ON-LINE $\mathcal{P}$-COLORING OF GRAPHS

Piotr Borowiecki

Faculty of Mathematics, Computer Science
and Econometrics, University of Zielona Góra
Prof. Z. Szafrana 4a, 65–516 Zielona Góra, Poland

e-mail: P.Borowiecki@wmie.uz.zgora.pl

Abstract

For a given induced hereditary property $\mathcal{P}$, a $\mathcal{P}$-coloring of a graph G is an assignment of one color to each vertex such that the subgraphs induced by each of the color classes have property $\mathcal{P}$. We consider the effectiveness of on-line $\mathcal{P}$-coloring algorithms and give the generalizations and extensions of selected results known for on-line proper coloring algorithms. We prove a linear lower bound for the performance guarantee function of any stingy on-line $\mathcal{P}$-coloring algorithm. In the class of generalized trees, we characterize graphs critical for the greedy $\mathcal{P}$-coloring. A class of graphs for which a greedy algorithm always generates optimal $\mathcal{P}$-colorings for the property $\mathcal{P} = K_3$-free is given.

Keywords: on-line algorithm, graph coloring, hereditary property.

2000 Mathematics Subject Classification: 05C15, 05C85, 68R10.

1. Introduction

In this paper, the concepts from two intensively studied frameworks, i.e., on-line coloring and generalized coloring of graphs, are combined to investigate the generalized on-line colorings. Among many generalizations of the classical graph coloring problem we are interested in on-line $\mathcal{P}$-colorings, where $\mathcal{P}$ is some induced additive hereditary property of graphs. All graphs considered in this paper are finite and simple, i.e., undirected, loopless and without multiple edges. We color vertices of graphs using single colors and deterministic algorithms.

1.1. On-line coloring

The instances of on-line problems are not given in advance and their successive parts become known over time. The solution is generated in a request-answer manner and each answer is based only on the already presented part of the instance. An *on-line algorithm* receives the sequence of requests $\sigma = (\sigma_1, \ldots, \sigma_n)$ and processes each request σ_i as soon as it is received, however, it has no right to modify an already generated part of the solution. It is usual to view the classical (proper) on-line coloring as a game of two adversaries called *Presenter* and *Painter*. We assume that Painter (representing on-line algorithm) does not know the structure of a graph to be colored. Presenter starts the game and reveals subsequent vertices of graph $G = (V, E)$ in some order $(v_1, \ldots, v_n)$ which is unknown to Painter. Vertex v_i is presented together with edges $E_i \subseteq E(G)$ adjacent to its already presented neighbors. Thus the request σ_i can be written as (v_i, E_i). Painter has to irrevocably assign a permissible color $c(v_i)$ to vertex v_i as soon as it is presented. In a sequence of alternate moves the goal of Painter is to use the smallest number of colors while the strategy of Presenter is to find the vertex ordering that forces Painter to use as many colors as possible. Presenter wins on G if the number of colors used by Painter is greater than the chromatic number $\chi(G)$. The readers interested in more details are referred to [3, 5] for surveys on classical on-line colorings.

1.2. $\mathcal{P}$-coloring

A *graph property* $\mathcal{P}$ is a nonempty isomorphism-closed subclass of all finite simple graphs. We say that a graph G has a property $\mathcal{P}$ if $G \in \mathcal{P}$. A property $\mathcal{P}$ of graphs is said to be *induced hereditary* if whenever $G \in \mathcal{P}$ and H is a vertex induced subgraph of G, then $H \in \mathcal{P}$. A property $\mathcal{P}$ is called *additive* if for each graph G all of whose components have the property $\mathcal{P}$ it follows that G has the property $\mathcal{P}$, too. All properties investigated in this paper are additive induced hereditary properties and the set of all these properties is denoted by $\boldsymbol{M}^a$, while the set of all *minimal forbidden subgraphs* characterizing the property $\mathcal{P} \in \boldsymbol{M}^a$ can be defined as follows:

$$\boldsymbol{C}(\mathcal{P}) = \{G \notin \mathcal{P} : \text{ each proper induced subgraph } H \text{ of } G \text{ belongs to } \mathcal{P}\}.$$

For example, edgeless graphs can be defined by $\boldsymbol{C}(\mathcal{P}) = \{K_2\}$ while for forests $\boldsymbol{C}(\mathcal{P}) = \{C_n : n \geq 3\}$. For any graph $G = (V, E)$ and arbitrarily chosen hereditary properties $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$ we define $(\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k)$-*partition*

of graph G as a partition $(V_1, V_2, \ldots, V_k)$ of the vertex set $V(G)$ such that each induced subgraph $G[V_i]$ has the property $\mathcal{P}_i$. In particular, if $\mathcal{P}_1 = \mathcal{P}_2 = \ldots = \mathcal{P}_k = \mathcal{P}$, then the partition is called $\mathcal{P}$-*partition* or equivalently $\mathcal{P}$-*coloring* of G. Note that $\mathcal{P}$-coloring for $\boldsymbol{C}(\mathcal{P}) = \{K_2\}$ is exactly classical, proper coloring. $\mathcal{P}$-coloring with k colors is called $(\mathcal{P}, k)$-*coloring* and the smallest k such that there exists a $(\mathcal{P}, k)$-coloring of G is called $\mathcal{P}$-*chromatic number of graph G*, denoted by $\chi^{\mathcal{P}}(G)$. Two vertices u and v are called $\mathcal{P}$-*adjacent* if there exists an induced subgraph H' such that $u, v \in V(H')$ and $H' \simeq H \in \boldsymbol{C}(\mathcal{P})$. It is easy to see that each subset V_i of any $\mathcal{P}$-partition is $\mathcal{P}$-*independent*, i.e., no two vertices of V_i are $\mathcal{P}$-adjacent. For a deeper discussion of hereditary properties we refer the reader to [2].

1.3. On-line $\mathcal{P}$-coloring

As in the case of proper on-line coloring the process of $\mathcal{P}$-coloring can be treated as a game of Presenter and Painter. The rules to follow are the same but a generalization of the winning criteria and permissibility of colors. Painter wins on G if no more than $\chi^{\mathcal{P}}(G)$ colors are used. Color p is called *permissible* for vertex v_i if its assignment to v_i results in a $\mathcal{P}$-coloring of the graph induced by $\{v_1, \ldots, v_i\}$. In other words, subgraph $G[V_p \cup \{v_i\}]$, induced by v_i and all vertices colored p, has the property $\mathcal{P}$ or equivalently $V_p \cup \{v_i\}$ is $\mathcal{P}$-independent. As many other on-line problems, on-line $\mathcal{P}$-coloring can be described using the request-answer pattern. Namely,

Problem [On-line $\mathcal{P}$-coloring]
Request : given $\sigma_i = (v_i, E_i)$, assign color to v_i.
Answer : assignment of a permissible color to vertex v_i.
  Goal : minimize the number of colors used.

In the general case Presenter seems to be more powerful than Painter. See the following example where Painter colors vertices greedily, i.e., always uses the smallest possible color.

Example 1.1. Let the property $\mathcal{P}$ be given by $\boldsymbol{C}(\mathcal{P}) = \{K_3\}$. Consider the sequence of requests $\sigma = (\sigma_1, \ldots, \sigma_7)$ such that $\sigma_1 = (v_1, \emptyset)$, $\sigma_i = (v_i, \{v_{i-1}v_i\})$ for $i = 2, 3, 4$ and $\sigma_5 = (v_5, \{v_1v_5\}, \{v_2v_5\})$, while for σ_6 and σ_7 the vertices are presented with the edges adjacent to v_3, v_4, v_5 and v_2, v_3, v_5, v_6 respectively. Greedy strategy of Painter results in the color assignment c such that $c(v_1) = \ldots = c(v_4) = 1$ (note that $\{v_1, \ldots, v_4\}$ is

$\mathcal{P}$-independent), $c(v_5) = c(v_6) = 2$ (to avoid monochromatic K_3) and finally $c(v_7) = 3$. Presenter wins since $\chi^{\mathcal{P}}(G)$ is equal to 2.

Greediness is one of the most popular approaches to the classical on-line graph coloring and has gained popularity mainly because of its simplicity and reasonable effectiveness. As demonstrated in the example, the principle of the smallest color assignment may be useful for $\mathcal{P}$-coloring and the algorithm following this rule will be referred to as `First-Fit`$(\mathcal{P})$ or `FF`$(\mathcal{P})$ for short. More formally it can be presented as follows:

Algorithm `First-Fit` $(\mathcal{P})$;
```
BEGIN
1   INITIALIZE (V(G):=∅, E(G):=∅, i:=0 );
2   REPEAT
3       i := i + 1;
3       READ(σ_i(v_i, E_i));
4       V(G) := V(G)∪{v_i};
5       E(G) := E(G)∪E_i;
6       k := 1;
7       WHILE v_i.color is not assigned DO
8           IF   G[V_k ∪ {v_i}] ∈ P
9               THEN v_i.color := k
10              ELSE k := k + 1;
11  UNTIL end of σ;
END.
```

A graph coloring algorithm is called *stingy* if for each vertex it tries to assign one of the already used colors (not necessarily the smallest one). Note that $\mathrm{FF}(\mathcal{P}) \in \mathcal{SA}$, where $\mathcal{SA}$ denotes the family of all stingy on-line algorithms.

The family $S = \{S_1, \ldots, S_p\}$ of *subgraphs forcing at* $v \in V(G)$ is defined as a subset of $\{H' : H' \leq G,\ v \in V(H'),\ H'$ is isomorphic to some $H \in \mathbf{C}(\mathcal{P})\}$ such that for any two distinct S_i, S_j we have $V(S_i) \cap V(S_j) = \{v\}$.

Property 1.1. *Let* $(V_1, V_2 \ldots, V_k)$ *be* $(\mathcal{P}, k)$*-coloring of* G *generated by algorithm* $\mathrm{A} \in \mathcal{SA}$. *Then for each* $i = 2, \ldots, k$ *there exists a family of subgraphs forcing at some vertex* $v_i \in V_i$.

We say that color c *surrounds vertex* v if there exists a subset U of vertices already colored c such that $G[U \cup \{v\}] \simeq H$ for some $H \in \mathbf{C}(\mathcal{P})$.

Note that each vertex v_i mentioned in Property 1.1 is surrounded by colors $1, \ldots, i-1$ and every stingy algorithm uses a new (still unused) color only if all previously used colors surround v_i.

Property 1.2. *Let $(V_1, V_2 \ldots, V_k)$ be $(\mathcal{P}, k)$-coloring of G generated by $\mathrm{FF}(\mathcal{P})$. If v is arbitrarily chosen vertex such that $c(v) \geq 2$, then each color $1, 2, \ldots, c(v) - 1$ surrounds v.*

It follows that for each vertex v such that $c(v) \geq 2$ there exists the family $\{S_1, \ldots, S_{c(v)-1}\}$ of subgraphs forcing at v.

A graph G with the fixed ordering of a vertex set is called the *on-line presentation of G*. If we take all possible on-line presentations of G, then the maximum number of colors used by A on G is called the *on-line $\mathcal{P}$-chromatic number of graph G for algorithm* A and it is denoted $\chi_{\mathrm{A}}^{\mathcal{P}}(G)$. Extending the scope we define the *on-line $\mathcal{P}$-chromatic number of graph G for the family $\mathcal{A}$* of on-line algorithms as the minimum of $\chi_{\mathrm{A}}^{\mathcal{P}}(G)$ taken over all algorithms $\mathrm{A} \in \mathcal{A}$. In symbols

$$\chi_{\mathcal{A}}^{\mathcal{P}}(G) = \min_{\mathrm{A} \in \mathcal{A}} \chi_{\mathrm{A}}^{\mathcal{P}}(G).$$

It is not hard to see that the following inequalities hold:

$$\chi^{\mathcal{P}}(G) \leq \chi_{\mathcal{A}}^{\mathcal{P}}(G) \leq \chi_{\mathrm{A}}^{\mathcal{P}}(G).$$

We say that an on-line $\mathcal{P}$-coloring algorithm A is *effective* for a family of graphs $\mathcal{G}$, if there exists a function $f(\chi^{\mathcal{P}})$ such that all graphs $G \in \mathcal{G}$ satisfy $\chi_{\mathrm{A}}^{\mathcal{P}}(G) \leq f(\chi^{\mathcal{P}}(G))$.

2. On-Line $\mathcal{P}$-Coloring of $\mathcal{P}$-Trees

The concept of $\mathcal{P}$-tree is a natural generalization of the classical tree notion.

Definition 2.1. Graph K_1 and every graph $H \in \boldsymbol{C}(\mathcal{P})$ is a $\mathcal{P}$-tree. Moreover, if T and T' are $\mathcal{P}$-trees, then $T \diamond T'$ is a $\mathcal{P}$-tree, where $\diamond$ denotes identification of any two vertices $u \in V(T)$ and $u' \in V(T')$.

The analysis of $\mathcal{P}$-colorings for $\mathcal{P}$-trees strongly relies on the properties of forcing $\mathcal{P}$-trees. The family $\mathcal{T}$ of forcing $\mathcal{P}$-trees is partitioned into subfamilies called levels. The *k-th* level is denoted by $\mathcal{T}_k$, while $T_{(i,k)}$ is used to

denote the i-th forcing $\mathcal{P}$-tree of level $\mathcal{T}_k$. The partition follows naturally from the following construction:

Every forcing $\mathcal{P}$-tree $T_{(i,k)}$ can be described by recursively defined sequences $\mathbf{H}_{(i,k)}$ and $\mathbf{R}_{(i,k)}$ and we can use $T_{(i,k)}(\mathcal{P}, \mathbf{H}_{(i,k)}, \mathbf{R}_{(i,k)})$ instead of $T_{(i,k)}$ when special emphasis on its structure is required. Let $H_{(i,k-1)}, k > 1$ be isomorphic to some $H \in \mathbf{C}(\mathcal{P})$ and let $t = |V(H_{(i,k-1)})|$. Accordingly, $\mathbf{H}_{(i,k)} = (H_{(i,k-1)}, \mathbf{H}_{(j_1,k-1)}, \ldots, \mathbf{H}_{(j_t,k-1)})$ and $\mathbf{R}_{(i,k)} = (R_{(i,k-1)}, \mathbf{R}_{(j_1,k-1)}, \ldots, \mathbf{R}_{(j_t,k-1)})$, where $R_{(i,k-1)}$ is a subset of $V(T_{(i,k)})$. We assume that for $k = 1$ there is only one forcing $\mathcal{P}$-tree $T_{(1,1)}(\mathcal{P}, \mathbf{H}_{(1,1)}, \mathbf{R}_{(1,1)}) \simeq K_1$ where $\mathbf{H}_{(1,1)} = (H_{(1,0)}, ())$ and $\mathbf{R}_{(1,1)} = (R_{(1,0)}, ())$, whereas $H_{(1,0)} \simeq K_1$, $R_{(1,0)} = V(T_{(1,1)})$ and () denotes an empty sequence. For $k \geq 2$ the forcing $\mathcal{P}$-tree $T_{(i,k)}$ of level $\mathcal{T}_k$ is constructed as follows:

(1) Let $H_{(i,k-1)}$ be isomorphic to any graph from $\mathbf{C}(\mathcal{P})$ and let $V(H_{(i,k-1)}) = \{u_1, \ldots, u_t\}$.

(2) Take t vertex-disjoint graphs $T_{(i_1,k-1)}, \ldots, T_{(i_t,k-1)}$ such that each of them is isomorphic to some graph from $\mathcal{T}_{k-1}$, however $T_{(i_p,k-1)} \simeq T_{(i_q,k-1)}$ is allowed for any $1 \leq i_p, i_q \leq t$.

(3) Create $R_{(i,k-1)} = \{r_1, \ldots, r_t\}$ taking exactly one vertex from the set $R_{(i_p,k-2)}$ of each graph $T_{(i_p,k-1)}$ selected in the step (2).

(4) Add an edge for any pair of vertices $r_p, r_q \in R_{(i,k-1)}$ if and only if $u_p u_q \in E(H_{(i,k-1)})$.

Note that the result of the step (3) may not be unique and that the final result depends on the labeling of $V(H_{(i,k-1)})$. See Figure 1 for an example of the construction. For proper coloring there is exactly one forcing $\mathcal{P}$-tree $T_{(1,k)}$ for each level $\mathcal{T}_k$ and in this case the construction gives canonical trees defined by Gyárfás and Lehel in [4].

Let $T_{(i,k)}$ be one of the forcing $\mathcal{P}$-trees for some property $\mathcal{P} \in \mathbf{M}^a$. Assume x to be a vertex whose family of the forcing subgraphs $S = \{S_1, S_2, \ldots, S_{k-1}\}$ has a maximum order and let vertices of S_j be labeled $\{x, v_1^j, v_2^j, \ldots, v_{p_j-1}^j\}$, where $p_j = |V(S_j)|$.

Lemma 2.1. *For each vertex $x \in R_{(i,k-1)}$, $k \geq 2$ there exists a family S of subgraphs forcing at x such that the removal of x and all edges of the subgraph $T_{(i,k)}[\bigcup_{j=1}^{k-1} V(S_j)]$ from $T_{(i,k)}$ results in the graph having $\sum_{j=1}^{k-1}(p_j - 1)$ components $F_1^1, F_2^1, \ldots, F_{p_1-1}^1, \ldots, F_1^j, F_2^j, \ldots, F_{p_j-1}^j, \ldots, F_1^{k-1}, F_2^{k-1}, \ldots, F_{p_{k-1}-1}^{k-1}$,*

each F_p^j being isomorphic to some forcing $\mathcal{P}$-tree from $\mathcal{T}_j$ and such that each F_p^j contains exactly one $v_p^j \in S_j$.
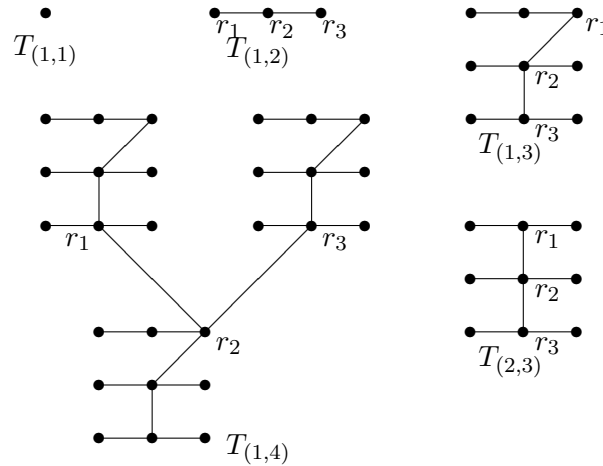

Figure 1. Selected forcing $\mathcal{P}$-trees for $\boldsymbol{C}(\mathcal{P}) = \{P_3\}$.

Proof. The existence of S follows immediately from the construction of forcing $\mathcal{P}$-trees. It is enough to see that for each $j = 1, \ldots, k-1$ removing from $T_{(i,k)}$ the edges of subgraph S_j results in a graph having $p_j - 1$ components, each isomorphic to some forcing $\mathcal{P}$-tree of level j and one component containing vertex x. See Figure 2 for the example of $T_{(1,4)}$ forcing $\mathcal{P}$-tree for the property given by $\boldsymbol{C}(\mathcal{P}) = \{P_4, K_3\}$. Note that $V(S_3) = R_{(1,3)} = \{x, v_1^3, v_2^3, v_3^3\}$, $V(S_2) = R_{(1,2)} = \{x, v_1^2, v_2^2\}$, $V(S_1) = R_{(1,1)} = \{x, v_1^1, v_2^1, v_3^1\}$, $R_{(1,0)} = \{x\}$. $\blacksquare$

Theorem 2.1. *Let $\mathcal{P} \in \boldsymbol{M}^a$ and let $T_{(i,k)}$ be any forcing $\mathcal{P}$-tree of level k. Then for every algorithm $\mathtt{A} \in \mathcal{SA}$ we have $\chi_{\mathtt{A}}^{\mathcal{P}}(T_{(i,k)}) \geq k$.*

Proof. We have to prove that for any algorithm $\mathtt{A} \in \mathcal{SA}$ and each vertex $y \in R_{(i,k-1)}$ there exists an on-line presentation of $T_{(i,k)}$ forcing $\mathtt{A}$ to use color k for y. The required ordering of the vertices follows directly from the construction of forcing $\mathcal{P}$-trees. Let y be the first vertex in order. Then for each level $j = k, \ldots, 2$ include all still unprocessed vertices from $R_{(p,j-1)}$ of each forcing subtree of level j. The subset of all vertices processed for the
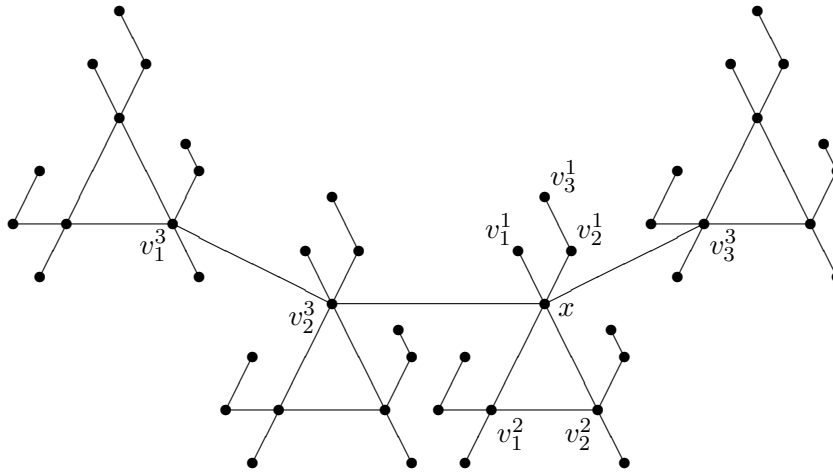
Figure 2.   Forcing $\mathcal{P}$-tree of 4-th level for $\boldsymbol{C}(\mathcal{P}) = \{P_4, K_3\}$.

same value of j is called the *layer*. Ordering of the vertices within a layer
is irrelevant. Finally, reverse the ordering. It is not hard to see that any
stingy algorithm A colors vertices of the first layer using the same color.
Assume that for layers $j = 1, \ldots, k - 2$ the ordering forces any stingy A to
use color j for all vertices of layer j. Note that each vertex x of layer $k - 1$
belongs to the set $R_{(p,k-1)}$ of some forcing $\mathcal{P}$-tree of level k and, according
to Lemma 2.1, there exists the family $\{S_1, \ldots, S_{k-1}\}$ of subgraphs forcing
at x. Moreover, each vertex $v_p^j \in V(S_j)\backslash\{x\}$, $j \leq k - 2$ belongs to the
layer j and because of the layer by layer presentation is already colored j.
Therefore each vertex of layer $k - 1$ is surrounded by colors $1, \ldots, k - 2$ and
any on-line algorithm is forced to assign them colors not smaller than $k - 1$.
However, since each layer is $\mathcal{P}$-independent, any stingy algorithm A will use
color $k - 1$ for all of its vertices and color k for the last vertex y.   ∎

Theorem 2.2. *Let $\mathcal{P} \in \boldsymbol{M}^a$ and let T be an arbitrary $\mathcal{P}$-tree. Then
$\chi_{\mathrm{FF}}^{\mathcal{P}}(T) > k$ if and only if T contains an induced subgraph isomorphic to
one of the forcing $\mathcal{P}$-trees of level $\mathcal{T}_{k+1}$.*

Proof. ($\Leftarrow$) If T contains a subgraph isomorphic to some $T_{(i,k+1)} \in \mathcal{T}_{k+1}$,
then by Theorem 2.1 we have $\chi_{\mathrm{FF}}^{\mathcal{P}}(T_{(i,k+1)}) > k$, hence $\chi_{\mathrm{FF}}^{\mathcal{P}}(T) > k$.

($\Rightarrow$) It is easy to check that the theorem holds for $k = 1$. Let $k \geq 1$
and assume that whenever $\chi_{\mathrm{FF}}^{\mathcal{P}}(T) > k - 1$, then T contains a subgraph

isomorphic to some $T_{(i,k)} \in \mathcal{T}_k$. Let $\chi_{\mathsf{FF}}^{\mathcal{P}}(T) > k$. Then there exists an on-line presentation of T such that $\mathsf{FF}(\mathcal{P})$ assigns color $k+1$ to some vertex $x \in V(T)$. By Property 1.2 there exists the family $\{S_1, \ldots, S_k\}$ of subgraphs forcing at x such that the colors of all vertices $v_p^j \in V(S_j)\backslash\{x\}$, $j = 1, \ldots, k$ are equal to j. If edges of the subgraph induced by $\bigcup_{j=1}^{k} V(S_j)$ were removed then we would have a disconnected graph such that each component T_p^j, $j = 1, \ldots, k$, $p = 1, \ldots, p_j - 1$ would contain exactly one vertex v_p^j colored j. Consequently, $\chi_{\mathsf{FF}}^{\mathcal{P}}(T_p^j) > j - 1$ and T_p^j contains a subgraph H_p^j isomorphic to some forcing $\mathcal{P}$-tree $T_{(q,j)}$ of level $\mathcal{T}_j$. As we see from the construction of forcing $\mathcal{P}$-trees and the proof of Theorem 2.1 each vertex $v_p^j \in R_{(q,j-1)}$ of the appropriate $T_{(q,j)}$, $j = 1, \ldots, k$. Hence T contains the subgraph induced by $\bigcup_{j=1}^{k} \bigcup_{p=1}^{p_j - 1} V(H_p^j) \cup \{x\}$ being isomorphic to one of the forcing $\mathcal{P}$-trees of level $\mathcal{T}_{k+1}$. ∎

Theorem 2.3. *Let* $\mathcal{P} \in \boldsymbol{M}^a$. *Then for any* $\mathcal{P}$*-tree* T *we have* $\chi_{\mathcal{SA}}^{\mathcal{P}}(T) = \chi_{\mathsf{FF}}^{\mathcal{P}}(T)$.

Proof. Let k be the largest integer such that there exists a subgraph of T isomorphic to some forcing $\mathcal{P}$-tree $T_{(i,k)}$. By Theorem 2.1 for every $\mathtt{A} \in \mathcal{SA}$ we have $\chi_{\mathtt{A}}^{\mathcal{P}}(T_{(i,k)}) \geq k$. Since T contains no subgraph isomorphic to forcing $\mathcal{P}$-tree of level $k+1$, by Theorem 2.2 it follows that $\chi_{\mathsf{FF}}^{\mathcal{P}}(T) \leq k$. Consequently, $\chi_{\mathtt{A}}^{\mathcal{P}}(T_{(i,k)}) \geq \chi_{\mathsf{FF}}^{\mathcal{P}}(T)$. On the other hand, since $T_{(i,k)}$ is an induced subgraph of T, we have $\chi_{\mathtt{A}}^{\mathcal{P}}(T) \geq \chi_{\mathtt{A}}^{\mathcal{P}}(T_{(i,k)})$. It follows that $\chi_{\mathtt{A}}^{\mathcal{P}}(T) \geq \chi_{\mathsf{FF}}^{\mathcal{P}}(T)$ and according to the minimum in the definition of $\chi_{\mathcal{A}}^{\mathcal{P}}$ we get $\chi_{\mathcal{SA}}^{\mathcal{P}}(T) = \chi_{\mathsf{FF}}^{\mathcal{P}}(T)$. ∎

It is known that there exist properties $\mathcal{P} \in \boldsymbol{M}^a$ such that for any graph G we have $\chi_{\mathcal{OL}}^{\mathcal{P}}(G) = \chi_{\mathsf{FF}}^{\mathcal{P}}(G)$, where $\mathcal{OL}$ stands for the family of all on-line algorithms (see [4] for the results obtained for proper coloring). It seems that nontrivial results for $\mathcal{OL}$ are very hard to prove in the scope of all $\mathcal{P} \in \boldsymbol{M}^a$. On the other hand, interesting problems arise for classes of algorithms and specific presentation strategies. The proof of the next theorem relies on a very natural strategy of graph presentation and the problem is whether there exist algorithms forced by this presentation but resist layer by layer presentation given in the proof of Theorem 2.1.

Theorem 2.4. *Let* $\mathcal{P} \in \boldsymbol{M}^a$. *Then for any forcing* $\mathcal{P}$*-tree* $T_{(i,k)}$ *of level* k *we have* $\chi_{\mathsf{FF}}^{\mathcal{P}}(T_{(i,k)}) \geq k$.

Proof. Let $x \in R_{(i,k-1)}$. We give the strategy of Presenter forcing $\mathtt{FF}(\mathcal{P})$ to use color k for x. According to Lemma 2.1 at every level of recursion, the procedure $\mathtt{Presentation}$ finds the family S forcing at x and subgraphs H_p^j induced by the vertices of components F_p^j.

Algorithm $\mathtt{Presentation}$ $(T_{(i,k)}, x)$;
BEGIN
1   Find family S of subgraphs forcing at x and subgraphs H_p^j;
2   FOR p := 1 TO $p_1 - 1$ DO
          Present the vertices of subgraph H_p^1;
3   FOR each subgraph H_p^j, $j \geq 2$ DO
          $\mathtt{Presentation}$ (H_p^j, v_p^j);
4   Present vertex x;
END.

It is easy to see that $\mathtt{Presentation}$ works fine for $k = 1, 2$. Let us assume that $k \geq 2$ and that for $j = 1, \dots, k-1$ it forces $\mathtt{FF}(\mathcal{P})$ to use color j at vertex x of each forcing $\mathcal{P}$-tree of level j. Independently of the order, all vertices of H_p^1 are colored 1. Since each H_p^j is isomorphic to some forcing $\mathcal{P}$-tree of level j, vertex v_p^j of H_p^j is colored j. Hence for each $j = 1, \dots, k-1$ all vertices of $V(S_j) \backslash \{x\}$ are colored j and since all of them are $\mathcal{P}$-neighbors of x, Painter is forced to use color k for x.   ■

3.   Effectiveness of Stingy On-Line Algorithms

Since for any property $\mathcal{P} \in \boldsymbol{M}^a$ there exists some $(\mathcal{P}, 2)$-coloring of every $\mathcal{P}$-tree, it follows from Theorem 2.1 that for any $\mathtt{A} \in \mathcal{SA}$ the difference between the worst and optimum solution values $\chi_{\mathtt{A}}^{\mathcal{P}}(G) - \chi^{\mathcal{P}}(G)$ may be arbitrarily large. This implies the following corollary:

Corollary 3.1. *Let $\mathcal{P} \in \boldsymbol{M}^a$. Then there does not exist any effective stingy on-line $\mathcal{P}$-coloring algorithm.*

The *performance guarantee* function ρ for algorithm $\mathtt{A}$ is defined as follows:

$$\rho_{\mathtt{A}}^{\mathcal{P}}(n) = \max\{\chi_{\mathtt{A}}^{\mathcal{P}}(G)/\chi^{\mathcal{P}}(G) : G \text{ is a graph of order } n\}.$$

The lower bound of the performance guarantee function for the algorithms from $\mathcal{OL}$ and proper coloring, was given by Bean [1]. For the stingy

algorithms Bean's result can be easily extended for any property $\mathcal{P} \in \boldsymbol{M}^a$. The extended lower bound is

$$\rho_{\mathtt{A}}^{\mathcal{P}}(n) \geq \frac{\log_\alpha n}{2},$$

where $\alpha = \min\{|V(H)| : H \in \boldsymbol{C}(\mathcal{P})\}$. The bound follows directly from Theorem 2.1 which in fact states that for any integer $k > 0$ there exists a $\mathcal{P}$-tree $T_{(i,k)}$ of order α^{k-1} and its on-line presentation which forces any stingy $\mathtt{A}$ to use at least k colors. However, the aforementioned bound, being in fact an easy consequence of the results presented in the preceding section, can be significantly improved.

Theorem 3.1. *Let $\mathcal{P} \in \boldsymbol{M}^a$ and let $\alpha = \min\{|V(H)| : H \in \boldsymbol{C}(\mathcal{P})\}$. For any integer $k > 0$ there exists a graph G_k of order $(2k-1)(\alpha-1)$ such that for any $\mathtt{A} \in \mathcal{SA}$ we have $\chi_{\mathtt{A}}^{\mathcal{P}}(G_k) \geq k$.*

Proof. A graph $G_k = (V, E)$ is any graph that satisfies the following conditions:

(a) $V(G_k)$ can be partitioned into $2k - 1$ subsets $(B_1^1, B_2^1, \ldots, B_k^1, B_1^2, B_2^2, \ldots, B_{k-1}^2)$ each inducing a subgraph isomorphic to some $H - x$, where $H \in \boldsymbol{C}(\mathcal{P})$ and x is arbitrary vertex of H.

(b) Both $\bigcup_{i=1}^{k} B_i^1$ and $\bigcup_{j=1}^{k-1} B_j^2$ are $\mathcal{P}$-independent.

(c) For every $v \in B_i^1$, $i = 2, \ldots, k$ each subgraph induced by $\{v\} \cup B_j^2$, $j > i$ is isomorphic to some $H \in \boldsymbol{C}(\mathcal{P})$.

(d) For every $u \in B_j^2$, $j = 2, \ldots, k-1$ each subgraph induced by $\{u\} \cup B_i^1$, $i > j$ is isomorphic to some $H \in \boldsymbol{C}(\mathcal{P})$.

Since $B_1^1 \cup B_1^2$ is $\mathcal{P}$-independent, any stingy $\mathtt{A}$ will assign the same color to all of its vertices. Let us assume that for $i = 1, \ldots, k-2$ all vertices of B_i^1 and B_i^2 have been colored using color i. Since each vertex $v \in B_{k-1}^1$ is $\mathcal{P}$-adjacent to all vertices of the sets B_j^2, $j = 1, \ldots, k-2$ (it is surrounded by colors $1, \ldots, k-2$), any stingy $\mathtt{A}$ will assign it color $k - 1$. Similarly, each $u \in B_{k-1}^2$ will be colored $k - 1$ and finally, each $v \in B_k^1$ will get color k. ∎

The graphs G_k are easily shown to be off-line $(\mathcal{P}, 2)$-colorable (see condition (b)), hence for any $\mathtt{A} \in \mathcal{SA}$ we have

$$\rho_{\mathtt{A}}^{\mathcal{P}}(n) > \frac{n}{4(\alpha-1)}.$$

4.   Optimal Greedy $\mathcal{P}$-Coloring

Despite of all these rather pessimistic results there exist families of graphs for which $\mathtt{FF}(\mathcal{P})$ always gives optimal $\mathcal{P}$-colorings.

Let $\mathcal{F}^*$ be the family of graphs whose vertex sets can be partitioned into two subsets C and I such that C induces a complete subgraph K_3 while I is K_3-free and for every $v \in C$ there exist $x, y \in I$ such that $G[\{x, v, y\}] \simeq K_3$. Let us define a partially ordered set $(\mathcal{F}^*, \leq)$ with respect to ordering relation $\leq$ of being an induced subgraph. We write $\mathcal{F}$ for the set of all minimal elements of $\mathcal{F}^*$, with the excluded graph K_5, i.e., $\mathcal{F} = \min(\mathcal{F}^*, \leq) \backslash \{K_5\}$.

Lemma 4.1. *Let $G = (V, E)$ be an $\mathcal{F}$-free graph whose vertex set $V(G)$ can be partitioned into K and I such that the subgraph induced by K is complete, while I is K_3-free. Moreover, if each $v \in K$ has $\mathcal{P}$-neighbors $x, y \in I$ such that $G[\{x, v, y\}] \simeq K_3$, then there exists a pair of vertices $x, y \in I$ such that $G[K \cup \{x, y\}]$ is complete.*

Proof. Let x, y be adjacent vertices from I and let $A = \{u : u \in K, G[\{x, u, y\}] \simeq K_3\}$. Since lemma is obvious for $A = K$, suppose $A \neq K$ and let $B = K \backslash A$. Assume that the lemma is true for a subgraph induced by B and all vertices from I which are $\mathcal{P}$-adjacent with any vertex from B. It follows that there exist $u, v \in I$ such that $G[\{u, z, v\}] \simeq K_3$, for every $z \in B$. If there existed $a \in A$ and $b \in B$ such that $G[\{u, a, v\}] \neq K_3$ and $G[\{x, b, y\}] \neq K_3$ we would have an induced subgraph of G isomorphic to one of the graphs from $\mathcal{F}$. Hence either $G[K \cup \{x, y\}]$ or $G[K \cup \{u, v\}]$ is complete.   ■

Theorem 4.1. *If a property $\mathcal{P} \in \boldsymbol{M}^a$ is given by $\boldsymbol{C}(\mathcal{P}) = \{K_3\}$, then algorithm $\mathtt{FF}(\mathcal{P})$ gives an optimal $\mathcal{P}$-coloring of any $\mathcal{F}$-free graph.*

Proof. Let $(V_1, \ldots, V_k)$ be $(\mathcal{P}, k)$-coloring of graph $G = (V, E)$ generated by algorithm $\mathtt{FF}(\mathcal{P})$. If $\mathtt{FF}(\mathcal{P})$ assigned color k to vertex v, then by Property 1.2 there exist vertices $u_1, u_2 \in V_{k-1}$ such that $G[\{u_1, v, u_2\}] \simeq K_3$. Let p be any color such that $2 \leq p < k$ and let vertices $\{v, u_1, u_2, \ldots, u_{2(k-p)}\}$ induce the clique Q of order $2(k-p) + 1$, $\mathcal{P}$-colored using colors $p, \ldots, k$ (note that $\chi^{\mathcal{P}}(K_n) = \lceil n/2 \rceil$). By Property 1.2 for each vertex w of Q there exist $x, y \in V_{p-1}$ such that $G[\{x, w, y\}] \simeq K_3$. Lemma 4.1 applied to the vertices of Q and V_{p-1} implies the existence of $x, y \in V_{p-1}$ such that $\{v, u_1 \ldots, u_{2(k-p)}, x, y\}$ induces $K_{2(k-p)+3}$.   ■

A slight modification of the assumptions of Theorem 4.1 leads to a stronger version which, by similar arguments, can be proved for all properties $\mathcal{P} \in \boldsymbol{M}^a$. However, it should be pointed out that for some properties family $\mathcal{F}$ is infinite.

References

[1] D.R. Bean, *Effective coloration*, J. Symbolic Logic **41** (1976) 469–480.

[2] M. Borowiecki, I. Broere, M. Frick, P. Mihók and G. Semanišin, *A survey of hereditary properties of graphs*, Discuss. Math. Graph Theory **17** (1997) 5–50.

[3] P. Borowiecki, *On-line coloring of graphs*, in: M. Kubale ed., Graph Colorings, Contemporary Mathematics 352 (American Mathematical Society, 2004) 21–33.

[4] A. Gyárfás and J. Lehel, *First-Fit and on-line chromatic number of families of graphs*, Ars Combinatoria **29C** (1990) 168–176.

[5] H.A. Kierstead, *Coloring Graphs On-line*, in: A. Fiat and G.J. Woeginger eds., Online Algorithms - The State of the Art, LNCS 1442 (Springer, Berlin, 1998) 281–305.