HAL
open science

How Bad is the Freedom to Flood-It?

Rémy Belmonte, Mehdi Khosravian Ghadikolaei, Masashi Kiyomi, Michael Lampis, Yota Otachi

To cite this version:

Rémy Belmonte, Mehdi Khosravian Ghadikolaei, Masashi Kiyomi, Michael Lampis, Yota Otachi. How Bad is the Freedom to Flood-It?. Journal of Graph Algorithms and Applications, 2019, 23 (2), pp.111-134. ⟨10.7155/jgaa.00486⟩. ⟨hal-02310396⟩

HAL Id: hal-02310396

https://hal.science/hal-02310396v1

Submitted on 15 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Autorisation HAL

HAL Authorization

Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 23, no. 2, pp. 111–134 (2019)
DOI: 10.7155/jgaa.00486

How Bad is the Freedom to Flood-It?

Rémy Belmonte[1] *Mehdi Khosravian Ghadikolaei*[2]
Masashi Kiyomi[3] *Michael Lampis*[2] *Yota Otachi*[4]

[1]The University of Electro-Communications, Chofu, Tokyo, Japan
[2]Université Paris-Dauphine, PSL University, CNRS,
LAMSADE, 75016 Paris, France
[3]Yokohama City University, Yokohama, Japan
[4]Kumamoto University, Kumamoto, Japan

Abstract

Fixed-Flood-It and Free-Flood-It are combinatorial problems on graphs that generalize a very popular puzzle called *Flood-It*. Both problems consist of recoloring moves whose goal is to produce a monochromatic ("flooded") graph as quickly as possible. Their difference is that in Free-Flood-It the player has the additional freedom of choosing the vertex to play in each move. In this paper, we investigate how this freedom affects the complexity of the problem. It turns out that the freedom is *bad* in some sense. We show that some cases trivially solvable for Fixed-Flood-It become intractable for Free-Flood-It. We also show that some tractable cases for Fixed-Flood-It are still tractable for Free-Flood-It but need considerably more involved arguments. We finally present some combinatorial properties connecting or separating the two problems. In particular, we show that the length of an optimal solution for Fixed-Flood-It is always at most twice that of Free-Flood-It, and this is tight.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
July 2018	October 2018	December 2018	December 2018	January 2019
		Published: January 2019		
	Article type: Regular Paper		Communicated by: G. Liotta	

This work is partially supported by JSPS and MAEDI under the Japan-France Integrated Action Program (SAKURA) Project GRAPA 38593YJ, and by JSPS KAKENHI Grant Numbers JP18K11157, JP18K11168, JP18K11169, JP18H04091. A preliminary version of this paper appeared in the proceedings of the 9th International Conference on Fun with Algorithms (FUN 2018), Leibniz International Proceedings in Informatics 100 (2018) 5:1–5:13, doi:10.4230/LIPIcs.FUN.2018.5.

E-mail addresses: remy.belmonte@uec.ac.jp (Rémy Belmonte) mehdi.khosravian-ghadikolaei@dauphine.fr (Mehdi Khosravian Ghadikolaei) masashi@yokohama-cu.ac.jp (Masashi Kiyomi) michail.lampis@dauphine.fr (Michael Lampis) otachi@cs.kumamoto-u.ac.jp (Yota Otachi)

1 Introduction

Flood-It is a popular puzzle, originally released as a computer game in 2006 by LabPixies (see [2]). In this game, the player is presented with (what can be thought of as) a vertex-colored grid graph, with a designated special *pivot* vertex, usually the top-left corner of the grid. In each move, the player has the right to change the color of all vertices contained in the same monochromatic (connected) component as the pivot to a different color of her choosing. Doing this judiciously gradually increases the size of the pivot's monochromatic component, until the whole graph is *flooded* with one color. The goal is to achieve this flooding with the minimum number of moves. See Figure 1 for an example.

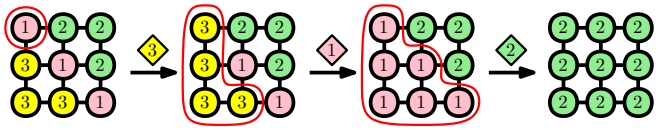


Figure 1: A flooding sequence on a 3×3 grid. Each move in this example changes the color of the top-left monochromatic (connected) component. Under such a restriction, the depicted sequence is shortest.

Following the description above, *Flood-It* immediately gives rise to a natural optimization problem: given a vertex-colored graph, determine the shortest sequence of flooding moves that wins the game. This problem has been extensively studied in the last few years (e.g. [14, 17, 19, 18, 11, 6, 22, 7, 20, 13, 8]; a more detailed summary of known results is given below), both because of the game's popularity (and addictiveness!), but also because the computational complexity questions associated with this problem have turned out to be surprisingly deep, and the problem has turned out to be surprisingly intractable.

The goal of this paper is to add some insights to our understanding of this interesting, puzzle-inspired, optimization problem, by taking a closer look at the importance of the *pivot vertex*. As explained above, the classical version of the game only allows the player to change the color of a special vertex and its component and has been studied under the name Fixed-Flood-It [17, 19, 18] (or Flood-It in some papers [2, 22, 6, 7, 13]). However, it is extremely natural to also consider a version where the player is also allowed to play a different vertex of her choosing in each turn. This has also been well-studied under the name Free-Flood-It [2, 14, 17, 19, 18, 6, 22]. See, e.g., Figure 2.

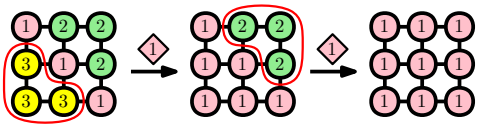


Figure 2: A flooding sequence with no restriction on selected monochromatic components, which is shorter than the one in Figure 1.

Since both versions of this problem have been studied before, the question of the impact of the pivot vertex on the problem's structure has (at least implicitly) been considered. Intuitively, one would expect Free-Flood-It to be a harder problem; after all, the player has to choose a color to play *and* a vertex to play it on, and is hence presented with a larger set of possible moves. The state of the art seems to confirm this intuition, as only some of the positive algorithmic results known for Fixed-Flood-It are known also for Free-Flood-It, while there do exist some isolated cases where Fixed-Flood-It is tractable and Free-Flood-It is hard, for example co-comparability graphs [9, 11] and grids of height 2 [2, 18]. Nevertheless, these results do not completely pinpoint the added complexity brought by the task of selecting a vertex to play, as the mentioned algorithms for Fixed-Flood-It are already non-trivial, and hence the jump in complexity is likely to be the result of *the combination* of the tasks of picking a color and a vertex. More broadly, [6] presented a generic reduction from Fixed-Flood-It to Free-Flood-It that preserves a number of crucial parameters (number of colors, optimal value, etc.) and gives convincing evidence that Free-Flood-It is always at least as hard as Fixed-Flood-It, but not necessarily harder.

Our Results We investigate the complexity of Free-Flood-It, mostly from the point of view of parameterized complexity,[1] as well as the impact on the combinatorics of the game of allowing moves outside the pivot.

Our first result is to show that Free-Flood-It is W[2]-hard parameterized by the number of moves in an optimal solution. We recall that for Fixed-Flood-It this parameterization is trivially fixed-parameter tractable: when a player has only k moves available, then we can safely assume that the graph uses at most (roughly) k colors, hence one can easily consider all possible solutions in FPT time. The interest of our result is, therefore, to demonstrate that the task of deciding which vertex to play next is sufficient to make Free-Flood-It significantly harder than Fixed-Flood-It. Indeed, the W[2]-hardness reduction we give, implies also that Free-Flood-It is not solvable in $n^{o(k)}$ time under the ETH (Exponential Time Hypothesis). This tightly matches the complexity of a trivial algorithm which considers all possible vertices and colors to be played. This is the first concrete example showing a case where Fixed-Flood-It is essentially trivial, but Free-Flood-It is intractable.

Motivated by this negative result we consider several other parameterizations of the problem. We show that Free-Flood-It is fixed-parameter tractable when parameterized by the number of possible moves and the clique-width. This result is tight in the sense that the problem is hard when parameterized by only one of these parameters. It also implies the fixed-parameter tractability of the problem parameterized by the number of colors and the modular-width. In a similar vein, we present a polynomial kernel when Free-Flood-It is parameterized by the input graph's neighborhood diversity and number of colors.

[1] For readers unfamiliar with the basic notions of this field, we refer to standard textbooks [4, 10].

An analogous result was shown for Fixed-Flood-It in [7], but because of the freedom to select vertices, several of the tricks used there do not apply to Free-Flood-It, and our proofs are slightly more involved. Our previously mentioned reduction also implies that Free-Flood-It does not admit a polynomial kernel parameterized by vertex cover, under standard assumptions. This result was also shown for Fixed-Flood-It in [7], but it does not follow immediately for Free-Flood-It, as the reduction of [6] does not preserve the graph's vertex cover.

Motivated by the above results, which indicate that the complexity of the problem can be seriously affected if one allows non-pivot moves, we also study some more purely combinatorial questions with algorithmic applications. The main question we pose here is the following. It is obvious that for all instances the optimal number of moves for Free-Flood-It is upper-bounded by the optimal number of moves for Fixed-Flood-It (since the player has strictly more choices), and it is not hard to construct instances where Fixed-Flood-It needs strictly more moves. Can we bound the optimal number of Fixed-Flood-It moves needed as a function of the optimal number of Free-Flood-It moves? Somewhat surprisingly, this extremely natural question does not seem to have been explicitly considered in the literature before. Here, we completely resolve it by showing that the two optimal values cannot be more than a factor of 2 apart, and constructing a family of simple instances where they are exactly a factor of 2 apart. As an immediate application, this gives a 2-approximation for Free-Flood-It for every case where Fixed-Flood-It is known to be tractable.

We also consider the problem's monotonicity: Fixed-Flood-It has the nice property that even an adversary that selects a single bad move cannot increase the optimal (that is, in the worst case a bad move is a wasted move). We construct minimal examples which show that Free-Flood-It does not have this nice monotonicity property, even for extremely simple graphs, that is, making a bad move may not only waste a move but also make the instance strictly worse. Such a difference was not explicitly stated in the literature, while the monotonicity of Fixed-Flood-It was seem to be known or at least assumed. The only result we are aware of is the monotonicity of Free-Flood-It on paths shown by Meeks and Scott [17].

Known results In 2009, the NP-hardness of Fixed-Flood-It with six colors was sketched by Elad Verbin as a comment to a blog post by Sariel Har-Peled [24]. Independently to the blog comment, Clifford et al. [2] and Fleischer and Woeginger [9] started investigations of the complexity of the problem, and published the conference versions of their papers at FUN 2010. Here we mostly summarize some of the known results on Free-Flood-It. For more complete lists of previous result, see e.g. [11, 14, 7].

Free-Flood-It is NP-hard if the number of colors is at least 3 [2] even for trees with only one vertex of degree more than 2 [14, 6], while it is polynomial-time solvable for general graphs if the number of colors is at most 2 [2, 17, 14]. Moreover, it is NP-hard even for height-3 grids with four colors [17]. Note

that this result implies that Free-Flood-It with a constant number of colors is NP-hard even for graphs of bounded bandwidth. If the number of colors is unbounded, then it is NP-hard for height-2 grids [18], trees of radius 2 [6], proper interval graphs and caterpillars [11]. Also, it is known that there is no constant-factor approximation with a factor independent of the number of colors unless P = NP [2].

There are a few positive results on Free-Flood-It. Meeks and Scott [19] showed that every colored graph has a spanning tree with the same coloring such that the minimum number of moves coincides in the graph and the spanning tree. Using this property, they showed that if a graph has only a polynomial number of vertex subsets that induce connected subgraphs, then Free-Flood-It (and Fixed-Flood-It) on the graph can be solved in polynomial time. This in particular implies the polynomial-time solvability on subdivisions of a fixed graph. It is also known that Free-Flood-It for interval graphs and split graphs is fixed-parameter tractable when parameterized by the number of colors [11].

We refer the interested readers to the recent survey on *Food-It* by Fellows et al. [8] for further details of known results and possible applications in bioinfomatics.

2 Preliminaries

For a positive integer k, we use $[k]$ to denote the set $\{1, \ldots, k\}$. Given a graph $G = (V, E)$, a coloring function $\mathbf{col}\colon V \to [c_{\max}]$, where $c_{\max}$ is a positive integer, and $u \in V$, we denote by $\mathbf{Comp}(\mathbf{col}, u)$ the maximal set of vertices S such that for all $v \in S$, $\mathbf{col}(u) = \mathbf{col}(v)$ and there exists a path from u to v such that for all its internal vertices w we have $\mathbf{col}(w) = \mathbf{col}(u)$. In other words, $\mathbf{Comp}(\mathbf{col}, u)$ is the monochromatic connected component that contains u under the coloring function $\mathbf{col}$.

Given $G, \mathbf{col}$, a *move* is defined as a pair (u, i) where $u \in V$, $i \in [c_{\max}]$. The *result* of the move (u, c) is a new coloring function $\mathbf{col}'$ defined as follows: $\mathbf{col}'(v) = c$ for all $v \in \mathbf{Comp}(\mathbf{col}, u)$; $\mathbf{col}'(v) = \mathbf{col}(v)$ for all other vertices. In words, a move consists of changing the color of u, and of all vertices in the same monochromatic component as u, to c. Given the above definition we can also define the result of a sequence of moves $(u_1, c_1), (u_2, c_2), \ldots, (u_k, c_k)$ on a colored graph with initial coloring function $\mathbf{col}_0$ in the natural way, that is, for each $i \in [k]$, $\mathbf{col}_i$ is the result of move (u_i, c_i) on $\mathbf{col}_{i-1}$.

The Free-Flood-It problem is defined as follows: given a graph $G = (V, E)$, an integer k, and an initial coloring function $\mathbf{col}_0$, decide if there exists a sequence of k moves $(u_1, c_1), (u_2, c_2), \ldots, (u_k, c_k)$ such that the result $\mathbf{col}_k$ obtained by applying this sequence of moves on $\mathbf{col}_0$ is a constant function (that is, $\forall u, v \in V$ we have $\mathbf{col}_k(u) = \mathbf{col}_k(v)$).

In the Fixed-Flood-It problem we are given the same input as in the Free-Flood-It problem, as well as a designated vertex $p \in V$ (the pivot). The question is again if there exists a sequence of moves such that $\mathbf{col}_k$ is monochromatic, with the added constraint that we must have $u_i = p$ for all

$i \in [k]$.

We denote by $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col})$ the minimum k such that for the input $(G, \mathbf{col})$ the FREE-FLOOD-IT problem admits a solution of k moves. We define $\mathrm{OPT}_{\mathrm{Fixed}}(G, \mathbf{col}, p)$ in the same way for the FIXED-FLOOD-IT problem.

2.1 Graph parameters

Let $G = (V, E)$ be a graph. For each vertex $v \in V$, we denote the (*open*) *neighborhood* of v by $N_G(v)$. That is, $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$. The *closed neighborhood* of v is the set $N_G[v] := \{v\} \cup N_G(v)$. For $S \subseteq V$, we denote by $N_G(S)$ its (*open*) *neighborhood* $\bigcup_{v \in S} N_G(v) \setminus S$. We omit the subscript G when the underlying graph is clear from the context.

Vertex cover number: A set $S \subseteq V$ is a *vertex cover* of a graph $G = (V, E)$ if each edge in E has at least one end point in S. The minimum size of a vertex cover of a graph is its *vertex cover number*. By $\mathsf{vc}(G)$, we denote the vertex cover number of G.

Neighborhood diversity: Two vertices $u, v \in V$ are *true twins* in G if $N[u] = N[v]$ and are *false twins* in G if $N(u) = N(v)$. Two vertices are *twins* if they are true twins or false twins. Note that true twins are adjacent and false twins are not. The *neighborhood diversity* of G, denoted $\mathsf{nd}(G)$, is the minimum number k such that V can be partitioned into k sets of twin vertices. It is known that $\mathsf{nd}(G) \leq 2^{\mathsf{vc}(G)} + \mathsf{vc}(G)$ for every graph G [15]. Given a graph, its neighborhood diversity and the corresponding partition into sets of twins can be computed in polynomial time [15]; in fact, using fast modular decomposition algorithms, the neighborhood diversity of a graph can be computed in linear time [16, 23].

Modular-width: Let H be a graph with $k \geq 2$ vertices $v_1, \ldots, v_k$, and let $H_1, \ldots, H_k$ be k graphs. The *substitution* $H(H_1, \ldots, H_k)$ of the vertices of H by $H_1, \ldots, H_k$ is the graph with the vertex set $\bigcup_{1 \leq i \leq k} V(H_i)$ and the edge set $\bigcup_{1 \leq i \leq k} E(H_i) \cup \{\{u, w\} \mid u \in V(H_i), w \in V(H_j), \{v_i, v_j\} \in E(H)\}$. For each i, the set $V(H_i)$ is called a *module* of $H(H_1, \ldots, H_k)$. The *modular-width* of G, denoted $\mathsf{mw}(G)$, is defined recursively as follows:

- If G has only one vertex, then $\mathsf{mw}(G) = 1$.
- If G is the disjoint union of graphs $G_1, \ldots, G_h$, then
$$\mathsf{mw}(G) = \max_{1 \leq i \leq h} \mathsf{mw}(G_i).$$
- If G is a connected graph with two or more vertices, then
$$\mathsf{mw}(G) = \min_{H, H_1, \ldots, H_{|V(H)|}} \max\{|V(H)|, \mathsf{mw}(H_1), \ldots, \mathsf{mw}(H_{|V(H)|})\},$$
where the minimum is taken over all tuples of graphs $(H, H_1, \ldots, H_k)$ such that $G = H(H_1, \ldots, H_{|V(H)|})$.

A recursive substitution structure giving the modular-width can be computed in linear-time [16, 23]. It is known that $\mathsf{mw}(G) \leq \mathsf{nd}(G)$ for every graph G [12].

Clique-width: A *k-expression* is a rooted binary tree such that

- each leaf has label $\circ_i$ for some $i \in \{1, \ldots, k\}$,
- each non-leaf node with two children has label $\cup$, and
- each non-leaf node with only one child has label $\rho_{i,j}$ or $\eta_{i,j}$ ($i, j \in \{1, \ldots, k\}$, $i \neq j$).

Each node in a k-expression represents a vertex-labeled graph as follows:

- a $\circ_i$-node represents a graph with one i-vertex;
- a $\cup$-node represents the disjoint union of the labeled graphs represented by its children;
- a $\rho_{i,j}$-node represents the labeled graph obtained from the one represented by its child by replacing the labels of the i-vertices with j;
- an $\eta_{i,j}$-node represents the labeled graph obtained from the one represented by its child by adding all possible edges between the i-vertices and the j-vertices.

A k-expression represents the graph represented by its root. The *clique-width* of a graph G, denoted by $\mathsf{cw}(G)$, is the minimum integer k such that there is a k-expression representing a graph isomorphic to G. From their definitions, $\mathsf{cw}(G) \leq \mathsf{mw}(G)$ holds for every graph G.

Figure 3 shows relationships among the graph parameters introduced above together with the well-known treewidth and pathwidth (see [4] for definitions of these two parameters).

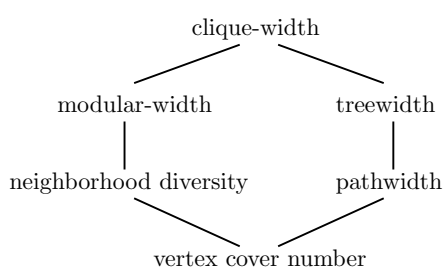


Figure 3: Graph parameters. Each segment implies that the one above is more general than the one below. For example, bounded modular-width implies bounded clique-width but not vice versa.

3 W[2]-hardness of Free-Flood-It

The main result of this section is that Free-Flood-It is W[2]-hard when parameterized by the minimum length of any valid solution (the natural param-

eter). The proof consists of a reduction from Set Cover, a canonical W[2]-complete problem.

Before presenting the construction, we recall two basic observations by Meeks and Vu [20], both of which rest on the fact that any single move can (at most) eliminate a single color from the graph, and this can only happen if a color induces a single component.

Lemma 3.1 ([20]) *For any graph $G = (V, E)$, and coloring function* **col** *that uses $c_{\max}$ distinct colors, we have $OPT_{Free}(G, \mathbf{col}) \geq c_{\max} - 1$.*

Lemma 3.2 ([20]) *For any graph $G = (V, E)$, and coloring function* **col** *that uses $c_{\max}$ distinct colors, such that for all $c \in [c_{\max}]$, $G[\mathbf{col}^{-1}(c)]$ is a disconnected graph, we have $OPT_{Free}(G, \mathbf{col}) \geq c_{\max}$.*

The proof of Theorem 3.6 relies on a reduction from a special form of Set Cover, which we call Multi-Colored Set Cover (MCSC for short). MCSC is defined as follows:

Definition 3.3 In Multi-Colored Set Cover (MCSC) we are given as input a set of elements R and k collections $\mathcal{S}_1, \ldots, \mathcal{S}_k$ of subsets of R. We are asked if there exist k sets $S_1, \ldots, S_k$ such that for all $i \in [k]$, $S_i \in \mathcal{S}_i$, and $\bigcup_{i \in [k]} S_i = R$.

Observe that MCSC is just a version of Set Cover where the collection of sets is given to us pre-partitioned into k parts and we are asked to select one set from each part to form a set cover of the universe. It is not hard to see that any Set Cover instance $(\mathcal{S}, R)$ where we are asked if there exists a set cover of size k can easily be transformed to an equivalent MCSC instance simply by setting $\mathcal{S}_i = \mathcal{S}$ for all $i \in [k]$, since the definition of MCSC does not require that the sub-collections $\mathcal{S}_i$ be disjoint. We conclude that known hardness results for Set Cover immediately transfer to MCSC, and in particular MCSC is W[2]-hard when parameterized by k.

Construction

We are now ready to describe our reduction which, given an MCSC instance with universe R and k collections $\mathcal{S}_i$ ($i \in [k]$) of subsets of R, produces an equivalent instance of Free-Flood-It, that is, a graph $G = (V, E)$ and a coloring function **col** on V. We construct this graph as follows:

- for every set $S \in \mathcal{S}_i$, construct a vertex in V. The set of vertices in V corresponding to sets of $\mathcal{S}_i$ is denoted by I_i and $\mathbf{col}(v) = i$ for each $v \in I_i$. $I_1 \cup ... \cup I_k$ induces an independent set colored $\{1, ..., k\}$.
- for each $i \in [k]$, construct $3k$ new vertices, denoted by L_i and connect all of them to all vertices of I_i such that $L_i \cup I_i$ induces a complete bipartite graph of size $3k \times |I_i|$. Then set $\mathbf{col}(v) = k + 1$ for each $v \in L_i$, for all $i \in [k]$.

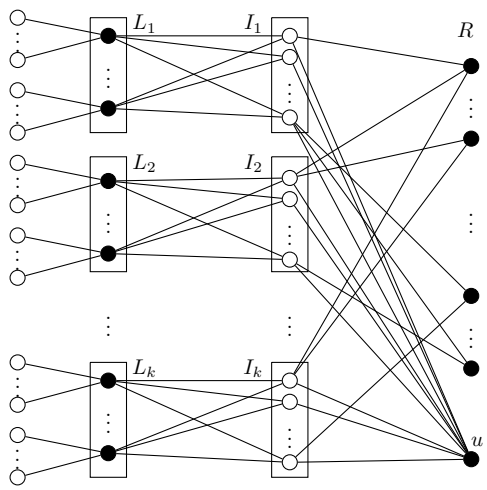


Figure 4: The graph $G = (V, E)$ of Free-Flood-It constructed from the given MCSC instance. All the vertices in each I_i have color i and all black vertices have color $k+1$. Boxes containing black vertices have size $3k$. Also each vertex in L_i has k neighbors with degree 1 colored $1, ..., k$.

- for each vertex $v \in L_i$ for $1 \leq i \leq k$, construct k new leaf vertices connected to v with distinct colors $1, ..., k$.
- for each element $r \in R$, construct a vertex r with $\mathbf{col}(u) = k+1$. For each $S \in \mathcal{S}_i$ such that $r \in S$ we connect r to the vertex of I_i that represents S.
- add a special vertex u with $\mathbf{col}(u) = k+1$ which is connected it to all vertices in I_i for $i \in [k]$.

An illustration of G is shown in Figure 4. In the following we will show that $(G, \mathbf{col})$ as an instance of Free-Flood-It is solvable with at most $2k$ moves if and only if the given MCSC instance has a set cover of size k which contains one set of each $\mathcal{S}_i$.

Lemma 3.4 *If $(\mathcal{S}_1, \ldots, \mathcal{S}_k, R)$ is a YES instance of* MCSC*, then it holds that $OPT_{Free}(G, \mathbf{col}) \leq 2k$.*

Proof: Suppose that there is a solution $S_1, \ldots, S_k$ of the given MCSC instance, with $S_i \in \mathcal{S}_i$, for $i \in [k]$ and $\bigcup_{i \in [k]} S_i = R$. Recall that for each S_i there is a vertex in I_i in the constructed graph representing S_i. Our first k moves consist of changing the color of each of these k vertices to $k+1$ in some arbitrary order.

Observe that in the graph resulting after these k moves the vertices with color $k+1$ form a single connected component: all vertices of R have a neighbor with color $k+1$ because $\bigcup S_i$ is a set cover; all vertices with color $k+1$ in some I_i are in the same component as u; and all vertices of $\bigcup_{i \in [k]} L_i$ are connected to one of the vertices we played. Furthermore, observe that this component dominates

the graph: all remaining vertices of $\bigcup I_i$, as well as all leaves attached to vertices of $\bigcup_{i\in[k]} L_i$ are dominated by the vertices of $\bigcup_{i\in[k]} L_i$. Hence, we can select an arbitrary vertex with color $k+1$, say u, and cycle through the colors $1,\ldots,k$ on this vertex to make the graph monochromatic. □

Now we establish the converse of Lemma 3.4.

Lemma 3.5 *If $OPT_{Free}(G,\mathbf{col}) \leq 2k$, then $(\mathcal{S}_1,\ldots,\mathcal{S}_k,R)$ is a YES instance of* MCSC.

Proof: Suppose that there exists a sequence of at most $2k$ moves solving $(G,\mathbf{col})$. We can assume without loss of generality that the sequence has length exactly $2k$, since performing a move on a monochromatic graph keeps the graph monochromatic. Let $(u_1,c_1),\ldots,(u_{2k},c_{2k})$ be a solution, let $\mathbf{col}_0 = \mathbf{col}$, and let $\mathbf{col}_j$ denote the coloring of G obtained immediately after the jth move has been performed. The key observation that we will rely on is the following:

(i) For all $i \in [k]$, there exist $j \in [k], v \in I_i$ such that $\mathbf{col}_j(v) = k+1$.

In other words, we claim that for each group I_i there exists a vertex that received color $k+1$ at some point during the first k moves. Before proceeding, let us prove this claim. Suppose for contradiction that the claim is false. Then, there exists a group I_i such that no vertex in that group has color $k+1$ in any of the colorings $\mathbf{col}_0,\ldots,\mathbf{col}_k$. We now consider the vertices of L_i and their attached leaves. Since L_i contains $3k > k+2$ vertices, there exist two vertices v_1,v_2 of L_i such that $\{u_1,\ldots,u_k\}$ contains neither v_1,v_2, nor any of their attached leaves. In other words, there exist two vertices of L_i on which the winning sequence does not change colors by playing them or their private neighborhood directly. However, since v_1,v_2 only have neighbors in I_i (except for their attached leaves), and no vertex of I_i received color $k+1$, we conclude that $\mathbf{col}_k(v_1) = \mathbf{col}_k(v_2) = k+1$, that is, the colors of these two vertices have remained unchanged, and the same is true for their attached leaves. Consider now the graph G with coloring $\mathbf{col}_k$: we observe that this coloring uses $k+1$ distinct colors, and that each color induces a disconnected graph. This is true for colors $1,\ldots,k$ because of the leaves attached to v_1,v_2, and true of color $k+1$ because of v_1,v_2 and the fact that no vertex of I_i has color $k+1$. We conclude that $\mathrm{OPT}_{\mathrm{Free}}(G,\mathbf{col}_k) \geq k+1$ by Lemma 3.2, which is a contradiction, because the whole sequence has length $2k$.

Because of claim (i) we can now conclude that for all $i \in [k]$ there exists a $j \in [k]$ such that $\mathbf{col}_{j-1}(u_j) = i$. In other words, for each color i there exists a move among the first k moves of the solution that played a vertex which at that point had color i. To see that this is true consider again for contradiction the case that for some $i \in [k]$ this statement does not hold: this implies that vertices with color i in $\mathbf{col}_0$ still have color i in $\mathbf{col}_1,\ldots,\mathbf{col}_k$, which means that no vertex of I_i has received color $k+1$ in the first k moves, contradicting (i).

As a result of the above, we therefore claim that for all $j \in [k]$, we have $\mathbf{col}_{j-1}(u_j) \neq k+1$. In other words, we claim that none of the first k moves

changes the color of a vertex that at that point had color $k+1$. This is because, as argued, for each of the other k colors, there is a move among the first k moves that changes a vertex of that color. We therefore conclude that for all vertices v for which $\mathbf{col}_0(v) = k+1$ we have $\mathbf{col}_j(v) = k+1$ for all $j \in [k]$. In addition, because in $\mathbf{col}_0$ all colors induce independent sets, each of the first k moves changes the color of a single vertex. Because of claim (i), this means that for each $i \in [k]$ one of the first k moves changes the color of a single vertex from I_i to $k+1$. We select the corresponding set of $\mathcal{S}_i$ in our MCSC solution.

We now observe that, since all vertices of $\bigcup_{i\in[k]} L_i$ retain color $k+1$ throughout the first k moves, $\mathbf{col}_k$ is a coloring function that uses $k+1$ distinct colors, and colors $1, \ldots, k$ induce disconnected graphs (because of the leaves attached to the vertices of each L_i). Thanks to Lemma 3.2, this means that $\mathbf{col}_k^{-1}(k+1)$ must induce a connected graph. Hence, all vertices of R have a neighbor with color $k+1$ in $\mathbf{col}_k$, which must be one of the k vertices played in the first k moves; hence the corresponding element is dominated by our solution and we have a valid set cover selecting one set from each $\mathcal{S}_i$. □

We are now ready to combine Lemmas 3.4 and 3.5 to obtain the main result of this section.

Theorem 3.6 Free-Flood-It *is W[2]-hard parameterized by OPT_{Free}, that is, parameterized by the length of the optimal solution. Furthermore, if there is an algorithm that decides if a* Free-Flood-It *instance has a solution of length k in time $n^{o(k)}$, then the ETH is false.*

Proof: The described construction, as well as Lemmas 3.4 and 3.5 give a reduction from MCSC, which is W[2]-hard parameterized by k, to an instance of Free-Flood-It with $k+1$ colors, where the question is to decide whether $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \leq 2k$. Furthermore, it is known that MCSC generalizes Dominating Set, which does not admit an algorithm running in time $n^{o(k)}$, under the ETH [4]. Since our reduction only modifies k by a constant, we obtain the same result for Free-Flood-It. □

We note that because of Lemma 3.1 we can always assume that the number of colors of a given instance is not much higher than the length of the optimal solution. As a result, Free-Flood-It parameterized by $\mathrm{OPT}_{\mathrm{Free}}$ is equivalent to the parameterization of Free-Flood-It by $\mathrm{OPT}_{\mathrm{Free}} + c_{\max}$ and the result of Theorem 3.6 also applies to this parameterization.

3.1 Kernel lower bound for Free-Flood-It

As a byproduct of the reduction above, we can show a kernel lower bound for Free-Flood-It parameterized by the vertex cover number.

Let P and Q be parameterized problems. A polynomial-time computable function $f\colon \Sigma^* \times N \to \Sigma^* \times N$ is a *polynomial parameter transformation* from P to Q if there is a polynomial p such that for all $(x, k) \in \Sigma^* \times N$,

- $(x, k) \in P$ if and only if $(x', k') = f(x, k) \in Q$, and

- $k' \leq p(k)$.

If such a function exits, then P is *polynomial parameter reducible* to Q.

Proposition 3.7 ([1]) *Let P and Q be parameterized problems, and P' and Q' be unparameterized versions of P and Q, respectively. Suppose P' is NP-hard, Q' is in NP, and P is polynomial parameter reducible to Q. If Q has a polynomial kernel, then P also has a polynomial kernel.*

Theorem 3.8 Free-Flood-It *parameterized by the vertex cover number admits no polynomial kernel unless* $\mathrm{PH} = \Sigma_3^{\mathrm{p}}$.

Proof: The reduction in this section can be seen as a polynomial parameter transformation from "MCSC parameterized by the solution size k and the size $|R|$ of the universe" to "Free-Flood-It parameterized by the vertex cover number" with a polynomial $p(k, |R|) = 3k^2 + |R|$. To see this observe that the black vertices in Figure 4 form a vertex cover of size $3k^2 + |R|$.

Since MCSC is NP-hard and the decision version of Free-Flood-It is in NP, Proposition 3.7 implies that if Free-Flood-It parameterized by the vertex cover number has a polynomial kernel, then MCSC parameterized by k and $|R|$ also has a polynomial kernel.

It is known that Set Cover (and thus MCSC) parameterized simultaneously by k and $|R|$ does not admit a polynomial kernel unless $\mathrm{PH} = \Sigma_3^{\mathrm{p}}$ [5]. This completes the proof. □

4 Clique-width and the number of moves

In this section, we consider as a combined parameter for Free-Flood-It the length of an optimal solution and the clique-width. We show that this case is indeed fixed-parameter tractable by using the theory of the monadic second-order logic on graphs. As an application of this result, we also show that combined parameterization by the number of colors and the modular-width is fixed-parameter tractable.

To prove the main claim, we show that Free-Flood-It with a constant length of optimal solutions is an MSO_1-definable decision problem. The syntax of MSO_1 (one-sorted monadic second-order logic) of graphs includes (i) the logical connectives $\vee$, $\wedge$, $\neg$, $\Leftrightarrow$, $\Rightarrow$, (ii) variables for vertices and vertex sets, (iii) the quantifiers $\forall$ and $\exists$ applicable to these variables, and (iv) the following binary relations:

- $u \in U$ for a vertex variable u and a vertex set variable U;
- $\mathbf{adj}(u, v)$ for two vertex variables u and v, where the interpretation is that u and v are adjacent;
- equality of variables.

If G models an MSO_1 formula φ with an assignment $X_1, \ldots, X_q \subseteq V(G)$ to the q free variables in φ, then we write $\langle G, X_1, \ldots, X_q \rangle \models \varphi$.

It is known that, given a graph of clique-width at most w, an MSO_1 formula φ, and an assignment to the free variables in φ, the problem of deciding whether G models φ with the given assignment is solvable in time $O(f(||\varphi||, w) \cdot n^3)$, where f is a computable function and $||\varphi||$ is the length of φ [3, 21].

Theorem 4.1 *Given an instance $(G, \mathbf{col})$ of* Free-Flood-It *such that G has n vertices and clique-width at most w, it can be decided in time $O(f(k, w) \cdot n^3)$ whether $OPT_{Free}(G, \mathbf{col}) \leq k$, where f is some computable function.*

Proof: Let V_i $(1 \leq i \leq c_{\max})$ be the set of vertices with color i in the input graph. We construct an MSO_1 formula φ with $c_{\max}$ free variables $X_1, \ldots, X_{c_{\max}}$ such that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \leq k$ if and only if G models φ with the assignment $X_i := V_i$ for $1 \leq i \leq c_{\max}$. We can define the desired formula $\varphi(X_1, \ldots, X_{c_{\max}})$ as follows:

$$\varphi(X_1, \ldots, X_{c_{\max}}) := \bigvee_{1 \leq c_1, \ldots, c_k \leq k} \exists_{v_1, v_2, \ldots, v_k \in V(G)} \bigvee_{1 \leq c \leq k} \forall_{u \in V(G)} \ \mathbf{color}_{c,k}(u),$$

where $\mathbf{color}_{c,i}(u)$ for $0 \leq i \leq k$ implies that the color of u is c after the moves $(v_1, c_1), \ldots (v_i, c_i)$.

We define $\mathbf{color}_{c,i}(u)$ recursively as follows. We first set $\mathbf{color}_{c,0}(u) := (u \in X_i)$. This is correct as we assign V_i to X_i. For $1 \leq i \leq k$, we set

$$\mathbf{color}_{c,i}(u) = \begin{cases} \mathbf{color}_{c,i-1}(u) \vee \mathbf{SameCCC}_{i-1}(u, v_i) & c = c_i, \\ \mathbf{color}_{c,i-1}(u) \wedge \neg \mathbf{SameCCC}_{i-1}(u, v_i) & c \neq c_i, \end{cases}$$

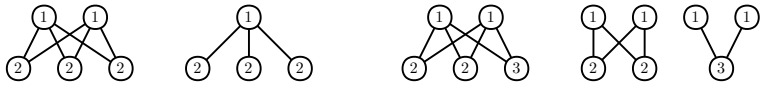
where $\mathbf{SameCCC}_i(u_1, u_2)$ implies that u_1 and u_2 are in the same monochromatic component after the moves $(v_1, c_1), \ldots (v_i, c_i)$. The formula precisely represents the recursive nature of the colors of the vertices. That is, a vertex u is of color c after the ith move (v_i, c_i) if and only if either its color is changed to c by the i move, or it was already of color c before the ith move and its color is not changed by the i move.

Given that $\mathbf{color}_{c,i}(u)$ is defined for all c and u, defining $\mathbf{SameCCC}_i(u_1, u_2)$ is a routine:

$$\mathbf{SameCCC}_i(u_1, u_2) = \bigvee_{1 \leq c \leq k} \exists_{S \subseteq V(G)} \Big((u_1, u_2 \in S) \wedge (\forall_{u \in S} \ \mathbf{color}_{c,i}(u)) \\ \wedge (\forall_{T \subseteq S} \ (T = \emptyset) \vee \exists_{x \in T} \exists_{y \in S \setminus T} \ \mathbf{adj}(x, y)) \Big).$$

Since $k \geq c_{\max} - 1$ by Lemma 3.1, it holds that $||\varphi||$ is bounded by a function of k. □

Corollary 4.2 *Given an integer k and an instance $(G, \mathbf{col})$ of* Free-Flood-It *such that G has n vertices and modular-width at most w, it can be decided in time $O(f(c_{\max}, w) \cdot n^3)$ whether $OPT_{Free}(G, \mathbf{col}) \leq k$, where f is some computable function.*



(a) Removing a false twin is not safe. (b) Removing a twin color is not safe.

Figure 5: Simple reductions that do not work for Free-Flood-It.

Proof: Observe that for every connected graph G of modular-width at most w, it holds that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \leq w + c_{\max} - 2$: we pick one vertex v from a module M; we next color one vertex in each module except M with $\mathbf{col}(v)$; we then play at v with the remaining $c_{\max} - 1$ colors. Thus we can assume that $k \leq w + c_{\max} - 2$. Since the modular-width of a graph is at most its clique-width by their definitions, Theorem 4.1 gives an $O(g(w + c_{\max}, w) \cdot n^3)$-time algorithm for some computable g, which can be seen as an $O(f(c_{\max}, w) \cdot n^3)$-time algorithm for some computable f. □

5 Neighborhood diversity and the number of colors

Since the modular-width of a graph is upper bounded by its neighborhood diversity, Corollary 4.2 in the previous section implies that Free-Flood-It is fixed-parameter tractable when parameterized by both the neighborhood diversity and the number of colors. Here we show that Free-Flood-It admits a polynomial kernel with the same parameterization. This section is devoted to a proof of the following theorem.

Theorem 5.1 Free-Flood-It *admits a kernel of at most* $\mathsf{nd}(G) \cdot c_{\max} \cdot (\mathsf{nd}(G) + c_{\max} - 1)$ *vertices.*

Fellows et al. [7] observed that for Fixed-Flood-It, a polynomial kernel with the same parameterization can be easily obtained since twin vertices of the same color can be safely contracted. In Free-Flood-It, this is true for true twins but not for false twins. See Figure 5a.

Though it might be still possible to show something like "if there are more than some constant number of false twins with the same color, then one can remove one of them without changing the minimum number of moves," here we show a weaker claim. Our reduction rules are as follows:

- Rule TT: Let u and v be true twins of the same color in $(G, \mathbf{col})$. Remove v.
- Rule FT: Let F be a set of false-twin vertices of the same color in $(G, \mathbf{col})$ such that $|F| = \mathsf{nd}(G) + c_{\max}$. Remove one arbitrary vertex in F.

Observe that after applying TT and FT exhaustively in polynomial time, the obtained graph can have at most $\mathsf{nd}(G) \cdot c_{\max} \cdot (\mathsf{nd}(G) + c_{\max} - 1)$ vertices.

This is because each set of twin vertices can contain at most $\mathsf{nd}(G) + c_{\max} - 1$ vertices. Hence, to prove Theorem 5.1, it suffices to show the safeness of the rules. Recall that a reduction rule is *safe* if it produces an equivalent instance.

Lemma 5.2 *Rule TT is safe.*

Proof: Let u and v be true twins of the same color. Observe that removing v is equivalent to contracting the edge $\{u, v\}$. Since u and v are in the same monochromatic component, the lemma holds. □

To guarantees the safeness of FT, we need the following technical lemmas.

Lemma 5.3 *Let $(G, \mathbf{col})$ be an instance of* Free-Flood-It *and $x, y \in V(G)$ be false-twin vertices of the same color c. A sequence $(u_1, c_1), \ldots, (u_k, c_k)$ with $u_i \notin \{x, y\}$ for $1 \leq i \leq k$ is a valid flooding sequence for $(G, \mathbf{col})$ if and only if it is a valid flooding sequence for $(G - x, \mathbf{col}|_{G-x})$.*

Proof: Let $(G', \mathbf{col}') = (G - x, \mathbf{col}|_{G-x})$. If a neighbor of x and y has color c, then the lemma trivially holds. Hence, in what follows, we assume that none of the vertices adjacent to x and y has color c. Assume that $(u_1, c_1), \ldots, (u_k, c_k)$ is valid for at least one of $(G, \mathbf{col})$ and $(G', \mathbf{col}')$. Then there is a move that changes the color of a neighbor of y to c since $u_i \neq y$ for $1 \leq i \leq k$. Let (u_i, c_i) be the first such move. Note that $c_i = c$ and that at least one of $\mathbf{Comp}(\mathbf{col}_{i-1}, u_i)$ and $\mathbf{Comp}(\mathbf{col}'_{i-1}, u_i)$ contains a neighbor of y.

The first part $(u_1, c_1), \ldots, (u_{i-1}, c_{i-1})$ of the sequence has the same effect to $(G, \mathbf{col})$ and $(G', \mathbf{col}')$. That is, the monochromatic components and connections among them are the same in $(G, \mathbf{col}_{i-1})$ and $(G', \mathbf{col}'_{i-1})$ except that $(G, \mathbf{col}_{i-1})$ contains the monochromatic component $\{x\}$. Note that $\{y\}$ is a monochromatic component of color c in both $(G, \mathbf{col}_{i-1})$ and $(G', \mathbf{col}'_{i-1})$, and $\{y\}$ and $\{x\}$ have the same adjacent monochromatic components in $(G, \mathbf{col}_{i-1})$.

Let $\mathcal{C}$ be the set of monochromatic components of color c in $(G, \mathbf{col}_{i-1})$ that are adjacent to $\mathbf{Comp}(\mathbf{col}_{i-1}, u_i)$. Similarly, let $\mathcal{C}'$ be the set of monochromatic components of color c in $(G', \mathbf{col}'_{i-1})$ that are adjacent to $\mathbf{Comp}(\mathbf{col}'_{i-1}, u_i)$. Observe that $\mathbf{Comp}(\mathbf{col}_{i-1}, u_i) = \mathbf{Comp}(\mathbf{col}'_{i-1}, u_i)$, $\mathcal{C}' = \mathcal{C} \setminus \{\{x\}\}$, and $\{y\} \in \mathcal{C} \cap \mathcal{C}'$.

Now we apply the move (u_i, c_i). Recall that $c_i = c$. It follows that

$$\mathbf{Comp}(\mathbf{col}_i, u_i) = \mathbf{Comp}(\mathbf{col}_{i-1}, u_i) \cup \textstyle\bigcup_{C \in \mathcal{C}} C, \text{ and}$$
$$\mathbf{Comp}(\mathbf{col}'_i, u_i) = \mathbf{Comp}(\mathbf{col}'_{i-1}, u_i) \cup \textstyle\bigcup_{C' \in \mathcal{C}'} C'.$$

Since $\mathbf{Comp}(\mathbf{col}_i, u_i) \setminus \mathbf{Comp}(\mathbf{col}'_i, u_i) = \{x\}$ and both $\mathbf{Comp}(\mathbf{col}_i, u_i)$ and $\mathbf{Comp}(\mathbf{col}'_i, u_i)$ include y, the components $\mathbf{Comp}(\mathbf{col}_i, u_i)$ and $\mathbf{Comp}(\mathbf{col}'_i, u_i)$ have the same adjacent monochromatic components. Also, it holds that for each $u \notin \mathbf{Comp}(\mathbf{col}_i, u_i)$, $\mathbf{Comp}(\mathbf{col}_{i-1}, u) = \mathbf{Comp}(\mathbf{col}'_{i-1}, u)$.

This implies that $(G, \mathbf{col}_i)$ and $(G', \mathbf{col}'_i)$ are equivalent and thus the remaining of the sequence, i.e. $(u_{i+1}, c_{i+1}), \ldots, (u_k, c_k)$, has the same effect to $(G', \mathbf{col}'_i)$ and $(G, \mathbf{col}_i)$. Therefore, $(G, \mathbf{col}_k)$ is constant if and only if so is $(G', \mathbf{col}'_k)$. □

Lemma 5.4 *Let S be a set of false-twin vertices with the same color in $(G, \mathbf{col})$. If $|S| \geq OPT_{Free}(G, \mathbf{col}) + 2$, then $OPT_{Free}(G, \mathbf{col}) = OPT_{Free}(G - x, \mathbf{col}|_{G-x})$ for every $x \in S$.*

Proof: We first show that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \leq \mathrm{OPT}_{\mathrm{Free}}(G - x, \mathbf{col}|_{G-x})$ for every $x \in S$. Let $(u_1, c_1), \ldots, (u_k, c_k)$ be an optimal valid flooding sequence for $(G - x, \mathbf{col}|_{G-x})$. Assume that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \geq k$ (otherwise we are done). Since $|S \setminus \{x\}| \geq \mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) + 1 \geq k + 1$, there is a vertex $y \in S \setminus \{x\}$ such that $u_i \neq y$ for $1 \leq i \leq k$. By Lemma 5.3, $(u_1, c_1), \ldots, (u_k, c_k)$ is valid for $(G, \mathbf{col})$ as well.

Next we show the other direction. Since S is a set of monochromatic false-twin vertices, it suffices to show that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \geq \mathrm{OPT}_{\mathrm{Free}}(G - x, \mathbf{col}|_{G-x})$ for some $x \in S$. Let $(u_1, c_1), \ldots, (u_k, c_k)$ be an optimal valid flooding sequence of $(G, \mathbf{col})$. Since $|S| \geq \mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) + 2 = k + 2$, there are two vertices $x, y \in S$ such that $u_i \notin \{x, y\}$ for $1 \leq i \leq k$. By Lemma 5.3, $(u_1, c_1), \ldots, (u_k, c_k)$ is valid for $(G - x, \mathbf{col}|_{G-x})$ as well. □

Corollary 5.5 *Rule FT is safe.*

Proof: Let G be a connected graph and $\mathbf{col}$ a coloring of G with $c_{\max}$ colors. Observe that $(G, \mathbf{col})$ admits a flooding sequence of length $\mathsf{nd}(G) + c_{\max} - 2$ as follows. Let T be a maximal set of twin vertices of G and c be a color used in T. For each maximal set of twin vertices $T' \neq T$ of G, pick a vertex $u \in T'$ and play the move (u, c). After these $\mathsf{nd}(G) - 1$ moves, the vertices of color c form a connected dominating set of G. Now pick a vertex v of color c and play the move (v, c') for each $c' \in [c_{\max}] \setminus \{c\}$. These $c_{\max} - 1$ moves make the coloring constant.

Now for some color class C and a false-twin class I of $(G, \mathbf{col})$, if $|C \cap I| \geq c_{\max} + \mathsf{nd}(G)$, then we can remove an arbitrary vertex in $C \cap I$ while preserving the optimal number of steps by Lemma 5.4. This implies the safeness of FT. □

Note that using the concept of twin colors, Fellows et al. [7] further reduced the number of colors in instances of Fixed-Flood-It and obtained a (nonpolynomial-size) kernel parameterized by the neighborhood diversity. They say that two colors are *twins* if the colors appear in the same family of the maximal sets of twin vertices. They observed that, in Fixed-Flood-It, if there are twin colors, then removing all vertices in one of these colors reduces the fewest number of moves exactly by 1. Unfortunately, this is not the case for Free-Flood-It. See Figure 5b.

6 Relation Between Fixed and Free Flood-It

The main theorem of this section is the following:

Theorem 6.1 *For any graph $G = (V, E)$, coloring function $\mathbf{col}$ on G, and $p \in V$ we have*

$$OPT_{Free}(G, \mathbf{col}) \leq OPT_{Fixed}(G, \mathbf{col}, p) \leq 2OPT_{Free}(G, \mathbf{col}).$$

Theorem 6.1 states that the optimal solutions for Free-Flood-It and Fixed-Flood-It can never be more than a factor of 2 apart. It is worthy of note that we could not hope to obtain a constant smaller than 2 in such a theorem, and hence the theorem is tight.

Theorem 6.2 *There exist instances of* Fixed-Flood-It *such that*

$$OPT_{Fixed}(G, \mathbf{col}, p) = 2OPT_{Free}(G, \mathbf{col}).$$

Proof: Consider a path on $2n + 1$ vertices properly colored with colors $1, 2$. If we set the pivot to be one of the endpoints then $\mathrm{OPT}_{\mathrm{Fixed}} = 2n$. However, it is not hard to obtain a Free-Flood-It solution with n moves by playing every vertex at odd distance from the pivot. □

Before we proceed to give the proof of Theorem 6.1, let us give a high-level description of our proof strategy and some general intuition. The first inequality is of course trivial, so we focus on the second part. We will establish it by induction on the number of non-pivot moves performed by an optimal Free-Flood-It solution. The main inductive argument is based on observing that a valid Free-Flood-It solution will either at some point play a neighbor u of the component of p to give it the same color as p, or if not, it will at some point play p to give it the same color as one of its neighbors. The latter case is intuitively easier to handle, since then we argue that the move that changed p's color can be performed first, and if the first move is a pivot move we can easily fall back on the inductive hypothesis. The former case, which is the more interesting one, can be handled by replacing the single move that gives u the same color as p, with two moves: one that gives p the same color as u, and one that flips p back to its previous color. Intuitively, this basic step is the reason we obtain a factor of 2 in the relationship between the two versions of the game.

The inductive strategy described above faces some complications due to the fact that rearranging moves in this way may unintentionally re-color some vertices, which makes it harder to continue the rest of the solution as before. To avoid this we define a somewhat generalized version of Free-Flood-It, called Subset-Free-Flood-It.

Definition 6.3 *Given $G = (V, E)$, a coloring function* **col** *on G, and a pivot $p \in V$, a* set-move *is a pair (S, c), with $S \subseteq V$ and $S = \mathbf{Comp}(\mathbf{col}, u)$ for some $u \in V$, or $\{p\} \subseteq S \subseteq \mathbf{Comp}(\mathbf{col}, p)$. The result of (S, c) is the coloring $\mathbf{col}'$ that sets $\mathbf{col}'(v) = c$ for $v \in S$; and $\mathbf{col}'(v) = \mathbf{col}(v)$ otherwise.*

We define Subset-Free-Flood-It as the problem of determining the minimum number of set-moves required to make a graph monochromatic, and Subset-Fixed-Flood-It as the same problem when we impose the restriction that every move must change the color of p, and denote as $\mathrm{OPT}_{\text{S-Free}}, \mathrm{OPT}_{\text{S-Fixed}}$ the corresponding optimum values.

Informally, a set-move is the same as a normal move in Free-Flood-It, except that we are also allowed to select an arbitrary connected monochromatic

set S that contains p (even if S is not maximal) and change its color. Intuitively, one would expect moves that set S to be a proper subset of $\mathbf{Comp}(\mathbf{col}, p)$ to be counter-productive, since such moves split a monochromatic component into two pieces. Indeed, we prove below in Lemma 6.4 that the optimal solutions to FIXED-FLOOD-IT and SUBSET-FIXED-FLOOD-IT coincide, and hence such moves do not help. The reason we define this version of the game is that it gives us more freedom to define a solution that avoids unintentionally recoloring vertices as we transform a given FREE-FLOOD-IT solution to a FIXED-FLOOD-IT solution.

Lemma 6.4 *For any graph $G = (V, E)$, coloring function* $\mathbf{col}$ *on G, and pivot $p \in V$ we have $OPT_{Fixed}(G, \mathbf{col}, p) = OPT_{S\text{-}Fixed}(G, \mathbf{col}, p)$.*

Proof: First, observe that $\mathrm{OPT}_{\text{S-Fixed}}(G, \mathbf{col}, p) \leq \mathrm{OPT}_{\text{Fixed}}(G, \mathbf{col}, p)$ is trivial, as any solution of FIXED-FLOOD-IT is a solution to SUBSET-FIXED-FLOOD-IT by playing the same sequence of colors and always selecting all of the connected monochromatic component of p.

Let us also establish the converse inequality. Consider a solution (S_1, c_1), (S_2, c_2), $\ldots$, (S_k, c_k) of SUBSET-FIXED-FLOOD-IT, where by definition we have $p \in S_i$ for all $i \in [k]$. We would like to prove that (p, c_1), (p, c_2), $\ldots$, (p, c_k) is a valid solution for FIXED-FLOOD-IT. Let $\mathbf{col}_i$ be the result of the first i set-moves of the former solution, and $\mathbf{col}'_i$ be the result of the first i moves of the latter solution. We will establish by induction the following:

1. For all $i \in [k]$ we have $\mathbf{Comp}(\mathbf{col}_i, p) \subseteq \mathbf{Comp}(\mathbf{col}'_i, p)$.
2. For all $i \in [k], u \in V \setminus \mathbf{Comp}(\mathbf{col}'_i, p)$ we have $\mathbf{col}_i(u) = \mathbf{col}'_i(u)$.

The statements are true for $i = 0$. Suppose that the two statements are true after $i - 1$ moves. The first solution now performs the set-move (S_i, c_i) with $S_i \subseteq \mathbf{Comp}(\mathbf{col}_{i-1}, p) \subseteq \mathbf{Comp}(\mathbf{col}'_{i-1}, p)$. We now have that $\mathbf{Comp}(\mathbf{col}_i, p)$ contains S_i plus the neighbors of S_i which have color c_i in $\mathbf{col}_{i-1}$. Such vertices either also have color c_i in $\mathbf{col}'_{i-1}$, or are contained in $\mathbf{Comp}(\mathbf{col}'_{i-1}, p)$; in both cases they are included in $\mathbf{Comp}(\mathbf{col}'_i, p)$, which establishes the first condition. To see that the second condition continues to hold observe that every vertex for which $\mathbf{col}_{i-1}(u) \neq \mathbf{col}_i(u)$ or $\mathbf{col}'_{i-1}(u) \neq \mathbf{col}'_i(u)$ belongs in $\mathbf{Comp}(\mathbf{col}'_i, p)$; the colors of other vertices remain unchanged. Since in the end $\mathbf{Comp}(\mathbf{col}_k, p) = V$ the first condition ensures that $\mathbf{Comp}(\mathbf{col}'_k, p) = V$. □

We are now ready to state the proof of Theorem 6.1.

Proof: [Theorem 6.1] As mentioned, we focus on proving the second inequality as the first inequality follows trivially from the definition of the problems. Given a graph $G = (V, E)$, an initial coloring function $\mathbf{col} = \mathbf{col}_0$, and a pivot $p \in V$, we suppose we have a solution to FREE-FLOOD-IT $(u_1, c_1), (u_2, c_2), \ldots, (u_k, c_k)$. In the remainder, we denote by $\mathbf{col}_i$ the coloring that results after the moves $(u_1, c_1), \ldots, (u_i, c_i)$. We can immediately construct an equivalent solution to SUBSET-FREE-FLOOD-IT from this, producing the same sequence of colorings:

$(\mathbf{Comp}(\mathbf{col}_0, u_1), c_1)$, $(\mathbf{Comp}(\mathbf{col}_1, u_2), c_2)$, $\ldots$, $(\mathbf{Comp}(\mathbf{col}_{k-1}, u_k), c_k)$. We will transform this solution to a solution of Subset-Fixed-Flood-It of length at most $2k$, and then invoke Lemma 6.4 to obtain a solution for Fixed-Flood-It of length at most $2k$. More precisely, we will show that for any $G, \mathbf{col}, p$ we have $\mathrm{OPT}_{\text{S-Fixed}}(G, \mathbf{col}, p) \leq 2\mathrm{OPT}_{\text{S-Free}}(G, \mathbf{col}, p)$.

For a solution $\mathcal{S} = (S_1, c_1), (S_2, c_2), \ldots, (S_k, c_k)$ to Subset-Free-Flood-It we define the number of bad moves of $\mathcal{S}$ as $b(\mathcal{S}) = |\{(S_i, c_i) \mid p \notin S_i\}|$. We will somewhat more strongly prove the following statement for all $G, \mathbf{col}, p$: for any valid Subset-Free-Flood-It solution $\mathcal{S}$, we have

$$\mathrm{OPT}_{\text{S-Fixed}}(G, \mathbf{col}, p) \leq |\mathcal{S}| + b(\mathcal{S}).$$

Since $|\mathcal{S}| + b(\mathcal{S}) \leq 2|\mathcal{S}|$, the above statement will imply the promised inequality and the theorem.

We prove the statement by induction on $|\mathcal{S}| + 2b(\mathcal{S})$. If $|\mathcal{S}| + 2b(\mathcal{S}) \leq 2$ then $\mathcal{S}$ is already a Subset-Fixed-Flood-It solution, so the statement is trivial. Suppose then that the statement holds when $|\mathcal{S}| + 2b(\mathcal{S}) \leq n$ and we have a solution $\mathcal{S}$ with $|\mathcal{S}| + 2b(\mathcal{S}) = n + 1$. We consider the following cases:

- The first move (S_1, c_1) has $p \in S_1$. By the inductive hypothesis there is a Subset-Fixed-Flood-It solution of length at most $|\mathcal{S}| + b(\mathcal{S}) - 1$ for $(G, \mathbf{col}_1, p)$. We build a solution for Subset-Fixed-Flood-It by appending this solution to the move (S_1, c_1), since this is a valid move for Subset-Fixed-Flood-It.
- There exists a move (S_i, c_i) with $S_i = \mathbf{Comp}(\mathbf{col}_{i-1}, u)$, for some $u \in N(\mathbf{Comp}(\mathbf{col}_{i-1}, p)) \setminus \mathbf{Comp}(\mathbf{col}_{i-1}, p)$ such that $c_i = \mathbf{col}_{i-1}(p)$. That is, there exists a move that plays a vertex u that currently has a different color than p, and as a result of this move the component of u and p merge, because u receives the same color as p and u has a neighbor in the component of p.

Consider the first such move. We build a solution $\mathcal{S}'$ as follows: we keep moves $(S_1, c_1) \ldots (S_{i-1}, c_{i-1})$; we add the moves $(\mathbf{Comp}(\mathbf{col}_{i-1}, p), \mathbf{col}_{i-1}(u))$, $(\mathbf{Comp}(\mathbf{col}_{i-1}, p) \cup \mathbf{Comp}(\mathbf{col}_{i-1}, u), \mathbf{col}_{i-1}(p))$; we append the rest of the previous solution $(S_{i+1}, c_{i+1}), \ldots, (S_k, c_k)$.

To see that $\mathcal{S}'$ is still a valid solution we observe that $\mathbf{Comp}(\mathbf{col}_{i-1}, p) \cup \mathbf{Comp}(\mathbf{col}_{i-1}, u)$ is monochromatic and connected when we play it, and that the result of the first $i - 1$ moves, plus the two new moves is exactly $\mathbf{col}_i$. We also note that $|\mathcal{S}'| + b(\mathcal{S}') = |\mathcal{S}| + b(\mathcal{S})$ because we replaced one bad move with two good moves. However, $|\mathcal{S}'| + 2b(\mathcal{S}') < |\mathcal{S}| + 2b(\mathcal{S})$, hence by the inductive hypothesis there exists a Subset-Fixed-Flood-It solution of the desired length.

- There does not exist a move as specified in the previous case. We then show that this reduces to the first case. If no move as described in the previous case exists and the initial coloring is not already constant, $\mathcal{S}$ must have a move (S_i, c_i) where $\{p\} \subseteq S_i \subseteq \mathbf{Comp}(\mathbf{col}_0, p)$ and $c_i = \mathbf{col}_{i-1}(u)$ for $u \in N(\mathbf{Comp}(\mathbf{col}_0, p)) \setminus \mathbf{Comp}(\mathbf{col}_0, p)$. In other words, this is a good move (it changes the color of p), that adds a new vertex u to the connected monochromatic component of p. Such a move must exist, since if the initial coloring is not

constant, the initial component of p must be extended, and we assumed that no move that extends it by recoloring one of its neighbors exists.

Consider the first such good move (S_i, c_i) as described above. We build a solution $\mathcal{S}'$ as follows: the first move is $(\mathbf{Comp}(\mathbf{col}_0, p), \mathbf{col}_0(u))$, where u is, as described above, the neighbor of $\mathbf{Comp}(\mathbf{col}_0, p)$ with $\mathbf{col}_{i-1}(u) = c_i$. For $j \in [i-1]$ we add the move (S_j, c_j) if $u \notin S_j$, or the move $(\mathbf{Comp}(\mathbf{col}_{j-1}, u) \cup \mathbf{Comp}(\mathbf{col}_0, p), c_j)$ if $u \in S_j$. In other words, we keep other moves unchanged if they do not affect u, otherwise we add to them $\mathbf{Comp}(\mathbf{col}_0, p)$. We observe that these moves are valid since we maintain the invariant that $\mathbf{Comp}(\mathbf{col}_0, p)$ and u have the same color and since none of the first $i-1$ moves of $\mathcal{S}$ changes the color of p (since we selected the first such move). The result of these i moves is exactly $\mathbf{col}_i$. We now append the remaining move $(S_{i+1}, c_{i+1}), \ldots$, and we have a solution that starts with a good move, has the same length and the same (or smaller) number of bad moves as $\mathcal{S}$ and is still valid. We have therefore reduced this to the first case. □

As we mentioned before, this combinatorial theorem implies 2-approximability of Free-Flood-It for the cases where Fixed-Flood-It is polynomial-time solvable. Also, as Fixed-Flood-It admits a $(c_{\max} - 1)$ approximation [2],[2] we have a $2(c_{\max} - 1)$ approximation for Free-Flood-It.

Corollary 6.5 Free-Flood-It *admits a* $2(c_{\max} - 1)$ *approximation, where* $c_{\max}$ *is the number of used colors.*

7 Non-monotonicity of Free-Flood-It

We now consider the (non-)monotonicity of the problem. A game has the *monotonicity property* if no legal move makes the situation worse. That is, if Fixed-Flood-It (or Free-Flood-It) has the monotonicity property, then no single move increases the minimum number of steps to make the input graph monotone. We believe that the monotonicity of Fixed-Flood-It was known as folklore and used implicitly in the literature. On the other hand, we are not sure that the non-monotonicity of Free-Flood-It was widely known. The only result we are aware of is by Meeks and Scott [17] who showed that on paths Free-Flood-It has the monotonicity property. In the following, we show that Free-Flood-It loses its monotonicity property as soon as the underlying graph becomes a path with one attached vertex.

To be self-contained, we start with proving the following folklore, which says that Fixed-Flood-It is monotone.

Lemma 7.1 *Let* $(G, \mathbf{col})$ *is an instance of* Fixed-Flood-It *with pivot* p*. For every color* c*, it holds that* $OPT_{Fixed}(G, \mathbf{col}') \leq OPT_{Fixed}(G, \mathbf{col})$*, where* $(G, \mathbf{col}')$ *is the result of the move* (p, c)*.*

Proof: Let $(p, c_1), \ldots, (p, c_k)$ be an optimal solution for $(G, \mathbf{col})$. We show that this sequence is valid for $(G, \mathbf{col}')$ too. Let $\mathbf{col} = \mathbf{col}_0$ and for $i \geq 1$,

[2]Their proof was only for grids, but it works for the general case.

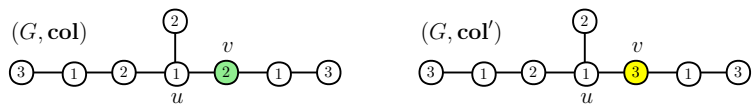


Figure 6: Non-monotonicity of Free-Flood-It.

let $\mathbf{col}_i$ be the coloring obtained from $\mathbf{col}_{i-1}$ by applying the ith move (p, c_i). We define $\mathbf{col}'_i$ in the analogous way. Observe that since all moves are played on the pivot p, we have $\mathbf{Comp}(\mathbf{col}_i, v) \in \{\mathbf{Comp}(\mathbf{col}, v), \mathbf{Comp}(\mathbf{col}_i, p)\}$ and $\mathbf{Comp}(\mathbf{col}'_i, v) \in \{\mathbf{Comp}(\mathbf{col}', v), \mathbf{Comp}(\mathbf{col}'_i, p)\}$ for every $v \in V(G)$ and for every i.

It suffices to show that $\mathbf{Comp}(\mathbf{col}_i, p) \subseteq \mathbf{Comp}(\mathbf{col}'_i, p)$ for all i. This obviously holds when $i = 0$. Assume that $\mathbf{Comp}(\mathbf{col}_i, p) \subseteq \mathbf{Comp}(\mathbf{col}'_i, p)$ for some $i \geq 0$. Let $v \in V(G) \setminus \{p\}$ such that $\mathbf{Comp}(\mathbf{col}, v)$ is not contained in but adjacent to $\mathbf{Comp}(\mathbf{col}_i, p)$, and thus contained in $\mathbf{Comp}(\mathbf{col}_{i+1}, p)$. Since $\mathbf{Comp}(\mathbf{col}_i, p) \subseteq \mathbf{Comp}(\mathbf{col}'_i, p)$, the monochromatic component $\mathbf{Comp}(\mathbf{col}, v)$ is either contained in or adjacent to $\mathbf{Comp}(\mathbf{col}'_i, p)$. Therefore, $\mathbf{Comp}(\mathbf{col}, v) \subseteq \mathbf{Comp}(\mathbf{col}'_{i+1}, p)$ holds, and thus $\mathbf{Comp}(\mathbf{col}_{i+1}, p) \subseteq \mathbf{Comp}(\mathbf{col}'_{i+1}, p)$. □

Meeks and Scott [17] showed the following monotonicity of Free-Flood-It on paths.

Proposition 7.2 ([17, Lemma 4.1]) *Let G be a path, $\mathbf{col}$ a vertex coloring of G, (v, c) a move, and $(G, \mathbf{col}')$ the result of the move. Then, $OPT_{Free}(G, \mathbf{col}') \leq OPT_{Free}(G, \mathbf{col})$.*

One may wonder whether the monotonicity property holds in general for Free-Flood-It. The following example, however, shows that it does not hold even for some graphs very close to paths. See the two instances in Figure 6. The instance $(G, \mathbf{col}')$ is obtained from $(G, \mathbf{col})$ by playing the move $(v, 3)$. We show that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) < \mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}')$.

Observe that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) = 3$: by Lemma 3.2, $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}) \geq 3$, and the sequence $(u, 2)$, $(u, 1)$, $(u, 3)$ floods the graph. Now suppose that $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}') \leq 3$. By Lemma 3.2, the first move in each optimal solution of $(G, \mathbf{col}')$ has to make the subgraph induced by some color connected, and then the second move has to remove the connected color. We can see that 2 is the only color that can play this role. If the first move is not played on u, then it is played on one of the two color-2 vertices and then the second move is played on the other color-2 vertex. Such a sequence of two moves cannot make either of color-1 or color-3 vertices connected. Thus by Lemma 3.2, it still needs at least two moves. Hence we can conclude that the first move is $(u, 2)$. Now the second move has to remove the color 2, and thus has to be played on u (or equivalently on any vertex in the monochromatic component including u). No matter which color we choose, we end up with an instance with at least two colors that are not connected. Again by Lemma 3.2, this instance needs more than one step, and thus $\mathrm{OPT}_{\mathrm{Free}}(G, \mathbf{col}') > 3$.

8 Concluding remark

The results in this paper somehow infer that Free-Flood-It is often at least as hard as Fixed-Flood-It. One may wonder this is always the case in some sense. Fellows et al. [8] asked the following question in their recent survey:

> *Is there any graph class for which Free-Flood-It can be solved in polynomial time, but Flood-It is NP-hard?*

where by Flood-It they mean Fixed-Flood-It. To tackle this problem, our results (e.g., Theorem 6.1) would be useful.

Acknowledgments

The authors thank the anonymous referees for their careful comments that improved the paper in many aspects.

References

[1] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009. doi:10.1016/j.jcss.2009.04.001.

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

[3] Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

[4] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

[5] Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and ids. *ACM Trans. Algorithms*, 11(2):13:1–13:20, 2014. doi:10.1145/2650261.

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

[7] Michael R. Fellows, Fábio Protti, Frances A. Rosamond, Maise Dantas da Silva, and Uéverton dos Santos Souza. Algorithms, kernels and lower bounds for the Flood-It game parameterized by the vertex cover number. *Discrete Applied Mathematics*, 245:94–100, 2018. doi:10.1016/j.dam.2017.07.004.

[8] Michael R. Fellows, Frances A. Rosamond, Maise Dantas da Silva, and Uéverton S. Souza. A survey on the complexity of flood-filling games. In *Adventures Between Lower Bounds and Higher Altitudes*, volume 11011 of *LNCS*, pages 357–376, 2018. doi:10.1007/978-3-319-98355-4_20.

[9] Rudolf Fleischer and Gerhard J. Woeginger. An algorithmic analysis of the honey-bee game. *Theor. Comput. Sci.*, 452:75–87, 2012. doi:10.1016/j.tcs.2012.05.032.

[10] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

[11] Hiroyuki Fukui, Yota Otachi, Ryuhei Uehara, Takeaki Uno, and Yushi Uno. On complexity of flooding games on graphs with interval representations. In *TJJCCGG 2012*, volume 8296 of *LNCS*, pages 73–84, 2013. doi:10.1007/978-3-642-45281-9_7.

[12] Jakub Gajarský, Michael Lampis, and Sebastian Ordyniak. Parameterized algorithms for modular-width. In *IPEC 2013*, volume 8246 of *LNCS*, pages 163–176, 2013. doi:10.1007/978-3-319-03898-8_15.

[13] 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.

[14] Aurélie Lagoutte, Mathilde Noual, and Eric Thierry. Flooding games on graphs. *Discrete Applied Mathematics*, 164:532–538, 2014. doi:10.1016/j.dam.2013.09.024.

[15] Michael Lampis. Algorithmic meta-theorems for restrictions of treewidth. *Algorithmica*, 64(1):19–37, 2012. doi:10.1007/s00453-011-9554-x.

[16] Ross M. McConnell and Jeremy P. Spinrad. Modular decomposition and transitive orientation. *Discrete Mathematics*, 201(1-3):189–241, 1999. doi:10.1016/S0012-365X(98)00319-7.

[17] Kitty Meeks and Alexander Scott. The complexity of flood-filling games on graphs. *Discrete Applied Mathematics*, 160(7–8):959–969, 2012. doi:10.1016/j.dam.2011.09.001.

[18] Kitty Meeks and Alexander Scott. The complexity of Free-Flood-It on $2 \times n$ boards. *Theor. Comput. Sci.*, 500:25–43, 2013. doi:10.1016/j.tcs.2013.06.010.

[19] Kitty Meeks and Alexander Scott. Spanning trees and the complexity of flood-filling games. *Theory Comput. Syst.*, 54(4):731–753, 2014. doi:10.1007/s00224-013-9482-z.

[20] Kitty Meeks and Dominik K. Vu. Extremal properties of flood-filling games. *CoRR*, abs/1504.00596, 2015. arXiv:1504.00596.

[21] Sang-il Oum. Approximating rank-width and clique-width quickly. *ACM Transactions on Algorithms*, 5, 2008. Article No. 10. doi:10.1145/1435375.1435385.

[22] Uéverton dos Santos Souza, Fábio Protti, and Maise Dantas da Silva. An algorithmic analysis of Flood-it and Free-Flood-it on graph powers. *Discrete Mathematics & Theoretical Computer Science*, 16(3):279–290, 2014. URL: http://dmtcs.episciences.org/2086.

[23] Marc Tedder, Derek G. Corneil, Michel Habib, and Christophe Paul. Simpler linear-time modular decomposition via recursive factorizing permutations. In *ICALP 2008 (1)*, volume 5125 of *LNCS*, pages 634–645, 2008. doi:10.1007/978-3-540-70575-8_52.

[24] Elad Verbin. Comment to "Is this game NP-Hard? by Sariel Har-Peled. http://sarielhp.org/blog/?p=2005#comment-993, 2009. Accessed: 2018-12-31.