cause the entire graph to become monochromatically c'-colored.

Now, the only way for S to connect the two (or more) disconnected c-colored regions while also eliminating one color in every move is by flooding the areas of one or more connected colors with the color c. To make this clear, observe that flooding any disconnected color with c would not eliminate that color (or any other), and making a move that floods with any other color than c cannot produce monochromatic c-colored paths between previously disconnected c-colored vertices in G. Thus, in order to connect c, the strategy S has to make (at least one) moves (v, c), where v is a vertex from the single color component of some connected color. Let R be the set of these color components. We define $W := \bigcup_{A \in R} \{f_0(v) \mid v \in A\}$. Clearly, $c \in \{c' \in C \mid W \text{ connects } c'\}$ and S connects all colors in W before c.

Now, as c is the first color that is not connected by I it follows that I connects all colors in W. However, a color is added to Γ whenever it becomes connected (see line 13). But this implies that $c \in \Delta$, since $W \subseteq \Gamma$ connects c, a contradiction.

Hence, Algorithm 1 finally connects every color c. Lines 17 and 18 guarantee that all remaining color components are flooded with the same color, producing a monochromatic flooding. Therefore, the algorithm successfully returns a free strategy S' for G and f_0.

It remains to show that S' has length $|C| - 1$. Observe that any move added to S' specifically floods only the single color component of some connected color. As long as every move prevents flooding a c-colored component with the same color c by accident, we are done. But this is the case, as in any iteration of the loop in line 10 that processes the color c, the color region P cannot contain a color component of c, by definition. Moreover, line 18 specifically leaves out any c-colored component when c has been chosen in line 17 as the final color for the graph. This implies that every move eliminates a color and, hence, we have exactly $|C| - 1$ moves in S'.

We observe that Algorithm 1 has polynomial runtime. Using any standard disjoint-set data structure, we can keep track of each monochrome region by first representing each vertex through a singleton set and, unionizing all sets of equal color, at the beginning. From this, we can extract the set of connected colors in polynomial time. By our arguments for termination of the algorithm, we can see that the while-loop has at most $|C|$ iterations, each of which is in polynomial time by using union-find and BFS. ◀

While the above result is only concerned with the free flood filling game, it is easy to see that a very similar but far simpler approach works for FIt, too. There, we would only have a single connected region from which we would be allowed to flood the graph. So, we could

only ever use one type of connecting move for a color c: flood the fixed pivot with c. If, at any point, while greedily applying these moves, we do not have a single color available that can be connected by a move like that, the graph cannot be fixed-flooded in $|C| - 1$ moves.

In the context of free flood filling $(P_5, \text{bull}, \text{gem})$-free graphs, however, the above result settles the *edge*-case, as formalized in the following corollary.

▶ **Corollary 8.** *For every graph G that has a dominating edge and for any initial flooding f_0, a shortest free strategy for G can be computed in polynomial time.*

Notice that our result is effectively stronger than required for our case study. In fact, any graph class that has an upper bound on the length of free strategies of at most $|C|$ steps is in polynomial time by the same arguments.

5 Monochromatic Bead Bracelets in Polynomial Time

The final step of our approach is to compute optimal free strategies for 5-bead bracelets in polynomial time. Indeed, this component of our approach originates from the first author's master's thesis [3]. From a conservative perspective, this problem is already resolved, as an elementary observation shows that 5-bead bracelets are co-gem-free. However, bead bracelets exhibit significantly richer structural properties than the mere presence of dominating P_4. This makes them a worthwhile object of study for more efficient algorithms – the method from [15] works in polynomial time but with a prohibitively large exponent. Additionally, we hope that our analysis of this case leads to further insights toward potential generalizations.

As outlined in Section 3, the method nevertheless builds on the C_5-hub structure to quickly capture a small dominating set and, from there, flood the entire graph color by color. Interestingly, this can be achieved in an even simpler way than in the *edge* case. In the remainder of this section, let G be a 5-bead bracelet constructed from beads $H_0, H_1, \ldots, H_4$. For readability, all indices used to refer to beads and related graph elements are taken modulo five without further mention.

The key observation is that disconnected colors cannot be located as arbitrarily as in the previous section. If a color c is, with respect to f_0, disconnected in G, then there exist vertices v and w in distinct color components of $f_0^{-1}(c)$. Since vertices from neighboring beads are adjacent, it follows that v and w must lie either in the same bead or in non-neighboring ones. The fundamental C_5 of G leaves little room and, consequently, vertices v and w from non-adjacent beads always have distance exactly two.

A second central insight is that, for every $k \in \{0, 1, \ldots, 4\}$, any vertex triple $u \in V(H_k)$, $v \in V(H_{k+1})$, and $w \in V(H_{k+2})$ taken from three consecutive beads forms a dominating set in G. By the full join of neighboring beads, we have $H_{k-1} + H_{k+1} \subseteq N(u)$, $H_k + H_{k+2} \subseteq N(v)$, and $H_k + H_{k+3} \subseteq N(w)$, and hence $N[u] + N[v] + N[w] = V(G)$. As in the previous section, this yields $|C| + 1$ as an upper bound on the length of a shortest strategy for G. Indeed, at most two moves suffice to conquer the triple u, v, w, and from this dominating set, $|C| - 1$ further moves are sufficient to flood every color except one.

As above, the main objective of an algorithm is to distinguish between the possible length cases of a shortest strategy, namely $|C| - 1$, $|C|$, or $|C| + 1$ moves. In fact, we show that the third case never occurs. Before doing so, however, we need to decide whether a given 5-bead bracelet G with initial flooding f_0 can be made monochromatic in the optimal $|C| - 1$ moves. While this could be determined using Algorithm 1 from Section 4, the situation here is considerably simpler, again. We therefore give a direct characterization.

▶ **Lemma 9.** *If G is a bracelet of 5 beads $H_0, H_1, \ldots, H_4$, then, on initial flooding f_0, there exists a free strategy of $|C|-1$ moves for G if and only if there are distinct $i, j \in \{0, 1, \ldots, 4\}$ such that, with respect to f_0, the bead H_i contains a connected color and*

- *H_j contains a (not necessarily different) connected color, or*
- *H_j contains a disconnected color c such that $f_0^{-1}(c) + V(H_i)$ is connected in G.*

Proof. Assume first that G has a free strategy $S = (x_1, c_1), \ldots, (x_s, c_s)$ of $s = |C| - 1$ moves. Like in the previous section, every move has to eliminate one color so that, at the end, G becomes monochromatic. So, with respect to f_0, at least c_1 has to be a connected color. We let i be the index of the bead H_i that contains x_1.

If there is any connected color c (possibly with $c = c_1$) where $f_0^{-1}(c) \cap V(H_j)$ is not empty for any $j \neq i$ then the statement of the lemma holds, already. So, we are left with the case where all connected colors are confined within H_i and all vertices of other beads are part of disconnected colors.

Assume that all colors c leave $f_0^{-1}(c) + V(H_i)$ disconnected. Then consider any vertex $v \in V(H_a)$ with $a = i + 1$ and let $c = f_0(v)$ be the corresponding disconnected color. By our assumption, $f_0^{-1}(c) + V(H_i)$ is not connected and, since neighboring beads are fully joined, it must be true for all $k \neq i$ that $f_0^{-1}(c) \cap V(H_k)$ is non-empty if and only if $k = a$ or $k = a + 2$. The same arguments provide, for any vertex $w \in V(H_b)$ with $b = i - 1$ and with initial color $c' = f_0(w)$, that $f_0^{-1}(c') \cap V(H_k)$ is non-empty for all $k \neq i$ if and only if $k = b$ or $k = b - 2$. Notice that one of the two conditions is true for all disconnected colors that do not appear in H_i only.

Recall, that every move m must eliminate a color and that this is only possible with a pivot that is part of the single color component $f_{m-1}^{-1}(z)$ of a connected color z. Moreover, any disconnected color c can only become connected via a move (x_m, c_m) with $c_m = c$ and pivot x_m making $f_{m-1}^{-1}(x_m) + f_{m-1}^{-1}(c)$ connected. As f_0 defines connected colors only within H_i, there is a prefix $(x_1, c_1), \ldots, (x_p, c_p)$ that uses pivots $x_1, \ldots, x_p$ from H_i and where move p is the first that conquers a vertex outside $V(H_i)$. The move p floods with a color $c_p = f_{p-1}^{-1}(v)$ of a vertex $v \notin V(H_i)$ that is adjacent to $f_{p-1}^{-1}(x_p)$, thus, with $v \in V(H_{i-1}) + V(H_{i+1})$. Without loss of generality, assume that v belongs to H_{i+1}. Using the above arguments, we find at least one vertex $w \in f_{p-1}^{-1}(c_p) \cap V(H_{i+3})$. Due to the C_5-form of G and by the definition of p, the vertex w is not adjacent to $f_{p-1}^{-1}(x_p)$ and, thus, $f_p^{-1}(c_p)$ remains disconnected. But the move p has already used the color c_p and, hence, there is no way for the remaining moves of S to connect this color and still flood with every color but one.

Now, for the converse direction, assume that H_i and H_j fulfill the condition of the lemma. We construct a free strategy for G of $|C| - 1$ moves that, in the first one or two moves, conquers a dominating three-vertex-path $D = u, v, w$ while still eliminating one color per move, and that uses D in the remaining moves to conquer the rest of the colors except one.

Let c be a connected color such that $f_0^{-1}(c) \cap V(H_i)$ is not empty. Furthermore, let the connected or disconnected color in H_j be c' (possibly with $c = c'$).

In the first case, H_i and H_j are adjacent, say with $i + 1 = j$. We select arbitrary vertices $u \in V(H_i) \cap f_0^{-1}(c)$, $v \in V(H_{i+1}) \cap f_0^{-1}(c')$, and $w \in V(H_{i+2})$. If $c = c'$ (where c' is clearly connected) $D = u, v, w$ is conquered with the single move $(u, f_0(w))$, which eliminates c. Otherwise, the two moves $(u, c'), (u, f_0(w))$ conquer D and eliminate c and c'. This happens even if c' is a disconnected color, since $f_0^{-1}(c') + V(H_i)$ is connected, which, in 5-bead-bracelets, implies the same for $f_0^{-1}(c') + f_0^{-1}(u)$.

In the second case, H_i and H_j are non-neighboring, without loss of generality, say with $i + 2 = j$. We select $u \in V(H_i) \cap f_0^{-1}(c)$ and $w \in V(H_{i+2}) \cap f_0^{-1}(c')$. This time, $c = c'$ means that there is a $v \in V(H_{i+1}) \cap f_0^{-1}(c)$ that completes a monochromatic D without any moves.

Otherwise, and if c' is connected, we select v arbitrarily in $V(H_{i+1})$ and make the two moves $(u, f_0(v)), (w, f_0(v))$ to conquer D, which eliminate c and c'. Finally, having c' disconnected but with the condition in the lemma, requires the existence of a vertex $v \in V(H_{i+1}) \cap f_0^{-1}(c)$, because G is a 5-bead bracelet and $f_0^{-1}(c') + V(H_i)$ is connected. Then the move (u, c') establishes D and eliminates c. This completes the proof. ◀

If G and f_0 do not admit a free strategy of $|C| - 1$ moves, then we show that there always exists a strategy of length $|C|$. The underlying idea is the following. Either a dominating three-vertex path can be conquered in a single move, after which $|C| - 1$ additional moves suffice to flood all colors, or conquering such a dominating path requires two moves but, in doing so, one color can be eliminated simultaneously. In the latter case, only $|C| - 2$ further moves are needed to flood the remainder of the graph.

More precisely, our polynomial-time approach for free-flooding 5-bead bracelets works as follows. It begins by checking the conditions for the existence of a strategy with $|C| - 1$ moves constructing such a strategy if they are satisfied. If this attempt fails, the algorithm checks if any two non-neighboring beads – say H_0 and H_2 – contain equally colored vertices $u \in V(H_0)$ and $w \in V(H_2)$. Then a dominating path $D = u, v, w$ can be created in a single move by flooding the color $f_0(u)$ from any pivot vertex $v \in V(H_1)$.

Otherwise, all non-neighboring beads are color-disjoint, and we require two moves to produce $D = u, v, w$: first, flooding the color $f_0(u)$ for some $u \in V(H_0)$ from an arbitrary pivot vertex $v \in V(H_1)$ and, second, flooding the color $f_0(w)$ for some $w \in V(H_2)$ from the same pivot v. With respect to f_1, the flooded area $f_1^{-1}(c)$ forms a single color component containing v, so that the second move recolors all vertices of this component. Thus, these two moves already eliminate the color $c = f_0(u)$.

Using the dominating path D as a hub, the remaining colors can then be flooded in the standard way. Consequently, the algorithm correctly returns a free strategy of $|C|$ moves.

▶ **Lemma 10.** *For every 5-bead bracelet G with initial flooding f_0, a shortest free strategy for G can be computed in polynomial time.*

Proof. To solve the problem, an algorithm can firstly determine, if a $|C| - 1$-move strategy exists and, if so, compute one. More precisely, it can check the conditions of Lemma 9 and, in case of a positive match, the proof of Lemma 9 describes the computation of the $|C| - 1$-move strategy via a dominating set D. It is easy to see that this works in polynomial time.

In case of failure, the algorithm can produce a dominating monochromatic set D with at most one extra move and, based on that, produce a $|C|$-move strategy. For that purpose, it checks in polynomial time, if there exist equally colored vertices $u \in V(H_k)$ and $w \in V(H_{k+2})$ in non-neighboring beads H_k and H_{k+2} for some $k \in \{0, 1, \ldots, 4\}$. If u and w are found, a dominating set $D = u, v, w$ can be made monochromatic in one move $(v, f_0(u))$ with arbitrary pivot $v \in V(H_{k+1})$. The rest of the strategy is to flood any sequence of $|C| - 1$ colors, which is produced in linear time.

Otherwise, the algorithm selects, in $\mathcal{O}(1)$-time, an arbitrary pivot $v \in V(H_1)$ and the colors $c = f_0(u)$ of any vertex $u \in V(H_0)$ and $c' = f_0(w)$ of any vertex $w \in V(H_2)$. The moves (v, c) and (v, c') make $D = u, v, w$ a monochromatic dominating set and also eliminate color c. This is because c can only appear in H_0 and at most one neighboring bead H_4 or H_1. In any case $f_0^{-1}(c) + v$ is connected and, hence, the first move makes $f_1^{-1}(c)$ a color component that contains v, the pivot of the second move. The second move recolors all vertices in $f_1^{-1}(c)$ and thereby eliminates c. Completing the strategy with the remaining $|C| - 2$ colors takes linear time. ◀

Together with Corollary 8, the above lemma implies our main result stated in Theorem 2. We would like to point out that the results of this section are, in fact, stronger than what is expressed by the concluding lemma. Indeed, it is not necessary for the graph G to be a bracelet whose beads $H_0, H_1, \ldots, H_4$ are cographs. Since the cograph structure of the beads is never used, our approach applies to any choice of non-empty graphs $H_0, H_1, \ldots, H_4$, provided they are arranged as a bracelet over a C_5.

6 Flood Filling Games are NP-complete on Thin-Spider Playfields

This section proves Theorem 3. Our approach adapts ideas from the methods of Fleischer and Woeginger [8], Fukui et al. [10], and Fellows et al. [6]. However, the reduction gadgets used in our proof are so-called *thin spiders*. Thin spiders, as defined by Nastos and Gao [14], are connected split graphs admitting a vertex partition into a clique K and an independent set I such that every vertex in K and in I has at most one neighbor in the respective other part.

It is easy to see that these graphs are exactly the claw- and diamond-free split graphs. Indeed, any induced claw would have its central vertex in K, forcing it to be adjacent to at least two vertices in I. Similarly, any induced diamond would place at least one of the two non-adjacent vertices into I, giving it at least two neighbors in K. The converse direction is obvious, if $|K| \leq 2$. Otherwise, let G be a split graph in which some vertex v has at least two neighbors x, y in the opposite part. Then we find an induced claw or diamond in G as follows. We may assume that the partition into K and I is chosen such that every vertex of I is non-adjacent to at least one vertex of K, as otherwise it could simply be moved from I into K. Firstly, if $v \in I$ and $x, y \in K$, then there exists a vertex $z \in K - N[v]$, and the vertices v, x, y, z induce a diamond. And, secondly, if $v \in K$ and $x, y \in I$, let $z \in K - N[x]$. Since v is adjacent to z, while y and z are not adjacent (as this would imply $|N[y] \cap K| > 1$), the vertices v, x, y, z induce a claw with central vertex v.

Notice that thin spiders exhibit very little structure: up to isomorphism, there is exactly one thin spider for each value of $|K|$ and $|I|$. In this sense, our result is slightly stronger than the earlier NP-hardness results for split graphs from [8, 10]. On the other hand, these works establish hardness even for instances with a proper initial flooding (where no pair of adjacent vertices has the same color), which strengthens their conclusions. In our setting, this restriction has to be dropped, as the simplicity of the thin-spider gadgets does not provide sufficient structure to compensate for it.

With these deeper insights into thin spiders, we are now ready to state the main result of this section.

▶ **Lemma 11** (Proof omitted). Flood-It *and* Free Flood-It *are* NP*-complete on thin-spider graphs.*

The proof of the lemma essentially works by a standard reduction from Minimum Vertex Cover to both flooding games. The basic idea follows the NP-hardness proof for trees by Fellows et al. [6]. The vertices of a given graph H are used as colors in the initial flooding of a gadget graph G; in our case, a thin spider, and in [6], a tree of diameter four. Both gadget types share the principle that every edge vw of H is represented by two leaves, one corresponding to each endpoint v and w.

Then H has a vertex cover W, if and only if all vertices of G can be conquered from a designated, fixed pivot p using exactly $|V(H)| + |W|$ moves. These moves are divided into three stages: the first runs through the colors of W, the second through $V(H) - W$, and

the third through W again. The first stage provides a head start for every vertex $w \in W$, allowing it to conquer the parent vertex w for all edges vw of H. This rapid expansion is exploited in the second stage: while conquering the parents of all remaining leaves, every leaf representing the v-endpoint of an edge $vw \in E(H)$ with $w \in W$ is flooded as well. At this point, the graph G is monochromatic except for the leaves representing the w-endpoints of edges of H with $w \in W$, which are conquered in the third stage.

An additional difficulty arises for Free Flood-It, where every *free* pivot has to be effectively *forced* to behave like the fixed pivot p. To this end, we adapt an idea of Fukui et al. [10] and attach additional leaves to G, one for each color in $V(H)$, whose parents lie in a designated part of K associated with the intended fixed pivot p. This construction restricts the available flooding moves and thereby allows them to be interpreted as if they were performed from p.

We conclude this section by deriving the announced consequences of Lemma 11. Since thin spiders are exactly the (claw, diamond)-free split graphs, and since split graphs are the $(2K_2, C_4, C_5)$-free graphs [9], we obtain the following corollary.

▶ **Corollary 12.** Flood-It *and* Free Flood-It *are* NP*-complete on graphs containing no induced claw, diamond,* $2K_2$*,* C_4 *and* C_5*.*

We observe that P_5 and co-banner contain an induced $2K_2$, the jewels house and banner contain an induced C_4, chair contains an induced claw, and both kite and gem contain an induced diamond. Hence, the corollary above implies Theorem 3.

7 Conclusion

To date, the complexity of Flood-It and Free Flood-It has been determined for 896 of the 1024 cograph generalizations. As a central contribution, this paper added 192 classes to this state of the art. The 128 unresolved classes consist of (i) (P_5, bull)-free graphs containing a gem or (ii) (bull, chair)-free graphs. Based on a preliminary analysis of structural restrictions imposed by these forbidden subgraphs, and the apparent absence of typical hardness conditions within these classes, we arrive at the following conjecture.

▶ **Conjecture 13.** Flood-It *and* Free Flood-It *are solvable in polynomial time on all* 128 *remaining classes.*

We note that our results are, in fact, slightly stronger than suggested by the classification in terms of forbidden jewels. While $(P_5, \text{bull}, \text{gem})$-free graphs exhibit the structure derived in Section 3, the converse does not hold. Since our algorithms effectively operate on either a dominating edge or a C_5-bracelet on arbitrary graphs, the resulting tractability extends beyond $(P_5, \text{bull}, \text{gem})$-free graphs. This observation suggests that flooding tractability is governed by the existence and / or fine structural properties of dominating subgraphs that serve as flooding hubs. Identifying and exploiting such structures may therefore provide promising directions for extending tractability to larger graph classes and for achieving a deeper understanding of flooding games in general.

References

1 Andreas Brandstädt, Feodor F. Dragan, Hoàng-Oanh Le, and Raffaele Mosca. New graph classes of bounded clique-width. *Theor. Comp. Sys.*, 38(5):623–645, September 2005. doi:10.1007/s00224-004-1154-6.

2 Raphaël Clifford, Markus Jalsenius, Ashley Montanaro, and Benjamin Sach. The complexity of flood filling games. *Theory of Computing Systems*, 50(1):72–92, January 2012. doi:10.1007/s00224-011-9339-2.

3 Martin Darmüntzel. Die Komplexität des Flood-It-Spiels für Verallgemeinerungen von Co-Graphen. Master's thesis, University of Rostock, 2025.

4 Martin Darmüntzel, Christian Rosenke, and Mark Scheibner. Flood-it with jewelry – characterizing the game complexity for cograph generalizations, 2026. arXiv:2606.23837.

5 Uéverton dos Santos Souza, Fábio Protti, and Maise Silva. An algorithmic analysis of flood-it and free-flood-it on graph powers. *Discrete Mathematics & Theoretical Computer Science*, Vol. 16 no. 3, December 2014. doi:10.46298/dmtcs.2086.

6 Michael R. Fellows, Uéverton dos Santos Souza, Fábio Protti, and Maise Dantas da Silva. Tractability and hardness of flood-filling games on trees. *Theoretical Computer Science*, 576:102–116, 2015. doi:10.1016/j.tcs.2015.02.008.

7 Michael R. Fellows, Frances A. Rosamond, Maise Dantas da Silva, and Uéverton S. Souza. *A Survey on the Complexity of Flood-Filling Games*, pages 357–376. Springer International Publishing, Cham, 2018. doi:10.1007/978-3-319-98355-4_20.

8 Rudolf Fleischer and Gerhard J. Woeginger. An algorithmic analysis of the honey-bee game. In *Fun with Algorithms*, pages 178–189, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. doi:10.1007/978-3-642-13122-6_19.

9 Stéphane Foldes and Peter L. Hammer. Split graphs. In *Proceedings of the Eighth Southeastern Conference on Combinatorics, Graph Theory and Computing*, pages 311–315, Winnipeg, MB, 1977. Utilitas Mathematica Publishing, Inc.

10 Hiroyuki Fukui, Yota Otachi, Ryuhei Uehara, Takeaki Uno, and Yushi Uno. On complexity of flooding games on graphs with interval representations. In Jin Akiyama, Mikio Kano, and Toshinori Sakai, editors, *Computational Geometry and Graphs*, pages 73–84, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

11 Wing-Kai Hon, Ton Kloks, Fu-Hong Liu, Hsiang-Hsuan Liu, and Hung-Lung Wang. Flood-it on AT-free graphs. *CoRR*, abs/1511.01806, 2015. arXiv:1511.01806.

12 Kitty Meeks and Alexander Scott. The complexity of flood-filling games on graphs. *Discrete Applied Mathematics*, 160(7):959–969, 2012. doi:10.1016/j.dam.2011.09.001.

13 Kitty Meeks and Alexander Scott. Spanning trees and the complexity of flood-filling games. *Theory of Computing Systems*, 54:731–753, 2014. doi:10.1007/s00224-013-9482-z.

14 James Nastos and Yong Gao. Bounded search tree algorithms for parameterized cograph deletion: Efficient branching rules by exploiting structures of special graph classes. *Discrete Mathematics, Algorithms and Applications*, 4, June 2010. doi:10.1142/S1793830912500085.

15 Christian Rosenke and Mark Scheibner. Fanciful figurines flip free flood-it – polynomial-time miniature painting on co-gem-free graphs, 2026. doi:10.48550/arXiv.2602.00690.