Saving colors and Max Coloring: some fixed-parameter tractability results

Bruno Escoffier

Saving colors and Max Coloring: some fixed-parameter tractability results*

Bruno Escoffier

Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6 UMR 7606, 4 place Jussieu, 75005 Paris, France
bruno.escoffier@lip6.fr

Abstract. Max coloring is a well known generalization of the usual Min Coloring problem, widely studied from (standard) complexity and approximation viewpoints. Here, we tackle this problem under the framework of parameterized complexity. In particular, we first show to what extend the result of [3] - saving colors from the trivial bound of n on the chromatic number - extends to Max Coloring. Then we consider possible improvements of these results by considering the problem of saving colors/weight with respect to a better bound on the chromatic number. Finally, we consider the fixed parameterized tractability of Max Coloring in restricted graph classes under standard parameterization.

Keywords: Graph coloring, Paremeterized complexity, FPT algorithms, Max coloring.

1 Introduction

As it is well known for decades, deciding whether a given simple graph is 3-colorable is an NP-complete problem. As an immediate consequence, dealing with parameterized complexity, the coloring problem parameterized by the value of the solution (number of colors) is not in XP unless $P = NP$. This result stimulates the study of this fundamental problem under other parameterizations. For instance, in [2, 15] is considered the coloring problem parameterized by the distance (adding/removing vertices or edges) of the input graph to a polynomial time solvable class of graphs. Chor et al. [3] considered the problem of "saving k colors", by addressing the following question: given a graph $G = (V, E)$ with n vertices and an integer k, is it possible to color G with $n - k$ colors? In other words, can we save k colors with respect to the trivial coloring (with n colors)? They show that the problem is FPT, and admits a kernel with a linear ($O(k)$) number of vertices.

On the other hand, among the many variants or generalizations of the coloring problem, Max Coloring (sometimes called Weighted Coloring) has been

* This is an extended version of a work presented at the 42nd International Workshop on Graph-Theoretic Concepts in Computer Science, WG 2016, [7].

deeply investigated in the last ten years, see for instance [1, 4, 5, 16, 17] and references therein. In this problem, coming from batch scheduling, each vertex v has a nonnegative integer weight $w(v)$; the weight of a color class (we consider in the article only proper colorings, i.e. two vertices sharing an edge cannot be in the same color class) is the *maximum* weight of its vertices, and the weight of a coloring is the sum of the weights of its color classes. The goal is to find a coloring of minimum weight.[1] This is a generalization of the usual coloring problem, which corresponds to the particular case where all the vertices have weight 1. Max Coloring is much harder than coloring; it is for instance NP-hard in bipartite graphs [5], in interval graphs [8], and has been recently shown even not solvable in polynomial time in trees under ETH [1].

The goal of this work is threefold, by addressing the following questions:

1. Is it possible to save weights for Max Coloring problem as it is possible to save colors in the usual coloring problem?
2. Is it possible to strengthen these results by starting with a less trivial upper bound on $\chi(G)$? This question holds for the usual coloring problem: can we save k colors not with respect to n but with respect to another bound $b < n$? If so, how does this extend to saving weights for Max Coloring?
3. Under the standard parameterization (number of colors), could we say something on Max Coloring in classes of graphs where the usual coloring problem is polynomial?

Let us first deal with the first point. The bound of n for min coloring consists of giving one color per vertex; this coloring has value $W = \sum_{i=1}^{n} w(v_i)$ for Max Coloring. Assume that weights are in non-increasing order. Then, there are two questions that naturally generalizes the question of coloring a graph with $n - k$ colors: given k, can we save a *total weight* of k, i.e., does there exist a coloring of weight at most $W - k$? Or, can we save k *vertex weights*, i.e., does there exist a coloring of weight at most $\sum_{i=1}^{n-k} w(v_i)$? We tackle these questions in Section 3 and show that both problems are FPT with respect to k.

Let us now consider the second point, and let us focus on the usual coloring problem. Saving k colors in FPT-time can be reached as follows: starting from the initial coloring with n colors we save colors by merging together pairs of non adjacent vertices. If it is possible to do this k times, then we have saved k colors. Otherwise, the graph is composed of a clique plus $O(k)$ vertices, and an optimal coloring is computable by matching techniques in FPT-time. Let us call a coloring *non trivial* if it is not possible to merge two colors of this coloring. It is very easy to build a non trivial coloring (in polynomial time), or to transform a coloring into a non trivial one without increasing the number of colors. Let c be the number of colors used by a non trivial coloring, and m be the number of edges in the graph. Since we have an edge between any pair of colors, we

[1] In batch scheduling, a color class is a set of tasks (batch) processed together; then the weight of the color class is the time to process the batch, and the total weight is the time to process every batch, i.e., every task.

have $m \geq c(c-1)/2$, meaning $\chi(G) \leq c \leq f(m) = \left\lfloor \frac{1}{2} + \sqrt{2m + \frac{1}{4}} \right\rfloor$. This bound $f(m)$ is generally much smaller than n, and it is worth considering the possibility to save colors with respect to this bound $f(m)$. In other words, the following question holds: *Given a graph G and an integer k, is it possible to color G with $f(m) - k$ colors?*

We will focus on this question in Section 4. Interestingly, we show that while the problem of saving k colors with respect to $f(m)$ is still FPT, it *does not* admit a kernel with a linear number of vertices under ETH, but (only) one with a quadratic number of vertices. Then we tackle the possible extension of this result to Max Coloring. More precisely, we consider the question of determining whether a coloring of weight at most $\sum_{i=1}^{f(m)} w(v_i) - k$ exists or not. We show that the problem is in XP. The core of the proof is a *polynomial time* reduction to a very specific class of graphs, namely graphs made of two cliques plus $O(k)$ vertices. This reduction should be also useful while determining if the problem is FPT, that we leave as an open question.

Then, in Section 5, we derive from previous works some results on Max Coloring under standard parameterization, in classes of graphs where min coloring is polynomial but Max Coloring remains hard: Max Coloring is FPT in chordal graphs (hence in interval graphs and trees), while it is not in XP for bipartite graphs (hence for perfect graphs). We give in Section 6 some concluding remarks.

2 Preliminaries

We give in this section several (standard) definitions dealing with graphs and parameterized complexity, that will be used along this article.

2.1 Graphs

We consider in this article undirected loopless graphs. A clique in a graph G is a set of pairwise adjacent vertices. An independent set is a set of pairwise non adjacent vertices. A (proper) coloring of G is a partition of the vertex set into independent sets (called color classes). As mentioned before, we will only consider proper colorings all along the article. The chromatic number of a graph is the minimum number of colors needed to properly color the vertices of the graph.

A graph is:

- Bipartite if its chromatic number is at most 2;
- An interval graph if we can associate to each vertex v_i an interval $I_i \subset R$ in such a way that v_i and v_j are adjacent if and only if I_i and I_j intersect;
- Chordal if every cycle on at least 4 vertices has a chord (edge linking two non consecutive vertices of the cycle).
- Perfect if in any induced subgraph G' of G the chromatic number of G' is equal to the size of a maximum clique in G'.

It is a well known fact that interval graphs are chordal, that bipartite graphs and chordal graphs are perfect, and that the usual coloring problem is polynomial in perfect graph (see for instance [10, 11]).

2.2 Complexity

The article focuses on parameterized complexity. A parameterized problem (see [6] for more details) is said to be:

- in the class FPT (or *fixed parameter tractable*) if it can be solved by an algorithm whose complexity is $f(k)n^{O(1)}$, for some function f, where n is the size of the instance and k the parameter.
- in the class XP if it can be solved by an algorithm whose complexity is $O(f(k)n^{g(k)})$, for some functions f and g, where n is the size of the instance and k the parameter.

We will use the term FPT-time (resp. XP-time) to denote time complexity $f(k)n^{O(1)}$ (resp. $O(f(k)n^{g(k)})$). Roughly speaking, XP denotes parameterized problems which can be solved in polynomial time for any fixed value k of the parameter (but the degree of the polynomial may depend on k). The class FPT further requires that the degree of the polynomial is independent of k; hence, the complexity is polynomial in n (with a fixed degree), while the dependency in the parameter is captured by a multiplicative factor $f(k)$.

A kernelization algorithm for a parameterized problem is a polynomial time algorithm which transforms an instance I with parameter k into an equivalent instance (called kernel) I' with parameter k', where $|I'| \leq g(k)$ for some function k. $g(k)$ is the size of the kernel; we say that a kernel is linear (resp. quadratic) if $g(k) = O(k)$ (resp. $g(k) = O(k^2)$). Dealing with graphs, we will be interested in the size of a kernel in terms of number of vertices and/or number of edges.

We will also use the exponential time hypothesis, saying that 3-SAT cannot be solved in subexponential time. More formally:

Exponential time hypothesis (ETH) [14]: there exists $\epsilon > 0$ such that there is no algorithm solving 3-SAT in time $O(2^{\epsilon n})$ where n is the number of variables.

3 Saving weight for Max Coloring

In the Max Coloring problem, the input is a vertex-weighted graph (G, w), where $w(v_i) \in \mathbb{N}^*$ denotes the weight of the vertex v_i. Given a coloring $\mathcal{C} = (S_1, S_2, \ldots, S_c)$ of G, the weight of a color S_j is $w(S_j) = \max\{w(v_i), v_i \in S_j\}$, and the weight of the coloring - to be minimized - is $w(\mathcal{C}) = \sum_{j=1}^{c} w(S_j)$. We assume that the vertices are ordered in non increasing weight, i.e. $w(v_1) \geq w(v_2) \geq \cdots \geq w(v_n)$. We also denote $W = \sum_{i=1}^{n} w(v_i)$.

As a generalization of the unweighted case, we consider the following problem.

Problem *Saving weight*

– Input: a vertex-weighted graph (G, w), and an integer k.
– Parameter: k
– Question: does there exist a coloring of G of weight at most $W - k$?

Let us also define the problem *Saving color weights* which is the same as *Saving weight* up to the fact that the question is now to determine if there exists a coloring of weight at most $\sum_{i=1}^{n-k} w(v_i)$.

We show that these problems are FPT. The technique generalizes the one for the unweighted case, by using matchings in a properly weighted graph. More precisely, we first show how matching techniques allow to solve Max Coloring in complement of bipartite graphs in polynomial time (this result will be also useful later in Section 4.2). Then we reduce the problem to the case where the graph has a very large clique, and use brute force to produce an FPT number of instances of Max Coloring in complement of bipartite graphs.

Lemma 1. *Max Coloring is polynomially solvable in the class of complement of bipartite graphs.*

Proof. Let G be a graph, whose vertices are partitioned into 2 cliques C_1 and C_2. We consider the complement graph $\overline{G}$ of G (same vertex set, and an edge is in $\overline{G}$ iff it is not in G), and we assign to edge (v_i, v_j) in $\overline{G}$ the weight $\min\{w(v_i), w(v_j)\}$. This corresponds to the amount of weight saved by using the same color for v_i and v_j.

Consider a matching in $\overline{G}$. This defines a coloring of G: an unmatched vertex constitutes one color, two matched vertices constitute one color. Reciprocally, any coloring of G defines a matching in $\overline{G}$. The weight of a matching M in $\overline{G}$ is $w(M) = \sum_{(v_i,v_j)\in M} \min\{w(v_i), w(v_j)\}$, and the weight of the associated coloring $\mathcal{C}$ is:

$$\begin{aligned} w(\mathcal{C}) &= \sum_{v\in\overline{M}} w(v) + \sum_{(v_i,v_j)\in M} \max\{w(v_i), w(v_j)\} \\ &= \sum_{v\in V} w(v) - \sum_{(v_i,v_j)\in M} \min\{w(v_i), w(v_j)\} = \sum_{v\in V} w(v) - w(M) \end{aligned}$$

where $\overline{M}$ is the set of vertices unmatched in M. So we can solve Max Coloring in G by computing a maximum weight matching in $\overline{G}$. □

Theorem 1. *Problems Saving weight and Saving color weights are FPT, solvable in time* $2^{O(k \log k)}p(n)$ *for some polynomial* p.

Proof. As for the unweighted case, consider first a maximal anti-matching (matching in the complement graph). If it contains at least k anti-edges, it means that we can save at least k weights, so a global weight at least k, and we answer YES (for both problems). Otherwise, we have $t < k$ anti-edges in the matching, meaning that the other $n - 2t$ vertices form a clique. Let C be this clique, and R be the endpoints of the anti-matching.

We consider all possible colorings of the vertices in R. Consider one of these colorings, with colors $\mathcal{C}_R = (R_1, \cdots, R_s)$. We contract in G all colors R_i, meaning that we replace vertices in R_i by one vertex r_i, adjacent to all neighbors of the contracted vertices, and assign the weight $w(r_i) = max\{w(v), v \in R_i\}$ to this new vertex. We also make the set of r_i's a clique. Clearly, there exists in G a coloring of weight at most B where vertices in R are colored as in $\mathcal{C}_R$ if and only if there exists a coloring of weight at most B in the contracted graph. This contracted graph is made of 2 cliques, so we can solve Max Coloring in polynomial time due to Lemma 1.

Then it is sufficient to produce all the possible colorings of R and to see if one allows to answer YES. There are at most $2^{O(t \log t)} = 2^{O(k \log k)}$ such colorings, so the result follows. □

Let us now speak about kernels. For saving colors in the usual coloring problem, a kernel with a linear number of vertices exists [3], and it is natural to investigate the size of kernels that can be reached for our generalized problem.

Theorem 2. *Problems Saving weight and Saving color weights admit a kernel with $O(k4^k)$ vertices.*

Proof. We first present the kernelization algorithm for *Saving weight*. As previsouly, let us start with a maximal anti-matching. If it contains at least k anti-edges, then we answer YES. Otherwise, we have $t < k$ anti-edges in the matching, let R be the $2t$ endpoints of this matching and C the remaining clique of size $n - 2t$.

Let $H = \{u_1, u_2, \ldots, u_h\}$ be a set of vertices in C, sorted in non decreasing weight order, which all have the same neighborhood. We consider the following reduction rule.

RR: if $f = h - 2t > 0$, then remove from G vertices $u_1, u_2, \ldots, u_f$ (k is not modified).

Note that if RR is valid, then after this reduction rule we get a graph where there are at most $2t$ vertices of C with the same neighborhood. Since C is a clique, there are at most 2^{2t} possible neighborhoods for vertices in C. Then $|C| \leq 2t4^t$. Since $|R| = 2t$ and $t \leq k$ the result follows.

It remains to show that RR is valid. Let us call (G', k) the graph obtained after one application of this rule, and let $M = \sum_{i=1}^{f} w(u_i)$. If there is a coloring of weight w' in G', then by coloring the removed vertices with a new color we get a coloring of G of weight $w = w' + M$.

Conversely, suppose that there is a coloring of G with weight w. In this coloring, since C is a clique, at least f vertices of H are isolated. Suppose that one vertex of H u_i is isolated, while another one u_j is not with $\delta = w(u_i) - w(u_j) > 0$. Since u_i and u_j have the same neighborhood, we can switch the color of these two vertices: u_j becomes isolated instead of u_i, so we reduce the weight of the coloring by δ, while in the other color we increase it by at most δ. The total weight does not increase.

This means that we can assume that the f vertices of H of smallest weight are isolated. Then, clearly the restriction of the coloring to G' has weight $w' = w - M$.

Hence, there is a coloring in G of weight (at most) w iff there is a coloring in G' of weight at most $w' = w - M$. Since the total weight of G equals the total weight of G' plus M, the result follows.

Now, let us consider *Saving color weights*. The beginning of the proof is the same, but we slightly modify the reduction rule in order to ensure that the removed vertices do not affect the "saved weights" (k lightest weights). To do this, we define RR2:

RR2: if $f = h-2t-k > 0$, then remove from G vertices $u_{k+1}, u_{k+2}, \ldots, u_{k+f}$ (k is not modified).

By the very same reasoning as previously, setting $M = \sum_{i=k+1}^{k+f} w(u_i)$, we know that there is a coloring in G of weight (at most) w iff there is a coloring in G' of weight at most $w' = w - M$. Since the k vertices $u_1, \ldots, u_k$ remains in G', we have that $\sum_{i=1}^{n'-k} w_{G'}(v'_i) = \sum_{i=1}^{n-k} w_G(v_i) - M$ (where, with a slight abuse of notation we subscript the graph to w, consider that v'_i are sorted in non increasing order in G' and v_i are sorted in non increasing order in G).

Then RR2 is valid. After this reduction rule we get a graph where there are at most $2t + k \leq 3k$ vertices of C with the same neighborhood. Since C is a clique, there are at most $2^{2t} \leq 4^k$ possible neighborhoods for vertices in C. Then $|C| \leq 3k4^k$. Since $|R| = 2t \leq 2k$ the result follows. □

The size of the previous kernel is exponential in k. We leave as open the question of getting a polynomial size kernel for problems *Saving weight* and *Saving color weights*.

Let us remark that it does not seem obvious to generalize the approach of [3] leading to a kernel with a linear number of vertices. Indeed, the crown reduction does not seem to directly lead to such a result. The approach uses the fact that in polynomial time we can:

1. either find a matching of size k in the complement $\overline{G}$ of G;
2. or find a crown decomposition in $\overline{G}$;
3. or conclude that the graph has at most $3k$ vertices.

As previously, if we are in case 1 then we have a coloring where we save at least k weights, so a weight at least k in total and the answer is yes. So the only case to deal with is the crown decomposition (C, H, B) in $\overline{G}$ where C is an independent set, C has no neighbor in B, and there is a matching saturating H between H and C.

Each vertex of C must receive a different color, and none of these colors can be used for vertices in B. In the unweighted case, each vertex of H can be assigned to a color used for coloring C (his mate in the matching), which is clearly the best we can do. However, here, we may save a larger weight by grouping vertices of H in another color (not used for C), for example if all the vertices in H have a large weight. By matching technique, we can optimally color $C \cup H$, but the extra colors may be used in B so the problem in B is not independent from our coloring of $C \cup H$.

4 Saving colors from non trivial colorings

4.1 Coloring with $f(m) - k$ colors

We now concentrate on saving colors with respect to an (improved) upper bound on the chromatic number. As explained in the introduction, we consider here the problem of saving k colors with respect to the 'non trivial coloring bound', formalized as follows - where $f(m) = \left\lfloor \frac{1}{2} + \sqrt{2m + \frac{1}{4}} \right\rfloor$.

Problem: *Saving non trivial colors*

- Input: A graph $G = (V, E)$ on n edges and m vertices, an integer k.
- Parameter: k
- Question: is $\chi(G) \leq f(m) - k$?

We first show that the problem is FPT and provide a kernel with $O(k^2)$ vertices and edges for it. The idea of the FPT algorithm is to reduce the problem to the one of saving k colors from the trivial coloring with n colors. Note that the bounds n and $f(m)$ become equal (only) when the graph is complete. The distance between the bounds gets bigger while the density of the graph decreases. Even for graphs of density 1/2, we have roughly $n^2/4$ edges, so $f(m)$ is roughly $n/\sqrt{2}$, much lower than n. So for the reduction we need to reach a graph which is almost complete; this is achievable by iteratively removing vertices of 'low' degree.

Theorem 3. *Saving non trivial colors is FPT. It admits a kernel with $O(k^2)$ vertices and edges.*

Proof. Let (G, k) be an instance of the problem. Let $b = f(m) - k$.

It is easy to see that if a vertex v of a graph H has degree $d(v) < c$, then H is c-colorable if and only if $H \setminus \{v\}$ is c-colorable. Then starting from (G, k) we apply the following reduction rule:

Rule 1: while there exists a vertex of degree at most $b - 1$, remove it from G.

Note that we consider that b is fixed: $b = f(m) - k$ where m is the number of edges of the initial graph, we do not update b when the graph is reduced. After the application of Rule 1, we get a graph $G' = (V', E')$ with n' vertices and m' edges, and every vertex of degree at least $b = f(m) - k \geq f(m') - k$. Clearly, G' is b-colorable iff G is b-colorable. Note that if $f(m') \leq 2k$ then the problem is trivially solvable in FPT-time, so assume now that $f(m') \geq 2k + 1$. Summing up the degrees, we get $2m' = \sum_{v \in V'} d_{G'}(v) \geq n'(f(m') - k)$. Then, since $\sqrt{2m'} \leq f(m') + 1/2$:

$$
\begin{aligned}
n' - f(m') &\leq \frac{(f(m') + 1/2)^2 - f(m')(f(m') - k)}{f(m') - k} \\
&= \frac{(k+1)(f(m') - k) + (k + 1/2)^2}{f(m') - k} \leq k + 1 + \frac{(k+1)^2}{f(m') - k} \leq 2(k+1)
\end{aligned}
$$

where the last inequality uses $f(m') \geq 2k + 1$.

So, $b = f(m) - k \geq f(m') - k \geq n' - 3k - 2$. If $b \geq n'$ the answer is trivially yes (G' is b-colorable, and so is G), otherwise we solve the problem in FPT-time using the FPT algorithm of [3] for deciding whether G' is $b = n' - k'$ colorable or not, where $k' = n' - b \leq 3k + 2$ (and $k' \geq 0$).

The previous arguments can be easily turned into a kernelization algorithm:

- We first apply Rule 1, and get the graph G'. (G, k) is a yes-instance iff (G', k_1) with $k_1 = k + f(m') - f(m) \leq k$ is a yes-instance.
- If $f(m') \leq 2k$, we have $m' = O(k^2)$ and $n' \leq 2m' = O(k^2)$ (since isolated vertices can be removed), so we have the claimed quadratic kernel.
- Otherwise we have $f(m') - k_1 = n' - k'$ for $k' = O(k)$. Applying the kernelization by Chor et al. [3], either we find an answer, or we have a graph (G'', k'') where G'' has $O(k') = O(k)$ vertices - hence $O(k^2)$ edges - such that G'' has an $n'' - k''$ coloring iff G' has a $n' - k'$ coloring, hence iff G has a $f(m) - k$ coloring. $n'' - k'' = f(m'') - k_2$ for some $k_2 \leq k''$, and the result follows. □

The kernelization algorithm produces a graph with $O(k^2)$ vertices and edges. While being of the same order in terms of size of the graph $O(m + n)$, the size of the kernel in terms of number of vertices is significantly bigger than the one obtained by [3] with the problem of coloring a graph with $n - k$ colors. However, we show that finding a smaller kernel for *Saving non trivial colors* is quite improbable, since this would contradict the ETH.

Theorem 4. *Saving non trivial colors does not admit a kernel with $o(k^2)$ vertices under ETH.*

Proof. Min coloring is not solvable in subexponential time under ETH even in graphs with a linear number of edges[2]. More precisely, under ETH there exist two constant $\epsilon > 0$ and d such that min coloring is not solvable in time $O(2^{\epsilon n})$ in graphs where $m \leq dn$.

Suppose that there is for *Saving non trivial colors* a kernelization algorithm that, given an instance (G, k), produces an equivalent instance (G', k') where the number of vertices in G' is $n' = o(k^2)$. Take a graph G with $m \leq dn$ edges. Let $c \leq n$, we want to determine whether G is c-colorable or not. Let $k = f(m) - c$. If $k \leq 0$ we can easily find a c-coloring of G in polynomial time. Otherwise, apply the kernelization algorithm to get in polynomial time (G', k') where G' is $(f(m') - k')$-colorable iff G is c-colorable, with $n' = o(k^2)$. $k \leq f(m) = O(\sqrt{m}) = O(\sqrt{n})$ (since $m \leq dn$), so $n' = o(k^2) = o(n)$. We can optimally color G' in time $2^{O(n')} = 2^{o(n)}$, thus determining in time $2^{o(n)}$ if G is c-colorable or not. Then we can determine the chromatic number of G in subexpontential time, thus contradicting ETH. □

[2] A classical reduction from 3-SAT produces a graph with $\Theta(var + clauses)$ vertices and edges, where *var* and *clauses* are the number of variables and clauses, see for instance http://cgi.csc.liv.ac.uk/~igor/COMP309/3CP.pdf.

To conclude this section, let us briefly mention other bounds on the chromatic number. A well known upper bound is $\chi(G) \leq \Delta + 1$ where Δ is the maximum degree of vertices in the graph. However, 3-coloring is NP-complete in graphs of maximum degree 4 [9], meaning that saving k colors with respect to the bound $\Delta+1$ is not in XP if $P \neq NP$. The same occurs for a refinement of this bound, namely $\chi(G) \leq \max\{\min\{d(v_i)+1, i\}, i = 1, \cdots, n\}$. Dealing with the lower bound $\omega(G)$ (size of the maximum clique), the NP-completeness of 3-coloring [9] also shows that determining whether a graph is $\omega(G)+k$ colorable is not in XP if $P \neq NP$ (even if a maximum clique is given).

4.2 Coloring with weight $\sum_{i=1}^{f(m)} w(v_i) - k$

We now generalize *Saving non trivial colors* to Max Coloring. Since any non trivial coloring uses at most $f(m) = \left\lfloor \frac{1}{2} + \sqrt{2m + \frac{1}{4}} \right\rfloor$ colors, there always exists a coloring of weight at most $\sum_{i=1}^{f(m)} w(v_i)$ (recall that v_i are in non-increasing order of weight). So we consider the following problem.

Problem *Saving non trivial weight.*

- Input: a vertex-weighted graph (G, w), and an integer k.
- Parameter: k
- Question: does there exist a coloring of G of weight at most $\sum_{i=1}^{f(m)} w(v_i) - k$?

In this problem, we can save weight either by reducing the number of colors (with respect to $f(m)$), or by producing colors with "small" weights, for instance by grouping together vertices of large weights. Note that an optimal solution for max coloring may use a number of colors much larger than the chromatic number, such as $\Theta(\log n)$ colors in a tree (see [5] for general bounds).

A first idea would be to use Rule 1 devised for the unweighted case (removing vertices of low degree), but this is no longer sound, because of this possibility to save on color weights. Even if there is an isolated vertex of large weight, removing it does not produce an equivalent instance (putting it with another vertex of large weight allows to save something). This case of isolated vertices can be handled (by properly modifying some weight in the remaining instance) but vertices of large weights cannot be removed even if they have small degree.

Then, vertices of large weight behave differently than vertices of small weight. Thus we decompose the vertex set according to the weights of the vertices, and this distinction is crucial in the solution of the problem. To make this distinction, we define $w^* = w(v_{f(m)})$: this is the smallest weight considered in the bound $\sum_{i=1}^{f(m)} w(v_i)$. We define $V_>$ (resp. $V_\leq$) as the set of vertices of weight greater than w^* (resp. at most w^*), and $n_> = |V_>|$, $n_\leq = |V_\leq|$.

The idea is to show that, unless a particular case occurs where we can safely answer YES, we can reduce the instance to a very particular case where the input graph is made of two cliques, plus $O(k)$ vertices (Lemma 4). Then, in XP-time we can further reduce the graph to two cliques, instance on which we can solve

the Max Coloring problem in polynomial time using Lemma 1 (Theorem 5). It is worth noticing that the only 'XP-time' step consists of dealing with the $O(k)$ extra-vertices.

We first consider two reduction rules.

Rule 2. While there exists in $V_{\leq}$ a vertex of degree at most $f(m)-k-1$, remove it.

The following Lemma states that this rule is sound.

Lemma 2. *Let (G, w, k) be an instance of Saving non trivial weight, v_ℓ a vertex in $V_{\leq}$ of degree at most $f(m) - k - 1$, and G' the graph obtained from G by removing v_ℓ[3]. Let $k' = \sum_{i=1}^{f(m')} w(v'_i) + k - \sum_{i=1}^{f(m)} w(v_i)$. Then (G, w, k) is a YES-instance iff (G', w, k') is a YES-instance.*

Proof. Clearly if (G, w, k) is a YES-instance, so is (G', w, k'): a coloring of G gives a coloring of at most the same weight in G'.

Conversely, suppose that (G', w, k') is a YES-instance, consider a non trivial coloring $\mathcal{C}' = (S'_1, \cdots, S'_p)$ of G' of weight at most $\sum_{i=1}^{f(m')} w(v'_i) - k' = \sum_{i=1}^{f(m)} w(v_i) - k$, where colors are ordered in non increasing weight. Since v_ℓ has degree at most $f(m) - k - 1$, we can add v_ℓ to one of the first $f(m) - k$ colors of $\mathcal{C}'$ to produce a non trivial coloring $\mathcal{C}$ of G, say color S'_j with $j \leq f(m) - k$.

If $w(v_\ell) \leq w(S'_j)$ then the weight of $\mathcal{C}$ is the same as the weight of $\mathcal{C}'$, so (G, w, k) is a YES-instance.

Otherwise, since $v_\ell \in V_{\leq}$, $w^* \geq w(v_\ell) > w(S'_j)$. But this means that in $\mathcal{C}$ each color $S_{j'}$ with $j' > j$ has weight (strictly) smaller than w^*. Since $j \leq f(m) - k$, we get:

$$w(\mathcal{C}) \leq \sum_{i=1}^{f(m)-k} w(v_i) + \sum_{i=f(m)-k+1}^{f(m)} (w^* - 1) \leq \sum_{i=1}^{f(m)} w(v_i) - k$$

So (G, w, k) is a YES-instance. □

Note that after this step if $k' \leq 0$ we can safely answer YES.

Rule 3. Compute a maximal matching M in the complement of $G[V_>]$. If M has size at least k, answer YES.

Lemma 3. *Rule 3 is sound.*

Proof. Suppose that M has size at least k. Color the vertices of $V_>$ with a non trivial coloring with $t \leq |V_>| - k$ colors. Then add the other vertices greedily. This produces a non trivial coloring $\mathcal{C}$, hence with at most $f(m)$ colors. At least k colors receive a weight at most w^* instead of their 'original' weight in the bound which is greater, so at least k is saved with respect to the bound. More formally:

[3] We use v'_i to denote vertices of G' (so $v'_i = v_i$ for $i < \ell$ and $v'_i = v_{i+1}$ for $i \geq \ell$).

– The first t colors of $\mathcal{C}$ have a total weight at most $\sum_{i=1}^{t} w(v_i)$;
– All other colors of $\mathcal{C}$ have weight at most w^*, and $\mathcal{C}$ has at most $f(m)$ colors.

So:

$$\begin{aligned} w(\mathcal{C}) &\leq \sum_{i=1}^{t} w(v_i) + (n_> - t)w^* + (f(m) - n_>)w^* \\ &\leq \sum_{i=1}^{t} w(v_i) + \sum_{i=t+1}^{n_>} (w(v_i) - 1) + (f(m) - n_>)w^* = \sum_{i=1}^{f(m)} w(v_i) - (n_> - t) \\ &\leq \sum_{i=1}^{f(m)} w(v_i) - k \end{aligned}$$

□

As a consequence, once Rules 2 and 3 have been applied, we get an equivalent instance where:

– Condition 1: every vertex in $V_\leq$ has degree at least $f(m) - k$;
– Condition 2: $V_>$ is made of a clique $C_>$ on at least $n_> - 2k$ vertices, plus at most $2k$ vertices.

Now, we consider Algorithm 1.

Algorithm 1
STEP 0: Compute a coloring $\mathcal{C} = (S_1, S_2, \dots)$ of G;
STEP 1: While there exists a vertex v in S_j which has no neighbor in some color S_i with $i < j$, color v with S_i;
STEP 2: If it is possible to empty a color S_i, do it and go to STEP 1.
STEP 3: Take any pair of colors of size 2, any pair of colors of size 1. If it is possible to re-color these 6 vertices with 3 colors, do it and go to STEP 1.

Lemma 4. *Algorithm 1 runs in polynomial time. Moreover, let (G, w, k) be an instance satisfying conditions 1 and 2. If G has more than $(k+1)^4$ edges, then Algorithm 1:*

– *Either produces a coloring with at most $f(m) - k$ colors;*
– *Or allows to find a decomposition of V into (C_1, C_2, T) where C_1 and C_2 are two cliques, and $|T| = O(k)$.*

Proof. We first explain why Algorithm 1 takes polynomial time:

– Step 1 is clearly polynomial, since each vertex changes color at most n times.
– For the second step, S_i being an independent set, emptying S_i is *not* possible if and only if there is a vertex in S_i adjacent to all other colors, so this step is also polynomial.

– Step 3 is clearly polynomial, since it involves 4 colors and 6 vertices.
– Since the number of colors reduces in steps 2 and 3, the steps are executed $O(n)$ times.

Now we deal with the claimed property. Let $\mathcal{C} = (S_1, \ldots, S_p)$ be the coloring output by the algorithm.

Thanks to Step 1, $\mathcal{C}$ is non trivial, hence $p \leq f(m)$. If $p \leq f(m) - k$, then the property of the Lemma holds (case 1). So we consider that $p > f(m) - k$, meaning $f(m) \leq p + k - 1$. Since $f(m) + 1/2 \geq \sqrt{2m}$, we get:

$$m < (p+k)^2/2 \tag{1}$$

Note that since $m \geq (k+1)^4$, we have $p \geq k^2 + k$, so

$$k \leq \sqrt{p} \tag{2}$$

Let s be the number of colors of size at least 2. Assume[4] that these are the first s colors in $\mathcal{C}$. We first show that almost all colors have size 1, as stated in the following fact.

Fact 1. The number of colors of size at least 2 is $s = o(p)$.

Proof of Fact 1. Due to Step 1, every vertex in S_j has at least $j-1$ neighbors, so we have:

$$m \geq \sum_{j=1}^{p} (j-1)|S_j| \geq p(p-1)/2 + s(s-1)/2 \tag{3}$$

As a consequence of inequalities (1) and (3), we get:

$$\frac{(s-1)^2}{2} \leq \frac{s(s-1)}{2} \leq \frac{(p+k)^2 - (p-1)^2}{2} = \frac{p(2k+2) + k^2 - 1}{2}$$

Using (2) we get $s = O(\sqrt{kp}) = o(p)$. □

Now, thanks to Step 2, in each color S_i there is a vertex which is adjacent to all other colors, let z_i be such a vertex. Note that for $i > s$ z_i is the unique vertex of S_i. Denote $Z_1 = \{z_i, i \leq s\}$, $Z_2 = \{z_i, i > s\}$ and $Z = Z_1 \cup Z_2$. We show that almost all vertices of 'small' weights are in Z. More precisely, let Q be the set of vertices outside Z of 'small' weight: $Q = V_{\leq} \setminus Z$ and $q = |Q|$.

Fact 2. The number of vertices in Q is $q \leq 4k + 10$.

Proof of Fact 2. Since z_i and z_j are adjacent for all $i \in \{1, \ldots, p\}$ and all $j > s$ $(j \neq i)$, we have at least $(p-s)(p-s-1)/2 + s(p-s)$ edges (z_i, z_j) in $G[Z]$.

[4] Thanks to Step 2, each color of size 1 is adjacent to all other colors, so we can reorder colors if needed to put colors of size 1 at the end while preserving the fact that each vertex in S_j is adjacent to each color S_i, $i < j$.

Each of the q vertices in Q have degree at least $f(m) - k \geq p - k$, thanks to Condition 1. So we have:

$$m \geq \frac{q(p-k)}{2} + \frac{(p-s)(p-s-1) + 2s(p-s)}{2} = \frac{q(p-k) + (p^2 - s^2 + s - p)}{2}$$
$$\geq \frac{q(p-k) + p(p-1) - s(s-1)}{2}.$$

Using (1) and (3) we have $s(s-1) \leq (p+k)^2 - p(p-1)$. Using (1) for bounding m, we obtain:

$$q \leq \frac{2(p+k)^2 - 2p(p-1)}{p-k} = \frac{p(4k+2) + 2k^2}{p-k} = 4k + 2 + \frac{6k^2 + 2k}{p-k}$$

Since as noted before $p \geq k^2 + k$, we get $q \leq 4k + 10$. □

Now, thanks to condition 2, the set of vertices of large weight $V_>$ is made of a clique $C_>$ plus $O(k)$ vertices. Then, besides Z and $C_>$, G contains $O(k)$ vertices: $O(k)$ vertices from $V_\leq$ (the set Q), and the remaining $O(k)$ vertices from $V_>$.

The third and last step of the proof is to show that Z is almost a clique, and this is where Step 3 of the algorithm intervenes: we show now that Z is almost a clique otherwise we could recolor 2 colors of size 2 and 2 colors of size 1 with 3 colors in total.

Fact 3. Z is made of a clique plus $O(k)$ vertices.

Proof of fact 3. $Z = Z_1 \cup Z_2$; we already know that Z_2 is a clique, and that every vertex in Z_2 is adjacent to every vertex in Z_1. So we only have to show that Z_1 is almost a clique (a clique plus $O(k)$ vertices).

Consider the $O(k)$ vertices that are outside $Z \cup C_>$. These vertices are contained in $s_0 = O(k)$ colors. Remove the vertices in these colors. Then in the remaining graph we have

- $p - s$ colors of size 1 (colors $\{z_j\}$, for $j > s$);
- $s - s_0$ colors of size exactly two, made of one vertex from $C_>$ and one vertex from Z_1.

Among the $s - s_0$ colors of size 2, suppose that in two of these colors $S_i = \{z_i, w_i\}, S_{i'} = \{z_{i'}, w_{i'}\}$ $(i, i' \leq s)$ z_i and $z_{i'}$ are not adjacent.

If there exists $j, j' > s$ such that $\{z_j, w_i\}, \{z_{j'}, w_{i'}\} \notin E$, then we would be able to recolor these 4 colors in 3 colors (z_i and $z_{i'}$ together, w_i and z_j, and $w_{i'}$ and $z_{j'}$), which is impossible thanks to Step 3. It means that for a pair $z_i, z_{i'}$ of non adjacent vertices we get at least $(p-s)/2$ edges from Z_2 to w_i or $w_{i'}$.

Take a maximal set of (independent) such pairs z_i, z'_i, and denote by q' the number of these pairs. Z_2 is a clique, and each vertex in Z_1 is adjacent to all $p-1$ other colors, so this already gives at least $p(p-1)/2$ edges. Then we must have:

$$\frac{q'(p-s)}{2} + \frac{p(p-1)}{2} \leq m \leq \frac{(p+k)^2}{2}$$

We get $q' \leq \frac{p(2k+1)+k^2}{p-s}$. As noted before, $s = O(\sqrt{kp})$ and $p \geq k^2$, so $q' = O(k)$. □

Then, from the previous facts we immediately get that our graph has:

- A clique C_Z which is a subset of Z
- A clique $C_>$ which is a subset of $V_>$
- $O(k)$ other vertices. □

Now, we are ready to show that the *Saving non trivial weight* is in XP.

Theorem 5. *Saving non trivial weight is in XP.*

Proof. We apply rules 2 and 3. Either we answer YES, or we have an equivalent instance satisfying conditions 1 and 2.

First consider the case where $m < (k+1)^4$. In this case we can solve the problem in FPT time. Indeed, if there are at least two isolated vertices y_1, y_2 in the graph with $w(y_1) \geq w(y_2)$, they can be replaced by one vertex y with weight $w(y_1)$: optimal colorings have obviously the same weight before and after this replacement, so we just have to adjust (reduce) the parameter if $w(y_2) \geq w^*$, so that the value $\sum_{i=1}^{f(m)} w(v_i) - k$ remains the same.

Hence, we can assume that there is at most one isolated vertex in G, and $n \leq 2m + 1 \leq 2(k+1)^4 + 1$, so the graph is already a kernel.

Now, suppose that $m \geq (k+1)^4$. Then we apply Algorithm 1. If this gives a coloring using at most $f(m) - k$ colors, then this coloring has a weight at most $\sum_{i=1}^{f(m)-k} w(v_i) \leq \sum_{i=1}^{f(m)} w(v_i) - k$ so we answer YES.

Otherwise, using Lemma 4 we get a partition of the graph into 2 cliques C_1 and C_2 and a set T of size $|T| = O(k)$.

Intuitively, we first consider all possible ways to partition vertices of $C_2 \cup T$ into color classes, and then in each case we 'match' them in an optimal way with vertices of C_1 using Lemma 1. More precisely, vertices of $T \cup C_2$ will be colored with between $|C_2|$ and $|T| + |C_2|$ colors: for vertices of T, we consider the $O((|C_2| + |T|)^{|T|}) = n^{O(k)}$ possibilities for vertices in T, by putting each vertex in one of the possible colors. In each of these cases, we contract the vertices in the same color into one vertex (with weight the maximum weight of merged vertices), and produce in each case a clique C_2'. Then we only have to solve the problem on the obtained graph made of 2 cliques C_1 and C_2', in polynomial time, using Lemma 1. The global running time is $n^{O(k)}$, so the result follows. □

Theorem 5 leaves the question of fixed-parameter tractability open: *is Saving non trivial weight FPT?* In the light of our proof, it suffices to show that the problem is (or is not) FPT on the class of graphs that are the union of two cliques C_1, C_2 and a set of T of $O(k)$ vertices.

Note then none of the ideas leading to Theorems 1 and 2 seem to work here. In Theorem 1 we consider all possible colorings of T and then polynomially solve each case; here, by doing this we get a graph made of 3 cliques, where coloring problems are already hard. In Theorem 2, we use a dominance condition which,

given a set of vertices in the clique with the same neighborhood, allows to remove all but only $O(k)$ vertices (of largest weights) in this set. Here, if we consider a set of vertices with the same neighborhood in T, there is no more dominance condition because the neighborhood in $C_1 \cup C_2$ also matters.

5 Max Coloring under standard parameterization

As mentioned in introduction, Max Coloring is hard to solve in classes of graphs where the usual coloring problem is polynomial. In particular:

- It is NP-hard in bipartite and interval graphs (hence in chordal and perfect graphs) [5, 8];
- It is not solvable in polynomial time in trees under ETH [1].

Note that Max Coloring is in APX in perfect graphs [17], and admits an approximation scheme in interval graphs [16].

The question of the parameterized complexity of this problem under standard parameterization occurs for Max Coloring in these graph classes.

Problem *Parameterized max coloring*

- Input: a vertex-weighted graph (G, w), an integer B.
- Parameter: B.
- Question: does there exist a coloring with weight at most B?

Dealing with bipartite graphs, the following result settles the question.

Theorem 6. *[5] In bipartite graphs, it is NP-hard to distinguish between instances where there exists a coloring of weight at most 7 from instances where each coloring has weight at least 8.*

As an immediate corollary, we get that *Parameterized max coloring* is not in XP in bipartite graphs assuming $P \neq NP$.

Let us now concentrate on trees/interval graphs/chordal graphs. We show that *Parameterized max coloring* is FPT in these classes of graphs, using dynamic programming on tree decompositions, based on the the following result.

Theorem 7. *[12] In graphs of maximum treewidth k, the problem of finding a lightest coloring having at most r colors is solvable in $O(n^{r+1})$.*

Here, the O-notation hides the dependency in k, which is $r^{O(k)}$. Actually, a first step in the proof of this theorem is to prove a result which can be stated as follows:

Lemma 5. *[12] Given any sets of r integers $a_1 \geq a_2 \geq \cdots \geq a_r \geq 0$, determining whether there exists a coloring with r colors $S_1, \ldots, S_r$ such that $w(S_i) \leq a_i$ can be done in time $r^{O(k)}n^{O(1)}$ in graphs of treewidth at most k.*

Theorem 7 follows from Lemma 5 by performing an exhaustive search over all possible tuples $(a_1, \ldots, a_r)$ (thus the term in n^r in the theorem).

We use instead Lemma 5, from which we can derive that *Parameterized max coloring* is FPT in chordal graphs.

Theorem 8. *Parameterized max coloring is FPT in chordal graphs.*

Proof. Given a chordal graph G, let c be the size of a maximum clique (computable in polynomial time). If $c > B$ then we can safely answer NO. Otherwise, $c \leq B$. G being chordal, its treewidth is $c - 1 \leq B - 1$.

On the other hand, there exists at most $(B+1)^B$ ways to choose B weights $a_1 \geq a_2 \geq \cdots \geq a_B \geq 0$ whose sum is at most B, hence $(B+1)^B$ tuples of color weights to check in order to determine if there exists a coloring of weight at most B. Using Lemma 5, we conclude that the problem is solved FPT-time $2^{O(B \log B)} n^{O(1)}$. □

To conclude this section, let us mention that another parameterization is very natural: the maximum weight of vertices. When parameterized by the maximum weight of vertices w_{max}, Max Coloring is:

- not in XP in bipartite graphs (from the fact that in Theorem 6 weights at most 4 are used):
- FPT in trees, from Theorem 8 since in trees the optimum value is at most twice the maximum weight (take a coloring with two colors).

However, what happens in interval (or chordal) graphs? As far as we know, the (standard) complexity of Max Coloring in interval graphs with bounded weights is still an open question. Dealing with parameterized complexity: *is Max Coloring FPT in interval graphs when parameterized by the maximum weight? Is it in XP?*

6 Conclusion

Besides the open questions mentioned above, and besides other coloring problems where a similar approach could be considered, analogous questions should be also interesting for other kinds of problems. Let us mention that the same arguments as for coloring show that the problem of *saving k bins* in bin packing is FPT, as well as other partitioning problems. Indeed, consider a problem where we have to partition a set X of size n into a minimum number of subsets, where only some subsets are admissible - stable sets in coloring, elements with sum of lengths at most 1 in bin packing - (see [13] for similar questions dealing with approximation). If any subset of an admissible subset is admissible, as it is the case in the two previous problems, then the question of determining whether there exists a partition with at most $n - k$ sets or not is FPT. Indeed, one computes a maximal set of admissible pairs. If there are at least k pairs then we are done (singletons are admissible, otherwise the problem has no feasible solution), otherwise the remaining $n - O(k)$ elements must be each in a separate

subset. Then using brute force on the $O(k)$ elements in the pairs and solving maximum matchings, we get an FPT algorithm. Improving upon better bounds could be also interesting for these problems.

References

1. Júlio Araújo, Nicolas Nisse, and Stéphane Pérennes. Weighted coloring in trees. *SIAM J. Discrete Math.*, 28(4):2029–2041, 2014.
2. Leizhen Cai. Parameterized complexity of vertex colouring. *Discrete Applied Mathematics*, 127(3):415–429, 2003.
3. Benny Chor, Mike Fellows, and David W. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. In Juraj Hromkovic, Manfred Nagl, and Bernhard Westfechtel, editors, *WG 2004*, volume 3353 of *Lecture Notes in Computer Science*, pages 257–269. Springer, 2004.
4. Dominique de Werra, Marc Demange, Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Weighted coloring on planar, bipartite and split graphs: Complexity and approximation. *Discrete Applied Mathematics*, 157(4):819–832, 2009.
5. Marc Demange, Dominique de Werra, Jérôme Monnot, and Vangelis Th. Paschos. Time slot scheduling of compatible jobs. *J. Scheduling*, 10(2):111–127, 2007.
6. Rodney G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer Publishing Company, Incorporated, 2012.
7. Bruno Escoffier. Saving colors and max coloring: Some fixed-parameter tractability results. In Pinar Heggernes, editor, *WG 2016*, volume 9941 of *Lecture Notes in Computer Science*, pages 50–61. Springer, 2016.
8. Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Weighted coloring: further complexity and approximability results. *Inf. Process. Lett.*, 97(3):98–103, 2006.
9. Michael R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified NP-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976.
10. Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57)*. North-Holland Publishing Co., Amsterdam, The Netherlands, 2004.
11. Martin Grötschel, Lászlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988.
12. D. J. Guan and Xuding Zhu. A coloring problem for weighted graphs. *Inf. Process. Lett.*, 61(2):77–81, 1997.
13. Refael Hassin and Jérôme Monnot. The maximum saving partition problem. *Oper. Res. Lett.*, 33(3):242–248, 2005.
14. Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.
15. Bart M. P. Jansen and Stefan Kratsch. Data reduction for graph coloring problems. *Inf. Comput.*, 231:70–88, 2013.
16. Tim Nonner. Clique clustering yields a PTAS for max-coloring interval graphs. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP 2011, Proceedings, Part I*, volume 6755 of *Lecture Notes in Computer Science*, pages 183–194. Springer, 2011.

17. Sriram V. Pemmaraju and Rajiv Raman. Approximation algorithms for the max-coloring problem. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, pages 1064–1075. Springer, 2005.