Speeding up Dynamic Programming for Some NP-hard Graph Recoloring Problems

Oriana Ponta[1,⋆], Falk Hüffner[2,⋆], and Rolf Niedermeier[2]

[1] Mathematisches Institut, Ruprecht-Karls-Universität Heidelberg, Im Neuenheimer Feld 288, D-69120 Heidelberg, Germany
oriana@urz.uni-heidelberg.de
[2] Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany
{hueffner,niedermr}@minet.uni-jena.de

Abstract. A vertex coloring of a tree is called *convex* if each color induces a connected component. The NP-hard Convex Recoloring problem on vertex-colored trees asks for a minimum-weight change of colors to achieve a convex coloring. For the non-uniformly weighted model, where the cost of changing a vertex v to color c depends on both v and c, we improve the running time on trees from $O(\Delta^\kappa \cdot \kappa n)$ to $O(3^\kappa \cdot \kappa n)$, where Δ is the maximum vertex degree of the input tree T, κ is the number of colors, and n is the number of vertices in T. In the uniformly weighted case, where costs depend only on the vertex to be recolored, one can instead parameterize on the number of *bad* colors $\beta \leq \kappa$, which is the number of colors that do not already induce a connected component. Here, we improve the running time from $O(\Delta^\beta \cdot \beta n)$ to $O(3^\beta \cdot \beta n)$. For the case where the weights are integers bounded by M, using fast subset convolution, we further improve the running time with respect to the exponential part to $O(2^\kappa \cdot \kappa^4 n^2 M \log^2(nM))$ and $O(2^\beta \cdot \beta^4 n^2 M \log^2(nM))$, respectively. Finally, we use fast subset convolution to improve the exponential part of the running time of the related 1-Connected Coloring Completion problem.

1 Introduction

The issue of recoloring vertex-colored graphs by a minimum-cost set of color changes in order to achieve a desired property of the color classes such as being connected has recently received considerable attention; approximation as well as fixed-parameter algorithms have been developed for the corresponding NP-hard problems [2, 7, 8, 15, 16, 20]. Here, we focus on exact fixed-parameter algorithms [9, 11, 18] for two prominent types of these problems, significantly improving on the associated exponential running time factors. The two types of problems we investigate are as follows. First, we study vertex-colored trees and the task is to recolor some of its vertices such that each color class forms

⋆ Supported by the Deutsche Forschungsgemeinschaft, Emmy Noether research group PIAF (fixed-parameter algorithms), NI 369/4.

a connected component. The problem was introduced by Moran and Snir [15], and it concerns the major part of this work. The second type of problem is not only concerned with trees, but with general graphs. However, here we cannot recolor freely, but a subset of vertices is uncolored and the task is to complete the coloring such that each color class forms a connected component [8].

Convex Recoloring. Most of this work deals with convex recoloring problems on trees. The most general version, that is, *non-uniformly weighted*, is defined as follows.

Convex Recoloring
Instance: A tree $T = (V, E)$ with a vertex coloring $C : V \to \mathcal{C}$ and a weight function $w : V \times \mathcal{C} \to \mathbb{Q}_+$, where $w(v, C(v)) = 0$ for all $v \in V$.
Task: Find a convex coloring $C' : V \to \mathcal{C}$ with minimum weight $w(C') := \sum_{v \in V} w(v, C'(v))$.

We defined Convex Recoloring only for trees. There are some positive results for the special case of paths [15, 16], but there do not seem to be positive results for general graphs (Moran et al. [17] considered the slightly more general class of galled networks).

Let κ be the number of colors $|\mathcal{C}|$ and n the number of vertices $|V|$. Let $\beta \leq \kappa$ be the number of *bad* colors, that is, colors that do not already induce a connected component.

Convex Recoloring was introduced by Moran and Snir [15], who showed that the decision version is NP-complete, even for unweighted paths. They also gave an algorithm for non-uniformly weighted Convex Recoloring running in $O(\Delta^\kappa \kappa n)$ time, where Δ is the maximum degree of the input graph. They further gave an algorithm running in $O((\kappa/\log\kappa)^\kappa \cdot \kappa n^4)$ time, thus showing that the problem is fixed-parameter tractable with respect to the parameter κ. For the uniformly weighted case, they showed that κ can be replaced by the potentially smaller parameter β in these running times.

For the unweighted case, Razgon [20] gave an $256^k \cdot n^{O(1)}$ time algorithm, where k is the number of vertices recolored. This can be related to other results by noting that $k \geq \beta/2$ (every color change can make at most two bad colors good). Bodlaender and Weyer [5] considered a different parameter, namely the *separation of colors* ℓ, which is the maximum number of colors separated by a vertex, where we say that a vertex v separates a color c if there is a path between two vertices of color c that passes through v. They presented a running time of $O(3^\ell \cdot \ell^3 n)$. Since they showed that $\ell \leq k+1$, this also improves Razgon's result to $O(3^k \cdot kn)$. Bar-Yehuda et al. [2] further improved the bound to $O(2^k \cdot kn + n^2)$ by doing a better analysis of a variant of the dynamic programming algorithm of Moran and Snir [15]. Bodlaender et al. [6] showed a problem kernel with $O(k^6)$ vertices, which was later improved to $O(k^2)$ vertices [7]. Finally, Moran and Snir [16] gave a factor-3 approximation for the uniformly weighted case running in $O(\kappa n^2)$ time. This was improved to a $(2+\epsilon)$-approximation running in $O(n^2 + n(1/\epsilon)^2 4^{1/\epsilon})$ time [2].

Bachoore and Bodlaender [1] gave an $O(4^k n)$ time algorithm for the variant where only the leaves are precolored.

Connected Coloring Completion. The second type of problem we study has only recently been introduced by Chor et al. [8]; accordingly, so far less results are known for this problem. As Convex Recoloring, it is motivated by applications in bioinformatics.

> 1-Connected Coloring Completion
> **Instance:** A graph $G = (V, E)$ with k uncolored vertices $U \subseteq V$ and a vertex coloring $C : V \setminus U \to \mathcal{C}$.
> **Task:** Find a convex coloring $C' : V \to \mathcal{C}$ that extends C, that is, for all $v \in V \setminus U : C'(v) = C(v)$.

Chor et al. [8] also considered the more general r-Connected Coloring Completion, where the goal is to find a coloring where each color induces at most r connected components. They showed that 1-Connected Coloring Completion is NP-hard, even for only two colors, but can be solved in $O(8^k \cdot k + 2^k \cdot kn)$ time on an n-vertex graph. They further showed that for the parameter treewidth, r-Connected Coloring Completion is fixed-parameter tractable for $r = 1$ but W[1]-hard for $r \geq 2$.

Our contributions. The main purpose of this paper can be seen in "engineering" dynamic programs for weighted Convex Recoloring problems and for (unweighted) 1-Connected Coloring Completion with respect to their exponential running time factors. To this end, we make use of two main technical tricks investigated in greater depth in the following two sections. First, we observe how a method for tree problems originally going back to Maffioli [14] (which meanwhile has found several applications, see, e.g., [4, 5]) also helps to significantly speed up and somewhat simplify dynamic programming algorithms for weighted convex recoloring problems. Second, we show how a recent general breakthrough result of Björklund et al. [3] concerning a more efficient computation of subset convolutions can be tailored towards applying it to recoloring problems.[3] More specifically, for non-uniformly weighted Convex Recoloring we improve a previous exponential factor of Δ^κ to 3^κ and further on to 2^κ, and for uniformly weighted Convex Recoloring we improve a previous exponential factor of Δ^β to 3^β and further on to 2^β; herein, Δ denotes the maximum vertex degree in the tree. Note that the improvements from exponential base 3 to 2 come along with increased polynomial factors in the running time. Finally, we also adapt the subset convolution trick to 1-Connected Coloring Completion in order to improve the previous exponential factor of 8^k to 4^k.

[3] Lingas and Wahlen [13] recently presented an application in the context of subgraph homeomorphism problems.

2 Fine-Grained Dynamic Programming

The major part of this work is concerned with improvements for non-uniformly and uniformly weighted Convex Recoloring based on a more efficient dynamic programming strategy. The essence of the underlying trick can be traced back to work of Maffioli [14]. We start with the somewhat less technical case concerning a dynamic program for *non-uniformly weighted* Convex Recoloring with respect to the parameter "number of colors" and then extend our findings to *uniformly weighted* Convex Recoloring with respect to the parameter "number of bad colors".

2.1 Non-Uniformly Weighted Convex Recoloring

In this section, we show how to improve the running time of the dynamic programming by Moran and Snir [15] from $O(\Delta^\kappa \cdot \kappa n)$ to $O(3^\kappa \cdot \kappa n)$, where κ is the number of colors and n is the number of vertices in the input graph. The dynamic programming works bottom-up from the leaves of the tree. The improvement comes from not considering all children of an inner vertex at once, but rather taking them into account one-by-one. This is a classical trick for dynamic programming on trees (see e. g., [14, 5, 4]). A more detailed presentation of our result is given in the thesis of Ponta [19].

We designate an arbitrary vertex r of T as the root. For each vertex $v \in V$, we denote by T_v the subtree induced by v and all descendants of v. For a vertex v with children $w_1, \ldots, w_p$ in an arbitrary but fixed order, we denote by $T_{v,i}$ the subtree induced by v, the first i children $w_1, \ldots, w_i$ of v, and all descendants of $w_1, \ldots, w_i$. Note that $T_{v,0}$ contains only the vertex v and that $T_{v,p}$ equals T_v.

The basic structure of Moran and Snir's original algorithm is preserved. The algorithm visits the vertices in postorder. We start by determining the trivial convex recolorings for the leaves of the tree and proceed with the computation of weights of convex recolorings of subtrees T_v for internal vertices v in a bottom-up fashion. A solution for T_v is constructed using the previously computed solutions for the subtrees induced by the children of v. The way a solution for the extended problem is computed differs from Moran and Snir's algorithm and is the key to the running time improvement.

For the description of the algorithm, we need two dynamic programming tables denoted by opt and opt_r. Let $C'[T_v]$ be the set of colors appearing in the subtree T_v.

Definition 1. *Let $v \in V$ and $\mathcal{D} \subseteq \mathcal{C}$ be a set of colors. A recoloring C' is a* $(T_v, \mathcal{D})$-coloring *if it is a convex recoloring of T_v such that $C'[T_v] = \mathcal{D}$. The cost of an optimal $(T_v, \mathcal{D})$-coloring of T_v is denoted by* $\mathrm{opt}(T_v, \mathcal{D})$.

If T_v has less than $|\mathcal{D}|$ vertices or $\mathcal{D} = \emptyset$, then no $(T_v, \mathcal{D})$-coloring exists, and we set $\mathrm{opt}(T_v, \mathcal{D}) = \infty$. A $(T_v, \mathcal{D})$-coloring is a convex recoloring of T_v that uses exactly the colors from $\mathcal{D}$. Thus, the cost of an optimal convex recoloring of T can be calculated as $\min_{\mathcal{D} \subseteq \mathcal{C}} \mathrm{opt}(T_r, \mathcal{D})$. To retrieve the recoloring that realizes this cost, we can use standard dynamic programming backtracing methods. It remains to describe how to fill in the dynamic programming table opt. For this, we need a second table opt_r.

Definition 2. *Let $v \in V$, $\mathcal{D} \subseteq \mathcal{C}$ and $c \in \mathcal{C}$. A recoloring C' is a $(T_v, \mathcal{D}, c)$-*coloring *if it is a $(T_v, \mathcal{D})$-coloring such that $C'(v) = c$. The cost of an optimal $(T_v, \mathcal{D}, c)$-coloring is denoted by* $\mathrm{opt}_r(T_v, \mathcal{D}, c)$.

We set $\mathrm{opt}_r(T_v, \mathcal{D}, c) = \infty$ if $c \notin \mathcal{D}$. It is easy to calculate opt from opt_r:

$$\mathrm{opt}(T_v, \mathcal{D}) = \min_{c \in \mathcal{D}} \mathrm{opt}_r(T_v, \mathcal{D}, c). \tag{1}$$

For a subtree T_v consisting of only the vertex v, we set $\mathrm{opt}_r(T_v, \{c\}, c) = w(v, c)$ and $\mathrm{opt}_r(T_v, \mathcal{D}, c) = \infty$ for $\mathcal{D} \neq \{c\}$. For an interior vertex v with children $w_1, \ldots, w_p$, we inductively assume that the values $\mathrm{opt}_r(T_{w_i}, \cdot, \cdot)$ for $1 \leq i \leq p$ are already calculated. We then iteratively calculate $\mathrm{opt}_r(T_{v,i+1}, \cdot, \cdot)$ for $i = 0, \ldots, p$, obtaining $\mathrm{opt}_r(T_v, \cdot, \cdot) = \mathrm{opt}_r(T_{v,p}, \cdot, \cdot)$. Thus, each iteration has to take into account the subtree $T_{w_{i+1}}$ in addition to the subtree $T_{v,i}$ considered in the previous iteration. In contrast, Moran and Snir [15] take into account all child subtrees at once. The childwise iterative approach to dynamic programming on trees has also been used e. g. to find minimum-weight subtrees of a tree [14, 4] or for unweighted Convex Recoloring with a different parameter [5]. In our context, the technique allows to avoid the maximum vertex degree Δ in the base of the exponential part of the running time of Moran and Snir's algorithm.

For a simpler notation of the recurrence for $\mathrm{opt}_r(T_{v,i+1}, \cdot, \cdot)$, we define the function $\mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}, c)$ for the $(i+1)$th child w_{i+1} of v and $\mathcal{D} \subseteq \mathcal{C}, v \in \mathcal{C}$ as

$$\mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}, c) = \min\{\mathrm{opt}(T_{w_{i+1}}, \mathcal{D} \setminus \{c\}), \mathrm{opt}_r(T_{w_{i+1}}, \mathcal{D} \cup \{c\}, c)\}. \tag{2}$$

Thus, the value $\mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}, c)$ is the minimum cost of a convex recoloring C' of $T_{w_{i+1}}$ that uses every color in $\mathcal{D} \setminus \{c\}$, no color from $\mathcal{C} \setminus (\mathcal{D} \cup \{c\})$, and uses color c in $T_{w_{i+1}}$ only if $C'(w_{i+1}) = c$.

The following lemma describes the central recurrence for opt_r.

Lemma 1. *Let v be an interior vertex with children $w_1, \ldots, w_p$. For any color set $\mathcal{D}$ and any color $c \in \mathcal{D}$ it holds that*

$$\mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c) = \min_{\substack{\mathcal{D}_1 \cup \mathcal{D}_2 = \mathcal{D} \setminus \{c\} \\ \mathcal{D}_1 \cap \mathcal{D}_2 = \emptyset}} (\mathrm{opt}_r(T_{v,i}, \mathcal{D}_1 \cup \{c\}, c) + \mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}_2, c)). \tag{3}$$

Proof. "$\geq$": Let C' be an optimal $(T_{v,i+1}, \mathcal{D}, c)$-coloring. The weight of the recoloring C' is then $w(C') = \mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c)$. Let $\mathcal{D}'_1 = C'[T_{v,i}] \setminus \{c\}$ be the set of colors different from c that C' uses in the recoloring of $T_{v,i}$, and let $\mathcal{D}'_2 = C'[T_{w_{i+1}}] \setminus \{c\}$ be the set of colors different from c that C' uses in the recoloring of $T_{w_{i+1}}$. Given the fact that $C'(v) = c$ and the convexity of $(C', T_{v,i+1})$, it follows that $\mathcal{D}'_1 \cap \mathcal{D}'_2 = \emptyset$. By the definitions of the sets $\mathcal{D}'_1$, $\mathcal{D}'_2$, and $\mathcal{D}$, it holds that $\mathcal{D}'_1 \cup \mathcal{D}'_2 = \mathcal{D} \setminus \{c\}$.

For a subtree T' of a tree T and a coloring C of T, let $C|_{T'}$ be the restriction of C to the vertices of T'. Since $C'|_{T_{v,i}}$ is a $(T_{v,i}, \mathcal{D}'_1, c)$-coloring of $T_{v,i}$ and $C'|_{T_{w_{i+1}}}$ is a $(T_{w_{i+1}}, \mathcal{D}'_2)$- or $(T_{w_{i+1}}, \mathcal{D}'_2, c)$-coloring of $T_{w_{i+1}}$, it holds that $w(C'|_{T_{v,i}}) \geq \mathrm{opt}_r(T_{v,i}, \mathcal{D}'_1 \cup \{c\}, c)$ and $w(C'|_{T_{w_{i+1}}}) \geq \mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}'_2, c)$. Consequently, $w(C')$ is at least the right-hand side of (3).

"$\leq$": Consider $\mathcal{D}'_1$ and $\mathcal{D}'_2$ with $\mathcal{D}'_1 \cup \mathcal{D}'_2 = \mathcal{D} \setminus \{c\}$ and $\mathcal{D}'_1 \cap \mathcal{D}'_2 = \emptyset$ such that the sum $\mathrm{opt}_r(T_{v,i}, \mathcal{D}'_1 \cup \{c\}, c) + \mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}'_2, c)$ is minimized. Denote with $C'_{v,i}$ the recoloring of $T_{v,i}$ witnessing the cost $\mathrm{opt}_r(T_{v,i}, \mathcal{D}'_1 \cup \{c\}, c)$ and with $C'_{w_{i+1}}$ the recoloring of $T_{w_{i+1}}$ witnessing the cost $\mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}'_2, c)$. We can then combine $C'_{v,i}$ and $C'_{w_{i+1}}$ to obtain a coloring C' for $T_{v,i+1}$. The weight of C' equals the right-hand side of (3). By construction, C' is a convex recoloring of $T_{v,i+1}$ that uses exactly the colors in $\mathcal{D}$ and has $C'(v) = c$. Thus, $w(C')$ is at least $\mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c)$. □

Theorem 1. *Non-uniformly weighted* Convex Recoloring *can be solved in $O(3^\kappa \cdot \kappa n)$ time for a tree with n vertices and κ colors.*

Proof. We have shown how to solve non-uniformly weighted Convex Recoloring by dynamic programming using the recurrences (1), (2), and (3). By visiting each vertex in postorder, it is possible to fill in opt, opt_c, and opt_r while only accessing already calculated entries. It remains to bound the running time. The bottleneck is clearly the calculation of (3). Since there are $O(n)$ edges in a tree, we have $O(n)$ values for the first component $T_{v,i+1}$. For fixed c and $T_{v,i+1}$, the computation of $\mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c)$ effectively needs to examine all 3-ordered partitions of $\mathcal{C} \setminus \{c\}$ of the form $(\mathcal{C} \setminus \mathcal{D}, \mathcal{D}_1, \mathcal{D}_2)$; there are $3^{\kappa-1}$ such partitions. In total, we arrive at the claimed running time. □

2.2 Uniformly Weighted Convex Recoloring

In this section, we show that for the uniformly weighted case, the parameter κ (number of colors) can be replaced by β (number of bad colors) in the running time of Theorem 1. This is particularly attractive for scenarios where the input is already almost convex. Moran and Snir [15] have shown how to get an $O(\Delta^\beta \cdot \beta n)$ time algorithm from their $O(\Delta^\kappa \cdot \kappa n)$ time dynamic programming algorithm for the non-uniformly weighted case. We show that analogously, our $O(3^\kappa \cdot \kappa n)$ time algorithm (Theorem 1) can be improved to $O(3^\beta \cdot \beta n)$ time for the non-uniformly weighted case. The approach is similar to that of Moran and Snir, but we considerably simplify some concepts and proofs.

When recoloring, typically good colors are overwritten by bad colors, in order to connect different regions of a bad color. It is tempting to just restrict the search of alternative colors to bad colors, which would reduce the size of the dynamic programming tables defined in Sect. 2.1 and give the desired speedup of replacing κ by β in the base of the exponential factor. However, this is not correct: sometimes a bad color has to be overwritten with a good color in order to wipe out a region of this bad color. The central observation of Moran and Snir [15] is that when overwriting a color with a good color, we do not have to decide immediately which good color to use—the goal is after all only to get rid of the bad color of the vertex that is being recolored. We capture this in the notion of a *restricted* recoloring, which is a coloring $V \to \mathcal{C} \cup \{*\}$, where $* \notin \mathcal{C}$ serves to mark vertices that are *uncolored*.[4] It is easy to see that a standard recoloring is convex iff all vertices on a path between two vertices with the same color c also have color c. In analogy, we say that a restricted recoloring is convex iff all vertices on a path between two vertices with the same color $c \neq *$ have color c.

In the uniform cost model, we can assign a cost to a restricted recoloring by simply giving cost $w(v)$ to the recoloring of $v \in V$ with $*$ (this is not possible in the non-uniform model, where cost also depends on the actual color used in recoloring a vertex). The following lemma shows that to find an optimal convex recoloring, it suffices to look for optimal restricted recolorings.

Lemma 2. *In the uniformly weighted model, any convex restricted recoloring can be converted in linear time into a convex recoloring of the same weight and vice versa.*

Proof. Given a convex restricted recoloring, we can fill in the colors of the uncolored vertices by a depth-first search starting from some not uncolored vertex, where we recolor an uncolored vertex with the color of its predecessor in the search. This clearly produces a convex recoloring with the same weight.

The only way a convex recoloring $\hat{C}$ of a coloring C might not already be a restricted recoloring is that some vertex color was overwritten with a good color. We construct C' from $\hat{C}$ by recoloring these vertices by $*$ instead. Clearly, C' has the same weight as $\hat{C}$, and we claim that C' is also convex. For this, consider two vertices v_1, v_2 with $C'(v_1) = C'(v_2) = c \neq *$. By construction of C', then also $\hat{C}(v_1) = \hat{C}(v_2) = c$. Thus, every vertex on the path between v_1 and v_2 is colored c by $\hat{C}$. If c is a bad color, then also every vertex on the path between v_1 and v_2 is colored c by C', since only good colors are used differently between $\hat{C}$ and C'; if c is a good color, then $\hat{C}$ has left v_1 and v_2 unchanged from C, and because c is a good color, any vertex between v_1 and v_2 must also be colored c by C, and thus also by $\hat{C}$ and C'. In summary, every vertex on the path between v_1 and v_2 is colored c by C', and thus C' is convex. □

By Lemma 2, it suffices to find the weight of an optimal restricted recoloring to solve the uniformly weighted Convex Recoloring problem. The dynamic programming from Sect. 2.1, based on the three tables opt, opt_r, and opt_c calculated by the recurrences (1), (2), and (3) in tree postorder remains almost unchanged. Therefore, we only point out the differences here.

Let $\mathcal{B}$ be the set of bad colors. The table opt now only covers restricted colorings, that is, $\text{opt}(T_v, \mathcal{D})$ is the weight of an optimal restricted coloring C' such that for each $c \in \mathcal{B}$ there is a vertex x in T_v with $C'(x) \neq C(x)$ and $C'(x) = c$ iff $c \in \mathcal{D}$. Table opt_r is adapted analogously, and we additionally allow $*$ as third argument. In the initialization, we also set $\text{opt}_r(T_v, \emptyset, *) = w(v)$

[4] Moran and Snir [15] use the more complicated notion of a *conservative recoloring*.

and $\text{opt}_r(T_v, \mathcal{D}, *) = \infty$ for all $\mathcal{D} \neq \emptyset$. For the case that $c = *$ in recurrence (3), we use

$$\text{opt}_r(T_{v,i+1}, \mathcal{D}, *) = \min_{\substack{\mathcal{D}_1 \cup \mathcal{D}_2 = \mathcal{D} \\ \mathcal{D}_1 \cap \mathcal{D}_2 = \emptyset}} (\text{opt}_r(T_{v,i}, \mathcal{D}_1, *) + \text{opt}(T_{w_{i+1}}, \mathcal{D}_2)). \quad (4)$$

We omit the proof, which is analogous to that in Sect. 2.1.

Theorem 2. *Uniformly weighted* Convex Recoloring *can be solved in* $O(3^\beta \cdot \beta n)$ *time for a tree with* n *vertices and* β *bad colors.*

3 Fast Subset Convolution

When we restrict the weights to be integers bounded by M, we can further improve the exponential part of the running time for Convex Recoloring by using *fast subset convolution.* This novel technique was developed by Björklund et al. [3], who used it to speed up several dynamic programming algorithms such as the classical Dreyfus–Wagner algorithm [10] for Steiner Tree in graphs. It was also used to improve the speed of subgraph homeomorphism algorithms [13].

Let f and g be functions defined on the power set of a finite set N with $|N| = p$, that is, $f, g : \mathcal{P}(N) \to I$. For any ring over I that defines addition and multiplication on elements of I, the *subset convolution* of f and g, denoted by $f * g$, is defined for each $S \subseteq N$ as

$$f * g : \mathcal{P}(N) \to I, \quad (f * g)(S) = \sum_{T \subseteq S} f(T) g(S \setminus T). \quad (5)$$

To calculate the subset convolution means to determine the value of $f * g$ for all 2^p possible inputs, assuming that f and g can be evaluated in constant time (typically by being stored in a table). A naive algorithm that calculates each value independently needs $O(\sum_{i=0}^{p} \binom{p}{i} 2^i) = O(3^p)$ ring operations. The following result shows a substantial improvement.

Theorem 3 (Björklund et al. [3]). *The subset convolution over an arbitrary ring can be computed with* $O(2^p \cdot p^2)$ *ring operations.*

Björklund et al. [3] showed how to apply Theorem 3 to also calculate the subset convolution for the integer min-sum semiring

$$f * g : \mathcal{P}(N) \to \mathbb{Z}, \quad (f * g)(S) = \min_{T \subseteq S} f(T) + g(S \setminus T) \quad (6)$$

by embedding it into the standard integer sum-product ring. Here, it is not appropriate to assume that addition and multiplication can be done in constant time, since the numbers involved can have up to n bits.[5] Björklund et al. [3] did not give a precise estimation, but it is not too hard to derive the following bound from their Theorem 3 [3].

Proposition 1. *The subset convolution over the integer min-sum ring with* $M := \max_{i \in (f(\mathcal{P}(N)) \cup g(\mathcal{P}(N)))} |i|$ *can be computed in* $O(2^p \cdot p^3 M \log^2(Mp))$ *time.*

[5] To avoid complicated terms, we assume a bound of $O(n \log^2 n)$ on the running time of integer multiplication of two n-bit numbers. Better bounds are known [12].

3.1 Convex Recoloring

We now use fast subset convolution over the integer min-sum to speed up the dynamic programming for Convex Recoloring. Recall that the bottleneck in deriving the running time of $O(3^\kappa \cdot \kappa n)$ (Theorem 1) comes from recurrence (3), which we recall here:

$$\mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c) = \min_{\substack{\mathcal{D}_1 \cup \mathcal{D}_2 = \mathcal{D} \setminus \{c\} \\ \mathcal{D}_1 \cap \mathcal{D}_2 = \emptyset}} (\mathrm{opt}_r(T_{v,i}, \mathcal{D}_1 \cup \{c\}, c) + \mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}_2, c)).$$

Consider fixed $T_{v,i+1}$ and c. Then (3) can be seen as a subset convolution over the integer min-sum semiring (like in (6)) by setting $f^c_{v,i}(\mathcal{D}) = \mathrm{opt}_r(T_{v,i}, \mathcal{D} \cup \{c\}, c)$ and $g^c_{w_i+1}(\mathcal{D}) = \mathrm{opt}_c(T_{w_{i+1}}, \mathcal{D}, c)$:

$$\mathrm{opt}_r(T_{v,i+1}, \mathcal{D}, c) = (f^c_{v,i} * g^c_{w_{i+1}})(\mathcal{D} \setminus \{c\}). \tag{7}$$

Theorem 4. *The non-uniformly weighted* Convex Recoloring *problem with integer weights bounded by M can be solved in $O(2^\kappa \cdot \kappa^4 n^2 M \log^2(nM))$ time for a tree with n vertices and κ colors.*

Proof. We solve non-uniformly weighted Convex Recoloring by dynamic programming using the recurrences (1), (2), and (3), where (3) is calculated by fast subset convolution as in (7). Using Proposition 1, for fixed $T_{v,i+1}$ and c, we can calculate (7) in $O(2^\kappa \cdot \kappa^3 nM \log^2(nM))$ time, because the values of $f^c_{v,i}$ and $g^c_{w_i+1}$ are bounded by nM, since they are weights of recolorings, and $\kappa \leq n$. The rest of the analysis is as in Theorem 1. ☐

In the same way as for Theorem 2, we obtain a running time of $O(2^\beta \cdot \beta^4 n^2 M \log^2(nM))$ for the uniformly weighted case.

Theorem 5. *The uniformly weighted* Convex Recoloring *problem with integer weights bounded by M can be solved in $O(2^\beta \cdot \beta^4 n^2 M \log^2(nM))$ time for a tree with n vertices and β bad colors.*

3.2 1-Connected Coloring Completion

Chor et al. [8] gave simple linear-time preprocessing rules that allow without loss of generality to assume $\kappa \leq k$, that is, there are at most as many colors as uncolored vertices. The data reduction also collapses each maximal connected monochromatic subgraph into a single vertex. Thus, the problem can be restated as finding a coloring of the set U of uncolored vertices such that each color induces a connected subgraph in U and each vertex in $V \setminus U$ with color c is adjacent to a vertex with color c in U.

Chor et al. [8] solved 1-Connected Coloring Completion by using a binary-valued dynamic programming table $T(\mathcal{C}', U')$ for $\mathcal{C}' \subseteq \mathcal{C}$ and $U' \subseteq U$ with the following semantics: $T(\mathcal{C}', U') = 1$ iff if it is possible to color U' with $\mathcal{C}'$ such that each color $c \in \mathcal{C}'$ induces a connected subgraph G_c in U that *dominates* the vertices colored c in $V \setminus U$, meaning that each such vertex is adjacent to at least one vertex in G_c. Thus, if $T(\mathcal{C}', U') = 0$, then it is not possible to solve the instance by assigning (exclusively) the colors from $\mathcal{C}'$ to the vertices in U'; but if $T(\mathcal{C}', U') = 1$, then solving the instance is still possible by finding a suitable allocation of the remaining colors $\mathcal{C} \setminus \mathcal{C}'$ to the remaining uncolored vertices $U \setminus U'$. Clearly, if $T(\mathcal{C}, U) = 1$, then the instance is solvable, and we can find the corresponding solution by backtracing.

Chor et al. [8] used the following recurrence to fill in T:

$$\begin{aligned} T(\mathcal{C}', U') = 1 \iff & \exists c \in \mathcal{C}', U'' \subset U' : T(\mathcal{C}' \setminus \{c\}, U'') = 1 \\ & \text{and } U' \setminus U'' \text{ induces a connected subgraph that} \\ & \text{dominates the vertices of color } c, \end{aligned} \tag{8}$$

which can be simplified to

$$T(\mathcal{C}', U') = \bigvee_{U'' \subseteq U'} \big(T(\mathcal{C}' \setminus \{c\}, U'') \wedge T(\{c\}, U' \setminus U'') \big) \tag{9}$$

for some $c \in \mathcal{C}'$. To be able to calculate recurrence (9), we need all values of $T(\{c\}, U')$ for $c \in \mathcal{C}$ and $U' \subseteq U$. The calculation can clearly be done in $O(2^k \cdot kn)$ time, since there are $2^k \cdot k$ such entries, and each can be calculated in linear time. A straightforward calculation of (9) for an entry then takes $O(2^k)$ time, and there are 4^k table entries, thus giving a total running time of $O(8^k + 2^k \cdot kn)$.

To speed up the exponential part of the calculation of (9), we use fast subset convolution over the or-and semiring.

Proposition 2. *The subset convolution over the or-and semiring*

$$f \circ g : \mathcal{P}(N) \to \{0, 1\}, \quad (f \circ g)(S) = \bigvee_{T \subseteq S} f(T) \wedge g(S \setminus T) \tag{10}$$

with $|N| = p$ *can be calculated in* $O(2^p \cdot p^3 \log^2 p)$ *time.*

Proof. It holds that

$$\begin{aligned} (f \circ g)(S) &= \bigvee_{T \subseteq S} f(T) \wedge g(S \setminus T) && (11) \\ &= \begin{cases} 1 & \text{if } \max_{T \subseteq S}(f(T) + g(S \setminus T)) = 2 \\ 0 & \text{otherwise} \end{cases} && (12) \\ &= \begin{cases} 1 & \text{if } (f * g)(S) = 2 \\ 0 & \text{otherwise,} \end{cases} && (13) \end{aligned}$$

and the subset convolution "$*$" in the integer min-sum semiring can be calculated using Proposition 1 with $M = 1$. □

Next, we define

$$\begin{aligned} f_{\mathcal{C}'}(U') &= T(\mathcal{C}' \setminus \{c\}, U') && (14) \\ g_{\mathcal{C}'}(U') &= T(\{c\}, U'), && (15) \end{aligned}$$

for some $c \in \mathcal{C}'$, which gives us

$$T(\mathcal{C}', U') = (f_{\mathcal{C}'} \circ g_{\mathcal{C}'})(U'). \tag{16}$$

Theorem 6. 1-Connected Coloring Completion *can be solved in* $O(4^k \cdot k^3 \log^2 k + 2^k \cdot kn)$ *time.*

Proof. By Proposition 2, for fixed $\mathcal{C}'$, we can calculate (16) in $O(2^k \cdot k^3 \log^2 k)$ time. There are 2^k subsets $\mathcal{C}' \subseteq \mathcal{C}$. Thus, together with the $O(2^k \cdot kn)$ time for the table initialization, we arrive at the claimed running time. □

4 Outlook

We improved known fixed-parameter tractability results based on dynamic programming for several NP-hard recoloring problems in trees and graphs. These problems are mainly motivated by applications in bioinformatics (particularly, phylogenetics). The running times now seeming practically feasible, so it would be desirable to experimentally test the algorithms on real-world data. In particular, it would be interesting to see how the improvements concerning the exponential factors that have been achieved due to fast subset convolution pay off in practice. Moreover, also the space consumption of our algorithms is exponential and so memory space could become the real bottleneck in applications—this invites further research on improvement strategies.

References

[1] E. H. Bachoore and H. L. Bodlaender. Convex recoloring of leaf-colored trees. In *Proc. 3rd ACiD*, volume 9 of *Texts in Algorithmics*, pages 19–33. College Publications, London, 2007. 3

[2] R. Bar-Yehuda, I. Feldman, and D. Rawitz. Improved approximation algorithm for convex recoloring of trees. *Theory of Computing Systems*, 2007. To appear. 1, 2

[3] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: fast subset convolution. In *Proc. 39th STOC*, pages 67–74. ACM, 2007. 3, 8

[4] C. Blum. Revisiting dynamic programming for finding optimal subtrees in trees. *European Journal of Operational Research*, 177(1):102–115, 2007. 3, 4, 5

[5] H. L. Bodlaender and M. Weyer. Convex and connected recolorings of trees and graphs. Unpublished manuscript, 2005. 2, 3, 4, 5

[6] H. L. Bodlaender, M. R. Fellows, M. A. Langston, M. A. Ragan, F. A. Rosamond, and M. Weyer. Kernelization for convex recoloring. In *Proc. 2nd ACiD*, volume 7 of *Texts in Algorithmics*, pages 23–35. College Publications, London, 2006. 2

[7] H. L. Bodlaender, M. R. Fellows, M. A. Langston, M. A. Ragan, F. A. Rosamond, and M. Weyer. Quadratic kernelization for convex recoloring of trees. In *Proc. 13th COCOON*, volume 4598 of *LNCS*, pages 86–96. Springer, 2007. 1, 2

[8] B. Chor, M. R. Fellows, M. A. Ragan, I. Razgon, F. A. Rosamond, and S. Snir. Connected coloring completion for general graphs: Algorithms and complexity. In *Proc. 13th COCOON*, volume 4598 of *LNCS*, pages 75–85. Springer, 2007. 1, 2, 3, 9, 10

[9] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999. 1

[10] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1(3):195–207, 1972. 8

[11] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. 1

[12] M. Fürer. Faster integer multiplication. In *Proc. 39th STOC*, pages 57–66. ACM, 2007. 8

[13] A. Lingas and M. Wahlen. On exact complexity of subgraph homeomorphism. In *Proc. 4th TAMC*, volume 4484 of *LNCS*, pages 256–261. Springer, 2007. 3, 8

[14] F. Maffioli. Finding a best subtree of a tree. Technical Report 91.041, Politecnico di Milano, Dipartimento di Elettronica, Italy, 1991. 3, 4, 5

[15] S. Moran and S. Snir. Convex recolorings of strings and trees: Definitions, hardness results and algorithms. In *Proc. 9th WADS*, volume 3608 of *LNCS*, pages 218–232. Springer, 2005. To appear in *Journal of Computer and System Sciences*. 1, 2, 4, 5, 6, 7

[16] S. Moran and S. Snir. Efficient approximation of convex recolorings. *Journal of Computer and System Sciences*, 73(7):1078–1089, 2007. 1, 2

[17] S. Moran, S. Snir, and W.-K. Sung. Partial convex recolorings of trees and galled networks: Tight upper and lower bounds. Manuscript, Feb. 2007. 2

[18] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Number 31 in Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006. 1

[19] O. Ponta. *The Fixed-Parameter Approach to the Convex Recoloring Problem*. Diplomarbeit, Mathematisches Institut, Ruprecht-Karls-Universität Heidelberg, 2007. 4

[20] I. Razgon. A $2^{O(k)}\text{poly}(n)$ algorithm for the parameterized convex recoloring problem. *Information Processing Letters*, 104(2):53–58, 2007. 1, 2