Weighted Coloring in Trees

Julio Araújo, Nicolas Nisse, Stéphane Pérennes

Weighted Coloring in Trees*

Julio Araujo[1,2,3], Nicolas Nisse[2,3], and Stéphane Pérennes[3]

1 **ParGO, Universidade Federal do Ceará, Fortaleza, Brazil**
2 **Inria, France**
3 **Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, Sophia Antipolis, France**

Abstract

A proper coloring of a graph is a partition of its vertex set into stable sets, where each part corresponds to a *color*. For a vertex-weighted graph, the *weight of a color* is the maximum weight of its vertices. The *weight of a coloring* is the sum of the weights of its colors. Guan and Zhu defined the *weighted chromatic number* of a vertex-weighted graph G as the smallest weight of a proper coloring of G (1997). If vertices of a graph have weight 1, its weighted chromatic number coincides with its chromatic number. Thus, the problem of computing the weighted chromatic number, a.k.a. Max Coloring Problem, is NP-hard in general graphs. It remains NP-hard in some graph classes as bipartite graphs. Approximation algorithms have been designed in several graph classes, in particular, there exists a PTAS for trees. Surprisingly, the time-complexity of computing this parameter in trees is still open.

The Exponential Time Hypothesis (ETH) states that 3-SAT cannot be solved in sub-exponential time. We show that, assuming ETH, the best algorithm to compute the weighted chromatic number of n-node trees has time-complexity $n^{\Theta(\log n)}$. Our result mainly relies on proving that, when computing an optimal proper weighted coloring of a graph G, it is hard to combine colorings of its connected components.

1998 ACM Subject Classification G.1.6 Optimization

Keywords and phrases Weighted Coloring; Max Coloring; Exponential Time Hypothesis; 3-SAT

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Given a loop-less graph $G = (V, E)$, a *(proper) k-coloring* of G is a surjective function $c : V \to \{1, \ldots, k\}$ that assigns to each vertex $v \in V$ a *color* $c(v) \in \{1, \ldots, k\}$, such that, for any $\{u, v\} \in E$, $c(u) \neq c(v)$. Equivalently, a k-coloring of G is a partition $c = (S_1, \ldots, S_k)$ of V such that, for any $1 \leq i \leq k$, S_i is a non-empty independent set of vertices that have the same color i. One of the most studied problems in Graph Theory consists in minimizing the number of colors of a proper coloring of a graph. Namely, Graph Coloring aims at computing the *chromatic number* of a graph G, denoted by $\chi(G)$, which is the minimum k for which G has a k-coloring. This is one of the Karp's NP-hard problems [8].

In [6], Guan and Zhu generalized Graph Coloring to vertex-weighted graphs. A *(vertex) weighted graph* (G, w) consists of a loop-less graph $G = (V, E)$ and a weight function $w : V \to \mathbb{R}_+$ over the vertices of G. Given a k-coloring $c = (S_1, \ldots, S_k)$ of a weighted graph (G, w), the *weight of color i* $(1 \leq i \leq k)$ is defined by $w(i) = \max_{v \in S_i} w(v)$. The *weight of coloring* c is $w(c) = \sum_{i=1}^{k} w(i)$. The *weighted chromatic number* of (G, w), denoted by

* This work was partly funded by the ANR projects AGAPE and GRATEL, and promoted by the Inria/FUNCAP project ALERTE and the Inria associate-team AlDyNet and CNPq–Brazil (contract PDE 202049/2012-4).

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 1–12

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

$\chi_w(G)$, is the minimum weight of a proper coloring of (G, w). The Weighted Coloring Problem (also known as Max-coloring [15, 12, 13, 14, 11]) takes a weighted graph (G, w) as input and asks whether $\chi_w(G)$ is bounded [6].

Observe that if the weight of each of the vertices of a graph (G, w) is equal to one, then the weight of a coloring is the number of its colors and thus, $\chi_w(G) = \chi(G)$. Therefore, Weighted Coloring generalizes Graph Coloring to weighted graphs, and, as a consequence, this problem is NP-hard in general graphs. Moreover, Weighted Coloring has been shown NP-hard in bipartite graphs [3], where Graph Coloring is trivial. In the last years, the Weighted Coloring Problem has been addressed several times, however the complexity of this problem is surprisingly still unknown in the class of trees.

Here, we show that, if 3-SAT cannot be solved in sub-exponential time (Exponential Time Hypothesis), then Weighted Coloring in trees is not in P.

Related work. Weighted Coloring has been shown to be NP-hard in the classes of split graphs, interval graphs, triangle-free planar graphs with bounded degree, and bipartite graphs [3, 14, 2, 5, 15]. On the other hand, the weighted chromatic number of cographs and of some subclasses of bipartite graphs can be found in polynomial-time [3, 2]. Constant-factor approximation algorithms have been designed for various graph classes such as interval graphs, perfect graphs, etc. [14, 11, 12, 13, 4]. In particular, it is known that Weighted Coloring can be approximated by a factor $\frac{8}{7}$ in bipartite graphs and cannot be approximated by a factor $\frac{8}{7} - \epsilon$ for any $\epsilon > 0$ in this graph class unless $P = NP$ [13].

Guan and Zhu showed that, given a fixed parameter $r \in \mathbb{N}$, the minimum weight of a coloring using at most r colors can be computed in polynomial-time[1] in the class of bounded treewidth graphs (a.k.a. partial k-trees) [6]. They left open the question of the time-complexity of the Weighted Coloring Problem in this class (partial k-trees) and, in particular, in trees. In [13], a sub-exponential algorithm and a polynomial-time approximation scheme to compute the weighted chromatic number of trees are presented. Later on, Escoffier et al. proposed a polynomial-time approximation scheme to compute the weighted chromatic number of bounded treewidth graphs [5]. Kavitha and Mestre recently presented polynomial-time algorithms for subclasses of trees [9]. They show that computing the weighted chromatic number can be done in linear time in the class of trees where nodes with degree at least three induce a stable set [9].

In the last years, many studies have been done on the Weighted Coloring Problem, however the complexity of this problem was still unknown on trees. Indeed, Weighted Coloring in trees has some intriguing properties: on the one hand, a reduction to another NP-hard problem was unlikely to exist due to the existence of a sub-exponential algorithm [13] (see also Section 2); on the other hand, all the classical methods to derive polynomial-time algorithms on trees failed [5, 9]. We provide here some explanation for these facts.

Our results. We show that, under the Exponential Time Hypothesis (ETH) (see Section 2), the best algorithm to compute the weighted chromatic number of trees has time-complexity $n^{\Theta(\log n)}$, where n is the number of vertices of the input tree. The existence of an algorithm that solves the Weighted Coloring Problem in time $n^{\Theta(\log n)}$ in bounded treewidth graphs follows easily from previous results. The difficulty is to prove that it is optimal under ETH. For this, we show that computing the weighted chromatic number of an n-node tree is as hard as deciding whether a 3-SAT formula with size $log^2 n$ can be satisfied, where the size of a formula is η if it has η variables and its number of clauses is a polynomial in η. So, our

[1] We emphasize that this algorithm is exponential in r

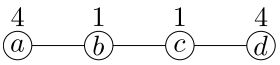

Figure 1 The unique optimal weighted coloring of P_4 uses strictly more than $\chi(P_4)$ colors.

reduction is rather technical, but we hope that it contains ideas that may be used in other contexts. Along the line of our reduction, one will discover another surprising aspect: the difficulty of the problem not only comes from the graph structure, but rather relies on the way weights are structured. This implies that choosing the right color for a node is hard. We indeed use non-binary constraint satisfaction formulae (i.e., constraint satisfaction formulae over positive integers) as main tool. Lastly, our reduction also proves that computing an optimal weighted coloring of a disconnected graph may be hard even if optimal colorings of each of its components can be done in polynomial-time.

Organization of the paper. The remainder of the paper is organized as follows. In Section 2, we formally state the main results of the paper: in Section 2.1, an $n^{\mathcal{O}(\log n)}$-time algorithm is derived from previous works, and in Section 2.2 we prove our main result assuming a technical reduction (Proposition 2). The remaining part of the paper is devoted to the proof of Proposition 2. In Section 3, we give the main ideas of its proof. Finally, in Section 4, we prove a technical result (Proposition 3) which allows us to prove Proposition 2.

2 Preliminaries

2.1 Sub-exponential algorithm

In this section, we show that there exists a sub-exponential algorithm to solve the Weighted Coloring Problem in the class of bounded treewidth graphs (including trees). This is an almost trivial consequence of previous works that mainly relies on the number of colors used by weighted colorings in these graphs.

There exist weighted graphs G for which any optimal weighted coloring uses strictly more than $\chi(G)$ colors: let us consider the 4-node path P_4 with $V(P_4) = \{a, b, c, d\}$, $w(a) = w(d) = 4$ and $w(b) = w(c) = 1$ (see Figure 1). Any coloring of P_4 with $2 = \chi(P_4)$ colors has weight 8, and the optimal coloring $\{\{a, d\}, \{b\}, \{c\}\}$ of P_4 has weight $\chi_w(P_4) = 6$ but uses 3 colors.

Luckily, the number of colors used by optimal weighted colorings can be bounded by $\mathcal{O}(\log n)$ in the class of bounded treewidth graphs with n nodes. Indeed, Guan and Zhu studied the number of colors used by an optimal weighted coloring [6]. More precisely, they proved that the maximum number of colors of an optimal weighted coloring of a weighted graph (G, w) is its first-fit chromatic number $\chi_{FF}(G)$ (a.k.a., *Grundy number*) [6]. This is tight since, for any graph G, there exists a weight function w such that an optimal weighted coloring of (G, w) uses $\chi_{FF}(G)$ colors. On the other hand, for any n-node graph G with tree-width at most k, $\chi_{FF}(G) = \mathcal{O}(k \log n)$ [10]. In particular, this implies that, for any n-node tree, there is an optimal weighted coloring using $\mathcal{O}(\log n)$ colors. Finally, in the class of bounded treewidth graphs and when the number $r \in \mathbb{N}$ of colors is fixed, there is an algorithm (using dynamic programming on the tree-decomposition) that computes the minimum weight of a coloring using at most r colors in time polynomial in $\mathcal{O}(n^r)$ where n is the number of vertices of the input graph [6].

By combining these results, the following proposition is straightforward:

▶ Proposition 1. There exists an algorithm that solves the Weighted Coloring Problem in time $n^{\mathcal{O}(\log n)}$ in the class of bounded treewidth graphs (including trees), where n is the number of vertices of the input graph.

2.2 Main Result

We now formalize our main result. Recall that an instance of the 3-SAT Problem is any Boolean formula $\Phi(v_1, \ldots, v_\eta)$ over the variables $v_1, \ldots, v_\eta$ in the conjunctive normal form (CNF) where each clause involves three variables. The *size* of Φ is η if it depends on η variables and its number of clauses is polynomial in η. The 3-SAT Problem asks whether there exists a truth assignment to the variables such that $\Phi(v_1, \ldots, v_\eta)$ is true. It is well known that the 3-SAT Problem is NP-complete [1]. A fundamental question is to know whether it can be solved in sub-exponential time. Note that, otherwise, $P \neq NP$.

▶ Conjecture 1. **Exponential Time Hypothesis (ETH)[7].**
3-SAT cannot be solved in time $2^{o(\eta)}$ where η is the size of the instance.

The main part of this paper is devoted to proving the following result.

▶ Proposition 2. For any Boolean formula Φ of size η, there exist a weighted tree (T, w) with $n = 2^{\mathcal{O}(\sqrt{\eta})}$ vertices and $M \in \mathbb{R}$ such that Φ is satisfiable if and only if $\chi_w(T) \leq M$. Moreover, (T, w) and M are computable in time polynomial in n.

Proposition 2 allows us to prove that there is no polynomial-time algorithm to solve the Weighted Coloring Problem in trees, unless ETH fails.

▶ **Theorem 1.** *If ETH is true, then the best algorithm to compute the weighted chromatic number of an n-node tree T has time-complexity $n^{\Theta(\log n)}$.*

Proof. The existence of such an algorithm directly follows from Proposition 1. For purpose of contradiction, let us assume that there exists an algorithm $\mathcal{A}$ that solves the Weighted Coloring Problem in time $n^{o(\log n)}$ in the class of trees, where n is the number of vertices of the input tree. Let Φ be any Boolean formula of size η. By Proposition 2, there exists a weighted tree (T, w) with $n = 2^{\mathcal{O}(\sqrt{\eta})} = 2^{o(\eta)}$ vertices and $M \in \mathbb{R}$ such that Φ is satisfiable if and only if $\chi_w(T) \leq M$. Consider the following algorithm to solve 3-SAT. For any Boolean formula Φ of size η, first compute (T, w) and M in time $2^{o(\eta)}$, then use Algorithm $\mathcal{A}$ to compute $\chi_w(T)$ in time $n^{o(\log n)} = 2^{o(\eta)}$. By definition, Φ is satisfiable if and only if $\chi_w(T) \leq M$. Therefore, the above algorithm solves the 3-SAT Problem in time $2^{o(\eta)}$ where η is the size of the instance, contradicting ETH. ◀

The remaining part of the paper is devoted to the proof of Proposition 2.

3 From boolean variables to integral variables

Proposition 2 establishes a link between the Weighted Coloring Problem and 3-SAT. Informally, to evaluate the time-complexity of the Weighted Coloring Problem, the ideal way would be to reduce any 3-SAT formula Φ to a weighted tree (T, w) such that (1) there is a correspondence between truth assignments of the variables of Φ and the optimal colorings of T, and (2) Φ is satisfiable if and only if $\chi_w(T)$ is at most some pre-defined value M (depending on Φ). To do such a reduction, we would like to proceed as follows: given a boolean formula Φ of size η, we build a weighted tree T such that any truth assignment of Φ for which Φ is satisfied, we have a coloring of T of bounded weight, where the weight of a color reflects the truth assignment of a variable. Hence, the desired weighted tree T

must be such that any optimal coloring of T uses η colors. However, proceeding that way, since the number of colors in an optimal weighted coloring of an n-node tree is at most $\mathcal{O}(\log n)$, T must have at least $n = 2^\eta$ nodes. Hence, a polynomial-time algorithm to solve the WEIGHTED COLORING Problem in T would only lead to an exponential-time algorithm for deciding whether Φ is satisfiable.

3.1 From 3-SAT to INT-SAT

To bypass the above problem, we will use an auxiliary formula. Intuitively, given a 3-SAT formula with η boolean variables, we will translate it into another logical formula with $\sqrt{\eta}$ *integral variables*. Using this new formula, we build a tree with $2^{\sqrt{\eta}}$ nodes, where the weights of the colors in coloring of bounded weight will correspond to the integral values of the variables. Note that our method is close to the *Split and List* method of [16]. More formally,

▶ **Definition 2.** Given a set of $n \times m$ boolean variables $(y^i_j)_{i<n,j<m}$, an *integral assignment* of these variables is a truth assignment such that, for any $0 \leq i < n$, at most one variable y^i_j, $0 \leq j < m$, receives value 1.

A boolean formula Φ with $n \times m$ boolean variables $(y^i_j)_{i<n,j<m}$ is *integrally satisfiable w.r.t.* $(y^i_j)_{i<n,j<m}$ if there is an integral assignment of its variables that satisfies Φ.

The *INT-SAT Problem* takes a formula Φ with variables $(y^i_j)_{i<n,j<m}$ as input and asks whether Φ is integrally satisfiable w.r.t. $(y^i_j)_{i<n,j<m}$.

In what follows, we widely use the fact that there is a one-to-one mapping between any integral assignment of a set of $n \times m$ boolean variables $(y^i_j)_{i<n,j<m}$ and the set of n-tuples $(x_1, \ldots, x_n)$ of integers in $\{0, \ldots, m\}$. Indeed, for any $i < n$, $x_i = j$ if and only if $y^i_j = 1$, and $x_i = 0$ if $y^i_j = 0$ for all $j < m$.

We now show that 3-SAT can be sub-exponentially reduced to INT-SAT. This is an important ingredient of the proof of Proposition 2. We also think this result has its own interest and could be used in other contexts.

▶ **Theorem 3.** *For any instance Φ of 3-SAT with size η, there is a Boolean formula Φ_{int} of size $n = 2^{\mathcal{O}(\sqrt{\eta})}$, with variables $(y^i_j)_{i<\sqrt{\eta},j<2^{\sqrt{\eta}}}$, s.t. Φ is satisfiable if and only if Φ_{int} is integrally satisfiable w.r.t. $(y^i_j)_{i,j}$. Φ_{int} can be computed in time $\mathcal{O}(n)$ and it is a CNF formula where all variables appear positively.*

Proof. Let $\Phi(u_1, \ldots, u_\eta)$ be an instance of 3-SAT of size $\eta = N^2$ (if $\eta \neq N^2$, we can add dummy variables). For any two integers $a < N$ and $b < 2^N$, let $bit(a, b)$ correspond to the a-th bit of the binary representation of b.

Let Φ_{int} be the formula obtained from Φ by replacing each literal u_{iN+j}, $0 \leq i < N$ and $0 \leq j < N$, by $\bigvee_{\{\ell | bit(j,\ell)=1,\ 0\leq\ell<2^N\}} v^i_\ell$. Then, each literal $\bar{u}_{iN+j}$, $0 \leq i < N$ and $0 \leq j < N$ is replaced by $\bigvee_{\{\ell | bit(j,\ell)=0,\ 0\leq\ell<2^N\}} v^i_\ell$. Hence, Φ_{int} has $N \cdot 2^N$ variables

$$(v^1_0, \ldots, v^1_{2^N-1}, v^2_0, \ldots, v^2_{2^N-1}, \ldots, v^N_0, \ldots, v^N_{2^N-1})$$

and $poly(N)$ clauses of size $\mathcal{O}(2^N)$. Because Φ is in CNF, it is also the case for Φ_{int}. Moreover, all variables appear positively in Φ_{int}.

It remains to show that Φ_{int} is integrally satisfiable if and only if Φ is satisfiable.

First, let us assume that Φ is satisfiable. Let $u_1, \ldots, u_\eta$ be a valid assignment of its variables and, for any $0 < i < N$, let x_i be the integer with $(u_{N(i-1)+1}, \ldots, u_{N(i-1)+N})$ as binary representation. Finally, for any $i < N$ and $j < 2^N$, let us define $v^i_j = 1$ if $x_i = j$ and

$v^i_j = 0$ otherwise. By definition of Φ_{int}, $(v^i_j)_{0\leq i<N,\ 0\leq j<2^N}$ is a valid assignment and Φ_{int} is therefore integrally satisfiable.

Conversely, let us assume that Φ_{int} is integrally satisfiable and let $(x_1, \ldots, x_N)$ be N integers representing a valid assignment for it. Let $u_1, \ldots, u_\eta$ be defined such that, for any $0 \leq i < N$, $(u_{N(i-1)+1}, \ldots, u_{N(i-1)+N})$ is the binary representation of x_i. Then, $u_1, \ldots, u_\eta$ is a satisfying assignment for Φ which is satisfiable. ◀

3.2 Proof of Proposition 2

Theorem 3 allows us to reduce any 3-SAT instance Φ of size η into an INT-SAT instance Φ_{int} with size $2^{\mathcal{O}(\sqrt{\eta})}$. The key point is that this reduction allows us to turn the choice of η boolean variables into the choice of $\sqrt{\eta}$ integers in $\{0, \ldots, 2^{\sqrt{\eta}}\}$. Then, in further sections, we build a tree T with $2^{\mathcal{O}(\sqrt{\eta})}$ vertices from the formula Φ_{int}, such that there is a one to one mapping between any optimal weighted coloring of T and the $\sqrt{\eta}$-tuples of integers in $\{0, \ldots, 2^{\sqrt{\eta}}\}$. Finally, our reduction ensures that the value of $\chi_w(T)$ depends on the integral satisfiability of Φ_{int} and therefore, on the satisfiability of Φ. More formally, the next section is devoted to proving the following result:

▶ Proposition 3. For any CNF Boolean formula Φ_{int} of size n where all variables $(y^i_j)_{i,j}$ appear positively, there exist a weighted tree $(T(\Phi_{int}), w)$ with size polynomial in n and $M \in \mathbb{R}$ s.t. Φ_{int} is integrally satisfiable w.r.t. $(y^i_j)_{i,j}$ if and only if $\chi_w(T(\Phi_{int})) \leq M$. The pair $(T(\Phi_{int}), w)$ and M are computable in time polynomial in n.

The proof of Proposition 2 is straightforward from Theorem 3 and Proposition 3.

4 Proof of Proposition 3

This section is devoted to the proof of Proposition 3.

Let us introduce some notations. Let $n \in \mathbb{N}$ and let $m = 2^n$. Let Φ_{int} be a Boolean formula with $n \times m$ variables $\{y^j_i \mid 0 \leq i < n, 0 \leq j < m\}$ and L clauses, where L is polynomial in n. We assume that Φ_{int} is in the Conjunctive Normal Form and that each variable appears positively. Moreover, we may assume that each variable appears at least once. That is, $\Phi_{int} = \bigwedge_{\ell \leq L} Q_\ell$ and, for any $\ell \leq L$, Q_ℓ is the disjunction of $p_\ell \geq 1$ positive variables.

Let $\epsilon > 0$ such that $nm\epsilon = o(\frac{1}{2^{4n}})$ and let

$$M = \sum_{i=0}^{4n+2} \frac{1}{2^i} + n(m-1)\epsilon < 2.$$

Let $w^j_i = 1/2^i + j\epsilon$, for any $0 \leq i \leq 4n+3$ and $0 \leq j \leq m$. Let $\mathcal{W} = \{w^j_i \mid 0 \leq i \leq 4n+3, 0 \leq j \leq m\}$ denote a set of weights. Note that the length of the encoding of these weights is polynomially bounded. For any $0 \leq k \leq 3$, let $W_k = w^0_{4n+k} = 1/2^{4n+k}$. Finally, for any rooted tree T, let $r(T)$ denote its root. A rooted tree S is a *(proper) subtree* of a rooted tree T if there is an edge e of T such that S is the connected component of $T \setminus \{e\}$ that does not contain $r(T)$. We now define various subtrees required to build $(T(\Phi_{int}), w)$.

4.1 Binomial trees

We first define a particular family of *binomial trees* B_i, $0 \leq i \leq 4n+2$. They will be used in the construction of $T(\Phi_{int})$. Their role is to force the color of most of the nodes in any coloring c of $T(\Phi_{int})$ with $w(c) \leq M$.

▶ **Definition 4.** For any $0 \leq i \leq 4n+2$, let B_i be the weighted rooted tree defined recursively as follows (see Figure 2).

- if $i = 0$, then B_0 has a unique node with weight w_0^0;
- otherwise, B_i has a root of weight w_i^0 whose children are the roots of copies of $B_0, B_1, \ldots, B_{i-1}$.

Note that B_i has 2^i nodes and that it just contains nodes of weight w_j^0, for $0 \leq j \leq i \leq 4n+2$. We will use these binomial trees with two main goals in our reduction:

- enforce the number of used colors and the weights of these colors (up to an additive constant $c\epsilon$) in any optimal weighted coloring of the tree we build from the 3-SAT formula;
- forbid the color i to appear in any vertex that is adjacent to a root of a binomial tree B_i.

We get these properties according to the following lemmas:

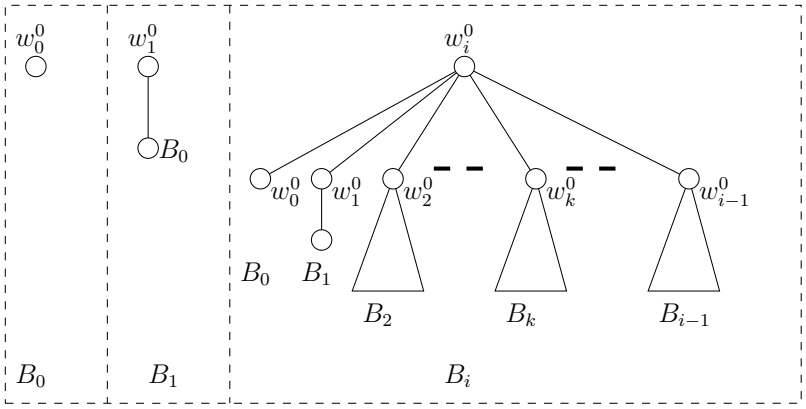

Figure 2 The construction of the binomial tree B_i.

▶ **Lemma 5.** *Let $0 \leq i \leq 4n+2$. Let (T, w) be a weighted tree having B_i as subtree. If there exists a coloring c of (T, w) with $w(c) \leq M$, then, for any $0 \leq k \leq i$:*

1. *all vertices of B_i with weight in w_k^0 receive the same color S_k of c; and*
2. *there exists a unique color class S_k in c of weight in $\{w_k^j \mid 0 \leq j \leq m\}$.*

Proof. The proof is by induction on the index i. In case $i = 0$, we prove both statements of the lemma at once by observing that any two vertices of (T, w) of weight in $\{w_0^{j'} \mid 0 \leq j' \leq m\}$ must belong to the same color class S_0, otherwise we would conclude that $w(c) \geq 2$, that would be a contradiction to the fact that $w(c) \leq M < 2$.

Now, let $0 \leq k \leq i$, observe that the set of nodes of B_i with weight in w_k^0 is an independent set that dominates the nodes of B_i with smaller weights (i.e., in $\{w_{k'}^0 \mid k < k' \leq i\}$).

By induction hypothesis, for any $0 \leq k < i$, the set of nodes of B_i with weight in w_k^0 receive the same color S_k of c and this color class is the unique with weight in $\{w_k^j \mid 0 \leq j \leq m\}$. Then, for any $0 \leq k < i$, the root of B_i cannot be colored S_k, since it has a neighbor with weight w_k^0. Let S_i be the color of the root of B_i in c. We proved that the color S_i cannot contain nodes with weight greater than w_i^{m-1} and that c cannot have another color $S_i' \neq S_i$ with weight in $\{w_i^j \mid 0 \leq j \leq m\}$, because, otherwise the weight of c would be at least $\frac{1}{2^i} + \sum_{k=0}^{i} \frac{1}{2^k} = 2 > M$ in both cases. ◀

▶ **Corollary 6.** *Let (T, w) be a weighted tree having B_{4n+2} as subtree. Let c be any coloring of (T, w) s.t. $w(c) \leq M$. Then, $c = (S_0, \ldots, S_k)$ with $k \geq 4n+2$ and, for any $0 \leq i \leq 4n+2$, S_i is the unique color with weight in $\{w_i^j \mid 0 \leq j \leq m\}$.*

The trees we consider below will always satisfy the requirements of Corollary 6. Therefore, we are able to identify a color by its weight. In other words, in what follows, for any coloring $c = (S_0, \ldots, S_k)$ of weight at most M and for any $i \leq 4n+2$, S_i will be the unique color such that $w(S_i) \in \{w_i^j \mid 0 \leq j \leq m\}$.

Recall that we defined, for any $0 \leq k \leq 3$, $W_k = w_{4n+k}^0 = 1/2^{4n+k}$. By a slight abuse of notation, for any $0 \leq k \leq 3$, we denote $W_k = S_{4n+k}$ as the unique color with weight W_k.

4.2 Auxiliary trees and Variable-trees

This section is mainly devoted to the construction of subtrees that will represent the boolean variables. First, the family of *auxiliary trees* A_i^j, $0 \leq i < 4n, 0 \leq j \leq m$, will be used to introduce some choice when coloring $T(\Phi_{int})$.

▶ **Definition 7.** For any $0 \leq i < 4n, 0 \leq j \leq m$, let A_i^j be the weighted rooted tree defined as follows (see Figure 3). Note that A_i^j consists of 2^{4n} nodes.

1. Let u be its root with weight $w(u) = W_0$, and connect it to a node v (its *s*ubroot) with weight $w(v) = w_i^j$;
2. v is made adjacent to the root of a copy of B_ℓ, for any $0 \leq \ell < i-1$;
3. u is made adjacent to the root of a copy of B_ℓ, for any $0 \leq \ell < 4n$, $\ell \neq i-1$.

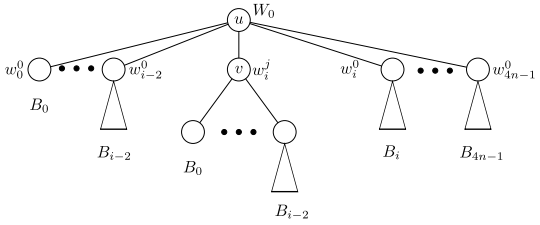

Figure 3 Auxiliary tree A_i^j.

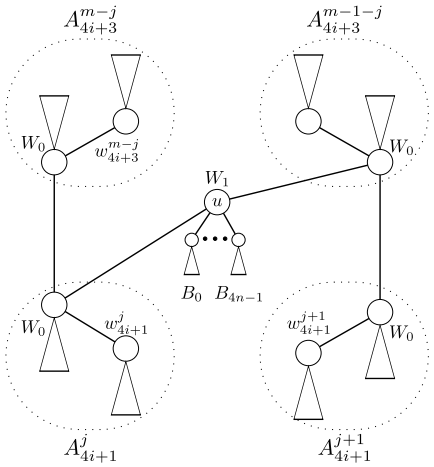

Figure 4 The variable tree $T(y_i^j)$.

▶ **Lemma 8.** *Let $0 \leq i < 4n$ and $0 \leq j \leq m$. Let (T, w) be any weighted tree having B_{4n+2} and A_i^j as subtrees. Let u and v be the root and the sub-root of A_i^j, respectively. For any coloring c of (T, w) with weight $w(c) \leq M$, then it holds:*

- *either v is colored S_{i-1} and u must be colored with the color W_0;*
- *or v is colored S_i (therefore, $w(S_i) \geq w_i^j$) and u can be colored with S_{i-1}.*

Proof. Recall that, by Corollary 6, we can identify the colors of c and their weights. By Lemma 5, the root of each subtree B_k, $0 \leq k < 4n$, must be colored with S_k and then the sub-root v can be colored only with color S_{i-1} or S_i. Note that, if v is colored with color S_p for some $p > i$, then $w(S_p) \geq w_i^j$, contradicting Corollary 6. In the first case, u is adjacent to a node with color S_k, for any $k < 4n$. Therefore, u must be colored with color $S_{4n} = W_0$. Otherwise, u may be colored with color S_{i-1}. ◀

Intuitively, the previous lemma states that, either we "pay" $j\epsilon$ in the weight of color S_i, or u must be colored with the color W_0. We now define the *variable-trees* $T(y_i^j)$ using the auxiliary trees.

▶ **Definition 9.** For any $0 \leq i < n, 0 \leq j < m$, let $T(y_i^j)$ be the weighted rooted tree, *representing the variable* y_i^j, defined as follows (see Figure 4):

- let u be its root with weight $w(u) = W_1$ and connected to the root of a copy of B_ℓ, for any $0 \leq \ell < 4n$;
- take one copy of $A_{4i+1}^j, A_{4i+1}^{j+1}, A_{4i+3}^{m-j}$ and A_{4i+3}^{m-1-j} and:
  - connect $r(A_{4i+1}^j)$ to $r(A_{4i+3}^{m-j})$, and $r(A_{4i+1}^{j+1})$ to $r(A_{4i+3}^{m-1-j})$;
  - connect u with $r(A_{4i+1}^j)$ and $r(A_{4i+3}^{m-j-1})$.

Note that $T(y_i^j)$ consists of $\mathcal{O}(2^{4n})$ nodes (i.e. polynomial in nm).

▶ **Lemma 10.** *Let (T, w) be any weighted tree having B_{4n+2} as subtree and containing $T(y_i^j)$ as subtree, for all $0 \leq i < n$ and $0 \leq j < m$. Let c be a coloring of T with weight $w(c) \leq M$.*

Then, there are $(j_0, \ldots, j_{n-1}) \in \{0, \ldots, m\}^n$ such that each root u of each subtree $T(y_i^j)$, for any $0 \leq i < n$ and $0 \leq j < m$, satisfies:

- *if $j \neq j_i$, then the color of u in c must be W_1;*
- *otherwise, neither of the two neighbors of u can be colored W_1 and neither of these two nodes need to be colored W_0.*

Proof. Since T contains B_{4n+2}, by Corollary 6, a coloring $c = (S_0, \ldots, S_k)$ of weight $w(c) \leq M$ is such that $k \geq 4n + 2$, and, for any $0 \leq i \leq 4n + 2$, S_i is the unique color such that $w(S_i) \in \{w_k^j \mid 0 \leq j \leq m\}$. In particular, $w(c) \geq \sum_{i=0}^{4n+2} 1/2^i = M - n(m-1)\epsilon$.

For any $0 \leq i < n$, let $j_i \leq m$ be such that $w(S_{4i+1}) = w_{4i+1}^{j_i}$.

First, let us assume that $j_i < m$. In particular, this means that every sub-root of a subtree A_{4i+1}^r, for each $j_i < r \leq m$, is colored S_{4i} (recall that its color is either S_{4i} or S_{4i+1}, by Lemma 8). Consequently, any root of a subtree A_{4i+1}^r, for each $j_i < r \leq m$, must be colored W_0. Therefore, by the construction of the variable-trees, any root of a subtree A_{4i+3}^{m-r}, for each $j_i < r \leq m$, cannot be colored W_0 because it is adjacent to a root of a subtree A_{4i+1}^r. Thus, by Lemma 8, it must be colored S_{4i+2} and the color of each sub-root of A_{4i+3}^{m-r} must be S_{4i+3}. Consequently, $w(S_{4i+3}) \geq w_{4i+3}^{m-(j_i+1)}$. Hence, for any $0 \leq i < n$, if $j_i < m$, we conclude that $w(S_{4i+3}) + w(S_{4i+1}) \geq w_{4i+1}^{j_i} + w_{4i+3}^{m-(j_i+1)} = (m-1)\epsilon + 1/2^{4i+1} + 1/2^{4i+3}$.

On the other hand, if $j_i = m$, it follows directly that $w(S_{4i+3}) + w(S_{4i+1}) \geq m\epsilon + 1/2^{4i+1} + 1/2^{4i+3}$.

Since $w(c) \leq M$, it implies that, for any $0 \leq i < n$, $j_i < m$ and $w(S_{4i+3}) = w_{4i+3}^{m-j_i-1}$ and, for any $0 \leq 2k < 4n$, $w(S_{2k}) = w_{2k}^0$. Consequently, by a similar argument, the roots of all subtrees A_{4i+3}^{m-j}, for each $0 \leq j \leq j_i$, must be colored W_0 and, then, the roots of all subtrees A_{4i+1}^r, for each $0 \leq j \leq j_i$, must be colored S_{4i}.

Let $0 \leq i < n$ and $0 \leq j < m$. Consider a subtree $T(y_i^j)$ of T. If $j \neq j_i$, then (exactly) one of the roots of A_{4i+1}^j and A_{4i+3}^{m-1-j} must be colored W_0. In that case, the color of the root u of $T(y_i^j)$ must be W_1. Indeed, u is adjacent to the root of B_k, $0 \leq k < 4n$, and therefore it cannot be colored S_k. Moreover, if u is colored W_2, then we have a contradiction as $w(c) > M$, because $w(u) = W_1$. On the other hand, if $j = j_i$, none of the roots of A_{4i+1}^j and A_{4i+3}^{m-1-j} need to be colored W_0. Finally, none of the roots of A_{4i+1}^j and A_{4i+3}^{m-1-j} can be colored W_1 because their weight is W_0 (it would imply $w(c) > M$). ◀

4.3 Clause-trees and definition of $T(\Phi_{int})$

We define the subtrees representing the clauses and combine them to get $T(\Phi_{int})$.

▶ **Definition 11.** Let $Q_\ell = \vee_{1 \leq k \leq p_\ell} u_k$ be any clause of Φ_{int} (recall that, for any $1 \leq k \leq p_\ell$, $u_k \in \{y_i^j \mid 0 \leq i < n, 0 \leq j < m\}$ and that $\ell \leq L$). For any $1 \leq k \leq p_\ell$, let $T(Q_\ell^k)$ be the rooted weighted tree defined recursively as follows:

1. $T(Q_\ell^1) = T(u_1)$;
2. for any $k > 1$, start with one copy of $T(Q_\ell^{k-1})$ with root a and one copy of $T(u_k)$ with root b. Let c, d be two nodes with weight W_1 and e, f be two nodes with weight W_2. For each node $v \in \{c, d, e, f\}$, and for any $0 \leq i < 4n$, add one copy of B_i and make its root adjacent to v. Add one copy of B_{4n+1} and make its root adjacent to e. Finally, we add the edges $\{\{a, f\}, \{b, c\}, \{c, f\}, \{d, e\}, \{e, f\}\}$ and d is chosen as the root.

Let us note $T(Q_\ell) = T(Q_\ell^{p_\ell})$ the *clause-tree* corresponding to Q_ℓ and that consists of $\mathcal{O}(p_\ell 2^{4n})$ nodes (i.e. polynomial in nm). $T(Q_\ell^k)$ is depicted in Figure 5.

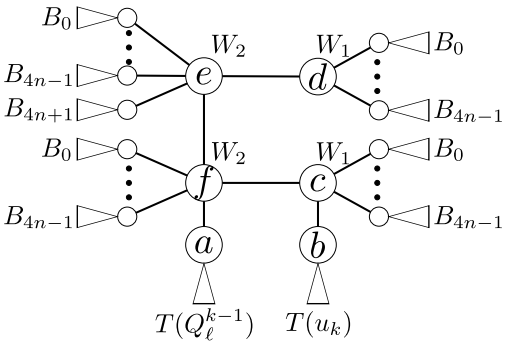

Figure 5 The clause tree $T(Q_\ell^k)$.

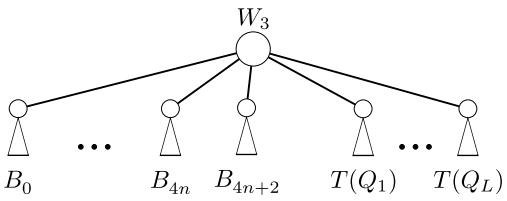

Figure 6 The final tree $T(\Phi_{int})$.

▶ **Lemma 12.** *Let (T, w) be any weighted tree having B_{4n+2} as subtree and containing $T(Q_\ell^k)$ as a subtree ($\ell \leq L$, $k \leq p_\ell$). Let c be any coloring of T with weight $w(c) \leq M$. If a and b are colored W_1, then the color of the root d of $T(Q_\ell^k)$ must be W_1;*

Proof. We prove it by induction on the number of variables k of Q_ℓ^k. Observe that in case $k = 1$, then $T(Q_\ell^k)$ is a variable-tree and the lemma trivially holds as the vertex b does not exist, thus the first statement is trivially satisfied, and, by Lemma 10, the color of its root must be either W_0 or W_1.

Now, consider that a and b are roots of a variable-tree and of a clause-tree on $k-1$ variables $T(Q_\ell^{k-1})$, respectively. By Lemma 10 and by the inductive hypothesis, the colors of a and b are either W_0 or W_1.

In case $c(a) = c(b) = W_1$, by the hypothesis $w(c) \leq M$, by Lemma 5 and Corollary 6, we conclude that c is colored W_0, f is colored W_2, e is colored W_0 and d is forced to be colored W_1. This proves the first statement of the lemma. Finally, by the construction of $T(Q_\ell^k)$, by Lemma 5 and Corollary 6, the root d may be colored either W_0 or W_1, since $w(c) \leq M$. ◀

▶ **Definition 13.** Let $T(\Phi_{int})$ be the weighted rooted tree obtained as follows (see Figure 6). Let r be the root with weight W_3. For any $1 \leq \ell \leq L$, the root of one copy of $T(Q_\ell)$ is made adjacent to r. For any $0 \leq i \leq 4n+2$, $i \neq 4n+1$, r is made adjacent to the root of one copy of B_i.

▶ **Lemma 14.** *$T(\Phi_{int})$ has size polynomial in $m = 2^n$.*

Proof. Observe that each clause-tree $T(Q_\ell)$ has size $\mathcal{O}(p_\ell 2^{4n})$ (see Definition 11), where p_ℓ is polynomial in m (since p_ℓ is at most the number nm of variables). Moreover, the number L of clauses is polynomial in m by the definition of Φ_{int}. ◀

▶ **Lemma 15.** *If Φ_{int} is integrally satisfiable, then $\chi_w(T(\Phi_{int})) \leq M$.*

Proof. Let $(y_i^j)_{i<n,j<m}$ be a valid integral assignment for Φ_{int}. For any $0 \leq i < n$, let j_i be the (unique) index such that $y_i^{j_i}$ is true. We construct a coloring c of $(T(\Phi_{int}), w)$ such that $w(c) \leq M$. By Lemma 5, in any coloring c of $T(\Phi_{int})$ such that $w(c) \leq M$, the colors of all nodes of the binomial subtrees of $T(\Phi_{int})$ are forced. Consequently, we only need to decide the colors of the following nodes: the roots and sub-roots of any copy of A_i^j, the roots of the trees $T(y_i^j)$, and the nodes added to connect the variables-trees into clause-trees (the nodes a, b, c, d, e, f in Figure 5), for any $0 \leq i < n$ and $0 \leq j < m$.

We first set the weight of color S_i for any $0 \leq i < 4n$. In particular, for any $0 \leq i < n$, the color S_{4i+1} must have weight $w_{4i+1}^{j_i}$. As we observed in the proof of Lemma 10, this choice fixes the colors of all roots and sub-roots of all the A_i^j trees, i.e. all the nodes in the variable trees, except to the roots of the variable-trees $T(y_i^{j_i})$, by Lemma 10.

More precisely, for any $0 \leq i < n$ and $0 \leq j < m$, let us consider a subtree $T(y_i^j)$. Let $j' \in \{j, j+1\}$. The sub-root of $A_{4i+1}^{j'}$ receives color S_{4i+1} if $j' \leq j_i$ and receives color S_{4i} otherwise. The root of $A_{4i+1}^{j'}$ receives color S_{4i} if $j' \leq j_i$ and receives color W_0 otherwise. The sub-root of $A_{4i+3}^{m-j'}$ receives color S_{4i+3} if $j' > j_i$ and receives color S_{4i+2} otherwise. The root of $A_{4i+3}^{m-j'}$ receives color S_{4i+2} if $j' > j_i$ and receives color W_0 otherwise. Finally, if $j \neq j_i$, the root of $T(y_i^j)$ is colored W_1. On the other hand, if $j = j_i$, none of the neighbors of the root of $T(y_i^j)$ is colored W_0, therefore, we can color it either W_0 or W_1.

Now, let $Q_\ell = \vee_{1 \leq k \leq p_\ell} u_k$ be any clause of Φ_{int}. We show that we can extend the previous coloring such that the root of the clause-tree $T(Q_\ell)$ is colored W_0 and the weight of the coloring is $< M$. This is by induction on p_ℓ. Indeed, if $p_\ell = 1$, then Q_ℓ consists of a unique variable and this variable must be assigned to true (since the formula is true). Hence, $Q_\ell = y_i^{j_i}$ for some $0 \leq i < n$. That is $T(Q_\ell)$ is a subtree $T(y_i^{j_i})$. Hence, we can color the root of it with W_0.

Now, assume that the result is correct for any clause of length $p \geq 1$ and let $p_\ell = p + 1$. Thus, $Q_\ell = u_{p+1} \vee (\vee_{1 \leq k \leq p} u_k)$. Recall that $T(Q_\ell)$ is built from a variable subtree $T(u_{p+1})$ and a clause-subtree $T(Q_\ell^p)$. There are two cases to consider: either our assignment satisfies $\vee_{1 \leq k \leq p} u_k$ or not. In the first case, the root of the clause-tree $T(Q_\ell^p)$ (node b in Figure 5) is colored W_0 by induction. Moreover, by above paragraphs, the root of $T(u_{p+1})$ (node a in Figure 5) can be colored W_1. It is then easy to extend this coloring such that the root of $T(Q_\ell)$ is colored W_0: in Figure 5, node c is colored W_1, node e is colored W_2 and nodes f and d are colored W_0. If our assignment does not satisfy $\vee_{1 \leq k \leq p} u_k$, then it must satisfy u_{p+1}. That is, $u_{p+1} = y_i^{j_i}$ for some $0 \leq i < n$. By a similar induction, we prove that the root of $T(Q_\ell^p)$ can be colored W_1. Moreover, by above paragraphs, the root of $T(u_{p+1}) = T(y_i^{j_i})$ can be colored W_0. This coloring can be extended such that the root of $T(Q_\ell)$ is colored W_0: in Figure 5, node f is colored W_1, node e is colored W_2 and nodes c and d are colored W_0.

Thus, we color the roots of all the clause-trees with color W_0 and the root of $T(\Phi_{int})$ with the color W_1.

Hence, the weight of this coloring c is $w(c) = \sum_{i=0}^{4n+2} \frac{1}{2^i} + n(m-1)\epsilon = M$. ◀

▶ **Lemma 16.** *If Φ_{int} is not integrally satisfiable, then $\chi_w(T(\Phi_{int})) > M$.*

Proof. Φ_{int} is not integrally satisfiable. Let c be a coloring of $T(\Phi_{int})$ with weight at most M. By Lemma 10, there are integers $(j_0, \ldots, j_{n-1})$ such that the color of the root of any subtree $T(y_i^j)$ is forced to be W_1, if $j \neq j_i$. Let $Y = (y_i^j)_{i<n,j<m}$ be the corresponding integral assignment. In other words, for any variable y_i^j $(0 \leq i < n, 0 \leq j < m)$, $y_i^j = 0$ if $j \neq j_i$. Since Φ_{int} is not integrally satisfiable, there is a clause Q that is not satisfied by this

assignment. Let us consider the clause-subtree $T(Q)$. It has been built from variable-trees corresponding to the variables constituting the clause Q. Because all these variables are assigned to false, the roots of these variable-trees are all colored with W_1, by Lemma 10.

By induction on the length of Q and by Lemma 12, the color of the root of $T(Q_\ell)$ must be W_1. Thus, the root of $T(\Phi_{int})$ can just be colored W_3. Consequently, the coloring c has weight $w(c) \geq \sum_{i=0}^{4n+3} \frac{1}{2^i} + n(m-1)\epsilon > M$. ◀

Proposition 3 follows directly from Lemmas 14, 15 and 16.

References

1 Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symp. on Theory of Computing (STOC)*, pages 151–158, 1971.

2 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.

3 Marc Demange, Dominique de Werra, Jérôme Monnot, and Vangelis Th. Paschos. Weighted node coloring: When stable sets are expensive. In *28th Int. Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 2573 of *LNCS*, pages 114–125. Springer, 2002.

4 Leah Epstein and Asaf Levin. On the max coloring problem. *Theor. Comput. Sci.*, 462:23–38, 2012.

5 Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Weighted coloring: further complexity and approximability results. *Inf. Process. Lett.*, 97(3):98–103, 2006.

6 D. J. Guan and Xuding Zhu. A coloring problem for weighted graphs. *Inf. Process. Lett.*, 61(2):77–81, 1997.

7 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

8 Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

9 Telikepalli Kavitha and Julián Mestre. Max-coloring paths: tight bounds and extensions. *J. Comb. Optim.*, 24(1):1–14, 2012.

10 C. Linhares Sales and B. Reed. Weighted coloring on graphs with bounded tree width. In *Annals of 19th International Symposium on Mathematical Programming*, 2006.

11 Sriram V. Pemmaraju, Sriram Penumatcha, and Rajiv Raman. Approximating interval coloring and max-coloring in chordal graphs. In *Proc. 3rd Int. Workshop on Experimental and Efficient Algorithms (WEA)*, volume 3059 of *LNCS*, pages 399–416. Springer, 2004.

12 Sriram V. Pemmaraju, Sriram Penumatcha, and Rajiv Raman. Approximating interval coloring and max-coloring in chordal graphs. *ACM J. of Exp. Algorithmics*, 10, 2005.

13 Sriram V. Pemmaraju and Rajiv Raman. Approximation algorithms for the max-coloring problem. In *Proceedings 32nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of *LNCS*, pages 1064–1075. Springer, 2005.

14 Sriram V. Pemmaraju, Rajiv Raman, and Kasturi R. Varadarajan. Buffer minimization using max-coloring. In *15th ACM-SIAM Symp. on Discrete Alg. (SODA)*, pages 562–571, 2004.

15 Sriram V. Pemmaraju, Rajiv Raman, and Kasturi R. Varadarajan. Max-coloring and online coloring with bandwidths on interval graphs. *ACM Trans. on Algorithms*, 7(3):35, 2011.

16 Ryan Williams. A new algorithm for optimal constraint satisfaction and its implications. In *31st International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3142 of *Lecture Notes in Computer Science*, pages 1227–1237. Springer, 2004.