Quadratic Kernelization for Convex Recoloring of Trees

Hans L. Bodlaender

Michael R. Fellows

Michael A. Langston

Mark A. Ragan

Frances A. Rosamond

Mark Weyer

Department of Information and Computing Sciences,
Utrecht University

Technical Report UU-CS-2007-035

www.cs.uu.nl
ISSN: 0924-3275

Quadratic Kernelization for Convex Recoloring of Trees*

Hans L. Bodlaender[1], Michael R. Fellows[23], Michael A. Langston[34], Mark A. Ragan[35], Frances A. Rosamond[23], and Mark Weyer[6]

[1] Department of Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands
hansb@cs.uu.nl
[2] Parameterized Complexity Research Unit, Office of the DVC(Research),
University of Newcastle, Callaghan NSW 2308, Australia,
{michael.fellows,frances.rosamond}@newcastle.edu.au
[3] Australian Research Council Centre of Excellence in Bioinformatics
[4] Department of Computer Science, University of Tennessee, Knoxville TN 37996-3450 and Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6164 U.S.A. langston@cs.utk.edu
[5] Institute for Molecular Bioscience, University of Queensland, Brisbane, QLD 4072 Australia,
m.ragan@imb.uq.edu.au
[6] Institut für Informatik, Humboldt-Universität zu Berlin, Berlin, Germany
mark.weyer@informatik.hu-berlin.de

Abstract. The Convex Recoloring (CR) problem measures how far a tree of characters differs from exhibiting a so-called "perfect phylogeny". For an input consisting of a vertex-colored tree T, the problem is to determine whether recoloring at most k vertices can achieve a convex coloring, meaning by this a coloring where each color class induces a connected subtree. The problem was introduced by Moran and Snir, who showed that CR is NP-hard, and described a search-tree based FPT algorithm with a running time of $O(k(k/\log k)^k n^4)$. The Moran and Snir result did not provide any nontrivial kernelization. Subsequently, a kernelization with a large polynomial bound was established. Here we give the strongest FPT results to date on this problem: (1) We show that in polynomial time, a problem kernel of size $O(k^2)$ can be obtained, and (2) We prove that the problem can be solved in linear time for fixed k. The technique used to establish the second result appears to be of general interest and applicability for bounded treewidth problems.

1 Introduction

The historically first and most common definition of *fixed-parameter tractability* for parameterized problems is solvability in time $O(f(k)n^c)$, where n is the input size, k is the parameter, and c is a constant independent of n and k. Background and motivation for the general subject of parameterized complexity and algorithmics can be found in the books [9, 10, 15]. This basic view of the subject is by now well-known.

* This research has been supported by the Australian Research Council Centre of Excellence in Bioinformatics, by the U.S. National Science Foundation under grant CCR–0311500, by the U.S. National Institutes of Health under grants 1-P01-DA-015027-01, 5-U01-AA-013512-02 and 1-R01-MH-074460-01, by the U.S. Department of Energy under the EPSCoR Laboratory Partnership Program and by the European Commission under the Sixth Framework Programme. The first author was partially supported by BSIK grant 03018 (BRICKS: Basic Research in Informatics for Creating the Knowledge Society). The second and fifth authors have been supported by a Fellowship to the Institute of Advanced Studies at Durham University, and hosted by a William Best Fellowship to Grey College during the preparation of the paper.

Less well-known is the point of view that puts an emphasis on FPT kernelization. For an extensive overview of kernelization, see [12]. Kernelization is central to FPT as shown by the lemma:

Lemma 1. *A parameterized problem Π is in FPT if and only if there is a transformation from Π to itself, and a function g, that reduces an instance (x,k) to (x',k') such that:*

1. *the transformation runs in time polynomial in $|(x,k)|$,*
2. *(x,k) is a yes-instance of Π if and only if (x',k') is a yes-instance of Π,*
3. *$k' \leq k$, and*
4. *$|(x',k')| \leq g(k)$.*

The lemma is trivial, but codifies a shift of perspective. To see how this works, consider the Moran and Snir FPT result for Convex Recoloring, with the running time of $O^*(k/\log k)^k)$. When $n > (k/\log k)^k$, then the Moran and Snir algorithm runs in polynomial time, in fact, in time $O(n^5)$. (So we can run the algorithm and determine the answer, and "transform" the input to either a canonical small NO instance or a canonical small YES instance accordingly.) If $n \leq (k/\log k)^k$ then we simply do nothing; we declare that the input is "already" kernelized to the bound $g(k) = (k/\log k)^k$. If a parameterized problem is FPT, that is, solvable in time $f(k)n^c$, then the lemma above gives us the trivial P-time kernelization bound of $g(k) = f(k)$. Normally for FPT problems, $f(k)$ is some exponential function of k, so the subclasses of FPT

$$Lin(k) \subseteq Poly(k) \subseteq FPT$$

of parameterized problems that admit P-time kernelization to kernels of size bounded by *linear* or *polynomial* functions of k would seem to be severe restrictions of FPT.

It is surprising that so many parameterized problems in FPT belong to these much more restrictive subclasses. The connection to heuristics goes like this: *FPT kernelization* is basically a systematic approach to *polynomial-time* pre-processing. Preprocessing plays a powerful role in practical approaches to solving hard problems. For any parameterized problem in FPT there are now essentially two different algorithm design competitions with independent payoffs:

1. to find an FPT algorithm with the best possible exponential cost $f(k)$, and
2. to find the best possible polynomial-time kernelization for the problem.

A classic result on FPT kernelization is the Vertex Cover $2k$ kernelization due to Nemhauser and Trotter (see [15]). The linear kernelization for Planar Dominating Set is definitely nontrivial [1]. The question of whether the Feedback Vertex Set (FVS) problem for undirected graphs has a $Poly(k)$ kernelization was a noted open problem in parameterized algorithmics, only recently solved. The first $Poly(k)$ kernelization for FVS gave a bound of $g(k) = O(k^{11})$ [6]. Improved in stages, the best kernelization bound is now $O(k^3)$ [4].

We prove here a polynomial time kernelization for the Convex Recoloring problem, to a reduced instance that has $O(k^2)$ vertices. The basic problem is defined as follows.

Convex Recoloring
Instance: A tree $F = (V, E)$, a set of colors $\mathcal{C}$, a coloring function $\Gamma : V \to \mathcal{C}$, and a positive integer k.
Parameter: k
Question: Is it possible to modify Γ by changing the color of at most k vertices, so that the modified coloring Γ' is *convex* (meaning that each color class induces a single monochromatic subtree)?

The Convex Recoloring problem is of interest in the context of maximum parsimony approaches to evaluating phylogenetic trees [13, 11]. Some further applications in bioinformatics are also described in [13]; see also [14].

This paper is organized as follows. In Section 2, we give some preliminary definitions and results. Amongst others, we define a generalized version of the problem, Annotated Convex Recoloring. Sections Sections 3 and 4 deal with kernelization of this annotated version. In Section 3, we give a number of kernelization rules that, in polynomial time, transform an instance to an equivalent instance with $O(k)$ vertices per color. Further rules are given in Section 4, and these give us the kernel of size $O(k^2)$. Then, in Section 5, we discuss how the results can be transformed to a kernel of size $O(k^2)$ for Convex Recoloring. In Section 6, we prove that there is a linear time algorithm for Convex Recoloring for every fixed k, using an auxiliary graph of bounded treewidth and a formulation of the problem in Monadic Second Order Logic.

2 Preliminaries

In Sections 3 and 4, we give a kernelization algorithm for the following generalized problem. In Section 5, we show how we can transform back to a kernel for Convex Recoloring.

Annotated Convex Recoloring
Instance: A forest $F = (V, E)$ where the vertex set V is partitioned into two types of vertices, $V = V_0 \cup V_1$, a set of colors $\mathcal{C}$, a coloring function $\Gamma : V \to \mathcal{C}$, and a positive integer k.
Parameter: k
Question: Is it possible to modify Γ by changing the color of at most k vertices in V_1, so that the modified coloring Γ' is *convex*?

A *block* in a colored forest is a maximal set of vertices that induces a monochromatic subtree. For a color $c \in \mathcal{C}$, $\beta(\Gamma, c)$ denotes the number of blocks of color c with respect to the coloring function Γ. A color c is termed a *bad color* if $\beta(\Gamma, c) > 1$, and c is termed a *good color* if $\beta(\Gamma, c) = 1$.

A recoloring Γ' of coloring Γ is *expanding*, if for each block of Γ', there is at least one vertex in the block that keeps its color, i.e., that has $\Gamma'(v) = \Gamma(v)$. Moran and Snir [13] have shown that there always exists an optimal expanding convex recoloring. It is easy to see that this also holds for the variant where we allow annotations and a forest, as above.

As in [3], we define for each $v \in V$ a set of colors $S(v)$: $c \in S(v)$, if and only if there are two distinct vertices w, x, $w \neq v$, $x \neq v$, such that w and x have color c (in Γ), and v is on the path from w to x.

We introduce another special form of convex recoloring, called *normalized*. For each vertex $v \in V$, we add a new color c_v. We denote $\mathcal{C}' = \mathcal{C} \cup \{c_v \mid v \in V\}$. A recoloring $\Gamma' : V \to \mathcal{C}'$ of coloring $\Gamma : V \to \mathcal{C}$ is *normalized*, if for each $v \in V$: $\Gamma'(v) \in \{\Gamma(v), c_v\} \cup S(v)$.

Lemma 2. *There is an optimal convex recoloring that is normalized.*

Proof. Take an arbitrary optimal convex recoloring Γ'. Define Γ'' as follows. For each $v \in V$, if $\Gamma'(v) \in \{\Gamma(v)\} \cup S(v)$, then set $\Gamma''(v) = \Gamma'(v)$. Otherwise, set $\Gamma''(v) = c_v$. One can show that $\Gamma''(v)$ is convex. Clearly, it is normalized, and recolors at most as many vertices as $\Gamma'(v)$. □

3 Kernelizing to a Linear Number of Vertices Per Color

In this section, we give a set of rules that ensure that for each color, there are at most $O(k)$ vertices with that color. There are two trivial rules: if F is an empty forest and $k \geq 0$, then we return YES; if $k < 0$, then we return NO if and only if the given coloring is not already convex.

We start by showing that if there are more than $2k + 3$ vertices with a color c, then we can find a vertex with fixed color c.

The next lemma is a well known folklore theorem.

Lemma 3. *Suppose there are α vertices with color c. Then there is a vertex $v \in V$ such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c.*

Lemma 4. *Suppose there are $\alpha \geq 2k + 3$ vertices with color c. Let v be a vertex such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c. Then in any convex recoloring of F with at most k recolored vertices, v has color c.*

Proof. Suppose we have $\alpha \geq 2k + 3$ vertices with color c. Let v be a vertex such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c.

Consider a convex recoloring of F with at most k recolored vertices. Suppose v has a color different from c in this recoloring. Take a vertex $w \neq v$ that had color c in the original coloring. There are at least $\frac{\alpha}{2} \geq k + 1$ vertices of color c in the original coloring that do not belong to the component of $F - v$ that contains w. These cannot all have been recolored, so there is at least one vertex x with color c in the recoloring that is in a different connected component of $F - v$ as w. It now follows from the convexity that w must have a color different from c in the recoloring: as v has a color different from c, there cannot be a path of vertices with color c from w to x.

As this holds for each vertex $w \neq v$ that has color c in the original coloring, it implies that we recolored all vertices with color c: contradiction. So, v must have color c. □

Rule 1 *Suppose there are $\alpha \geq 2k+3$ vertices with color c. Let v be a vertex, such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c. Assume that $v \notin V_0$, or $\Gamma(v) \neq c$. Now*

- *If v has a color, different from c, and $v \in V_0$, then return NO.*
- *If v has a color, different from c, and $v \notin V_0$, then set $\Gamma(v) = c$ and decrease k by one.*
- *Fix the color of v: put $v \in V_0$.*

Standard techniques allow us to find all vertices for which Rule 1 can be applied in $O(kn)$ time. Soundness of Rule 1 follows directly from Lemma 4. Soundness of the next Rule 2 follows because we cannot recolor all vertices in this connected component, and there is no path from v to a vertex in this component.

Rule 2 *Suppose $v \in V_0$ has color c. Suppose one of the connected components of F contains at least $k+1$ vertices with color c, but does not contain v. Then return NO.*

Rule 3 *Let v be a vertex of color c. Suppose $F - v$ has a component with vertex set W that contains at least $k+1$ vertices with color c, that contains a neighbor w of v. Assume that $w \notin V_0$ or $\Gamma(w) \neq c$. Then*

- *If w has a color, different from c, and $w \in V_0$, then return NO.*
- *If w has a color, different from c, and $w \notin V_0$, then give w color c, and decrease k by one.*
- *Fix the color of w: put $w \in V_0$.*

Lemma 5. *Rule 3 is sound.*

Proof. We show that in any convex recoloring which gives v color c, and which recolors at most k vertices, w must have color c. This shows soundness.

Suppose we have a convex recoloring, with colors v with c, and which recolors at most k vertices. At least one vertex in W with color c is not recolored, say x. Now, w is on the path from v to x, and v and x have color c, so w must have color c. □

Rule 4 *If there is a tree T in the colored forest that contains two distinct vertices u and v, both of which have fixed color c, then modify T by contracting the path between u and v (resulting in a single vertex of fixed color c). If r denotes the number of vertices of this path that do not have color c, then $k' = k - r$.*

Rule 5 *If there are two distinct trees in the colored forest that both contain a vertex of fixed color c, then return NO.*

Soundness of Rule 4 follows because of the observation that because u and v have fixed color c, all vertices on the path from u to v must receive color c. Rule 5 is obviously sound.

Rules 4 and 5 ensure that for each color c, there is at most one vertex with fixed color c.

Let $v \in V_0$ have color c. A connected component of $F - v$ with vertex set W is said to be *irrelevant*, if both of the following two properties hold:

1. The vertices with color c in W form a connected set that contains a neighbor of v, and
2. For each color $c' \neq c$, if there are vertices with color c' in W, then c' is not bad.

If a component is not irrelevant, it is said to be *relevant*.

Lemma 6. *Let $v \in V_0$ and let W be the vertex set of an irrelevant connected component of $F - v$. Then there is a convex recoloring with a minimum number of recolored vertices that does not recolor a vertex in W.*

Proof. Let Γ' be a normalized convex recoloring of Γ with a minimum number of recolored vertices. For each $w \in W$, $S(w) = \{\Gamma(w)\}$, so $\Gamma'(w) \in \{\Gamma(w), c_w\}$. Now, the recoloring Γ'' with for all $x \notin W$, $\Gamma''(x) = \Gamma'(x)$, and for all $w \in W$, $\Gamma''(w) = \Gamma(w)$ is convex, and recolors the same number or less vertices than Γ'.

Lemma 6 shows that the following rule is sound.

Rule 6 *Let $v \in V_0$. Let W be the vertex set of an irrelevant connected component of $F - v$. Then remove W, and all edges incident to vertices in W, from the tree.*

Rule 7 *Suppose $F - v$ contains at least $2k + 1$ components, and each component of $F - v$ is relevant. Then return NO.*

Lemma 7. *Rule 7 is sound.*

Proof. To show this, we introduce a 'currency' with which we pay recolorings of vertices: a *har* is 'half-a-recoloring'. For a convex recoloring, we assign at most two hars to vertices. Suppose we recolor a vertex v from color c to color c'. Then one har is assigned to v, and one har is assigned to an arbitrary vertex which has color c' in the recoloring. (If no such vertex exists, the second har is not assigned.)

Suppose we have a convex recoloring with at most k recolored vertices. There must be a component of $F - v$ for which no vertex has assigned to it a har. Let W be the vertex set of this component. As the component was relevant, one of the following cases must hold.

- There is a vertex $w \in W$ with color c, and the vertices with color c do not form a connected set. Take two vertices with color c in W, say w and x in different blocks. All vertices on the path from w to x belong to W. Either w, x, or a vertex with a color different from c on the path from w to x must be recolored, and thus a har is assigned to W, contradiction.
- The vertices with color c in W form a connected set, but do not contain a neighbor of v. Similar to the previous case, all vertices in W with color c must be recolored, or all vertices on the path from v to vertices in W must get color c; in all cases, a har is assigned to a vertex in W.
- There is a vertex $w \in W$ with bad color $c' \neq c$. If w is recolored, it gets a har, contradiction. Otherwise, the block of color c' containing w is a subset of W (as v has a fixed color $\neq c'$). Other vertices with color c' must have been recolored, and thus, a har is given to a vertex in W with color c'.

As in each case, we obtained a contradiction, there does not exist a convex recoloring of F with at most k recolored vertices. □

We can observe that an instance that is reduced with respect to Rules 1 – Rule 7 has $O(k^2)$ vertices per color: if there are more than $2k+2$ vertices of color c, there is a vertex v with fixed color c, and each of the at most $2k$ components of $F - v$ can contain at most $2k$ vertices of color c. A further reduction of the number of vertices per color can be used by introducing new colors.

Rule 8 *Let $v \in V_0$ be a vertex with fixed color c. Let W be the vertex set of a relevant component of $F - v$ that contains vertices with color c. Suppose there are vertices with color c, not in $\{v\} \cup W$. Then*

- *Take a new color c', not yet used, and add it to $\mathcal{C}$.*
- *Take a new vertex v', and add it to F.*
- *Set $v' \in V_0$. Color v' with c'.*
- *For all $w \in W$ with color c, give w color c'. (Note that k is* not *changed.)*
- *For each edge $\{v, w\}$ with $w \in W$: remove this edge from F and add the edge $\{v', w\}$.*

So, basically we 'split' v into two copies; one with neighbors in W, and one with its other neighbors. For W and the second copy of v, v', we replace the color c by a new color. The idea is that the recolorings in the two parts do not influence each other with respect to what happens with color c, as v is a fixed vertex with color c separating these parts. Note that if W is in a different subtree of forest F as v, the new vertex v' is isolated.

Lemma 8. *Rule 8 is sound.*

Proof. Let F be the forest before applying the rule, and F' the forest after the rule.

Suppose we have a convex recoloring of F with at most k recolored vertices. Now, recolor the vertices in F' as follows: if a vertex gets a color $c'' \neq c$ in F, then recolor it in the same way in F'. If w gets color c, then color it with color c if $w \notin W$, and color it with color c' if $w \in W$. Color v' with c'. It is not hard to check that this is gives a convex coloring, again with at most k recolored vertices.

Suppose we have a convex recoloring of F' with at most k recolored vertices. Now, recolor all vertices in F as in this recoloring, but give all vertices with color c or c' in F' color c in F. This gives again a convex recoloring. □

Theorem 1. *An instance that is reduced with respect to the above Rules has at most $2k+2$ vertices per color.*

Proof. Consider a color c. If there is no vertex with fixed color c, then there are at most $2k+2$ vertices with color c, otherwise Rule 1 applies. If there is a vertex v with fixed color c, then at most one component of $F - v$ contains vertices with color c, otherwise, Rule 8 applies and decreases the number of vertices with color c. This component contains at most k vertices with color c (Rules 2, 3, and 4), so in this case, there are at most $k+1$ vertices with color c. □

4 Kernelizing to a Quadratic Number of Vertices

In this section, we work in a series of steps towards a kernel with $O(k^2)$ vertices. A number of new structural notions are introduced.

4.1 Bad colors

In several of the counting arguments in Section 4, we also use the notion of half-a-recoloring ('har'), also used in the proof of Lemma 7. Here, we assign hars to colors. If we recolor a vertex, we assign one har to its old color, and one har to its new color. As a bad color that does not get a har assigned to it remains bad, we have that the following rule is sound.

Rule 9 *If there are more than $2k$ bad colors, then return NO.*

The colors that are not bad are distinguished into three types: gluing, stitching, and irrelevant.

4.2 Gluing colors

A vertex v is *gluing* for color c, if it is adjacent to two distinct vertices x and y, both of color c, but the color of v is unequal to c. Note that if v is gluing for color c, then v separates (at least) two blocks of color c: x and y belong to different blocks. When we recolor v with color c, then these blocks become one block (are 'glued' together). A vertex v is *gluing*, if v is gluing for some color, and if the color of v is not bad. A color c is *gluing*, if there is a gluing vertex with color c, and c is not bad. (Note that the vertex will be gluing for some other color $c' \neq c$.)

For a color c, let W_c be the set of vertices that have color c or are gluing for color c. A *bunch* of vertices of color c is a maximal connected set of vertices in W_c, i.e., a maximal connected set of vertices that have either color c or are gluing for color c.

Lemma 9. *Suppose W is a bunch of color c that contains ℓ vertices that are gluing for color c. Then in any convex recoloring, at least ℓ vertices in W are recolored, and for each, either the old, or the new color is c.*

Proof. First, note that the bunch contains at least $\ell+1$ blocks of vertices of color c. Suppose we recolor less that ℓ vertices in W. Then, there is at least one block in W whose vertices are not recolored. Take an arbitrary vertex v_r from that block, and consider the rooted tree with vertex set W and v_r as root. Now, each gluing vertex v for color c in the bunch has a child w with color c. So, in a convex recoloring, we must either give v color c, or give w a color different from w. This gives at least ℓ recolored vertices. □

Thus, a bunch with ℓ gluing vertices implies that ℓ hars are assigned to color c. As a consequence, we have the soundness of the following rule.

Rule 10 *If there are more than $2k$ vertices that are gluing, then return NO.*

As a result, there are $O(k)$ colors that are gluing, and thus $O(k^2)$ vertices that have a gluing color. We next bound the number of bunches.

Lemma 10. *Suppose there are ℓ bunches with color c. Then the number of recolored vertices whose old or new color equals c, is at least $\ell - 1$.*

Proof. If each bunch contains a recolored vertex whose old or new color equals c, then the result clearly holds. Otherwise, take a vertex v that is not recolored whose color equals c.

Consider a bunch that does not contain v. If the bunch belongs to a different subtree as v, then all vertices with old color c in the bunch must be recolored. Now, suppose that the bunch belongs to the same subtree as v, but no vertex in the bunch is recolored with old or new color c. Let w be the first vertex outside the bunch on the path from the bunch to v. w must be recolored with new color c, and w is incident to a vertex with old and new color c in the bunch. Note that w cannot be incident to a vertex with old and new color c in another bunch (by the definition of bunch), and so can be associated uniquely to this bunch. Now the lemma follows. □

So, a color with ℓ bunches gets at least $\ell - 1$ hars assigned to it. Thus, the soundness of the following rule follows.

Rule 11 *Suppose that for each bad color c, there are ℓ_c bunches. If the sum over all bad colors c of $\ell_c - 1$ is at least $2k + 1$, then return NO.*

4.3 Stitching colors

We now define the notion of stitching vertices. To arrive at an $O(k^2)$ kernel, we have to define this notion with some care.

We first define the *cost* of a path between two vertices with the same color that belong to different bunches. Let v and w be two vertices with color c, in the same subtree of F, but in different bunches of color c. The *cost* of the path from v to w is the sum of the number of vertices with color unequal to c on this path, and the sum over all colors $c' \neq c$ such that

1. c' is not bad and c' is not gluing, and
2. there is at least one vertex with color c' on the path from v to w in T

of the following term: consider the blocks with color c' in the forest, obtained by removing the path from v to w from T. If one of these blocks contains a vertex in V_0 (i.e., it has fixed color c'), then sum the sizes of all other blocks. Otherwise, sum the sizes of the all blocks except one block with the largest number of vertices.

A *stitch* of color c is a path between two vertices v, w, both of color c, such that:

1. v and w are in different bunches,
2. all vertices except v and w have a color, different from c, and do not belong to V_0, and
3. the cost of the path is at most k.

A vertex v is *stitching* for color c, if:

1. the color of v is different from c,
2. v is not gluing for color c, and
3. v is on a stitch of color c.

A vertex is *stitching*, if it is stitching for at least one color. (Note that this vertex will be stitching for a bad color.) A color c is *stitching* if there is at least one vertex with color c that is stitching, and it is not bad or gluing. A color is *irrelevant*, if it is neither bad, gluing, or stitching. Summarizing, we have the following types of vertices:

1. vertices with a bad color,
2. vertices with a gluing color,
3. vertices with a stitching color that belong to a stitch,
4. vertices with a stitching color, that do not belong to a stitch — these will be partitioned into pieces in Section 4.4, and
5. vertices with an irrelevant color.

Lemma 11. *Suppose there is a convex recoloring with at most k recolored vertices. Then there is an optimal convex recoloring that is normalized such that for each bad color c, all vertices that receive color c have color c in the original coloring or are gluing for c or are stitching for c.*

Proof. Suppose we have an optimal convex recoloring Γ' with at most k recolored vertices. We assume that Γ' is normalized (cf. Lemma 2). From all such possible Γ', take one with a maximum number of vertices v with $\Gamma'(v) = c_v$.

Consider a bad color c. Let V_c be the set of vertices that have color c and are not recolored, i.e., $V_c = \{v \mid \Gamma'(v) = \Gamma(v) = c\}$.

Claim. Each vertex with color c in the recoloring Γ' belongs to V_c or is on a path between two vertices in V_c; all vertices on this path have color c in the recoloring.

Proof. Suppose not. Take a vertex v which is recolored, received color c, and is not on a path between two vertices in V_c. Let W be the set of vertices w in F, with the property that w has color c in the recoloring, and each path from w to a vertex in V_c passes through v. By assumption, we have that $v \in W$. Now, if we recolor all vertices in W by a color of the form c_w, i.e., change Γ' by setting for each $w \in W$ the color of w to c_w, we obtain a recoloring with the same number of recolored vertices as Γ'. This recoloring is also convex, by the construction of W, but has a larger number of vertices that are mapped to their color c_w, contradiction. □

Now consider a vertex v which has color c in Γ'. Suppose v does not have color c in Γ. Then, by the previous claim, there must be two vertices w and $x \in V_c$ such that v is on the path from w to x.

Consider the path from w to x. Let w' be the last vertex in V_c on this path before v, and let x' be the first vertex in V_c on this path after v. Consider the path from w' to x'. If w' and x' are in the same bunch, then v is the only vertex between w' and x' on the path, i.e., v is gluing for c. So, assume w' and x' belong to different bunches.

To end the proof of this lemma, we will show that the cost of the path from w' to x' is at most k, as this implies that this path is a stitch, and hence v is stitching for c.

Note that each vertex on the path from w' to x' has a color, different from c in Γ, but is recolored to c. Consider a vertex y on this path. Suppose its original color is c'. Suppose $W_1, \ldots, W_r$ are the blocks of color c', of the forest, obtained by removing the path from w' to x' from F. At most one of these sets can contain vertices with color c' in the recoloring, as the removed vertices must have a color, different from c'. So, the number of vertices with color c' that are recolored is at least the sum over all sizes of the sets $W_1, \ldots, W_r$, except for the largest of these sets in case no set contains a vertex with fixed color, or the sum over all sizes of these sets except the one that contains a vertex with fixed color, plus the number of vertices with color c' on the path from w' to x'. As this holds for all vertices on the path from w' to x' and their original colors, we have that the number of vertices that is recolored is at least the cost of the path. So, the path has cost at most k, and hence is a stitch. □

Lemma 12. *Suppose there is a convex recoloring with at most k recolored vertices. Then there is an optimal convex recoloring such that no vertex with an irrelevant color is recolored.*

Proof. Again, assume we have a normalized recoloring Γ' with at most k recolored vertices. Using the construction of the previous lemma, we may assume that for each bad color c, all vertices with color c in the recoloring, have color c in the original color, or are gluing or stitching for c.

Consider an irrelevant color c. As no vertex with color c is on a stitch, we have, by the previous lemma, that no vertex with color c is recolored to a bad color. As the recoloring is normalized, all recolored vertices v with color c receive the color c_v.

Suppose we recolor at least one vertex with color c. Now construct a recoloring Γ'' as follows: for all vertices v, with $\Gamma(v) \neq c$, set $\Gamma''(v) = \Gamma'(v)$, and for all v with $\Gamma(v) = c$, set $\Gamma''(v) = c$. Γ'' is convex (simple case analysis shows this), but recolors fewer vertices than Γ' does, contradiction. □

From Lemma 12, it follows that we can fix the color of vertices with an irrelevant color, and thus that the following rule is sound.

Rule 12 *Let $v \in V_1$ be a vertex with an irrelevant color. Put v into V_0 (i.e., fix the color of v.)*

The following rule, in combination with earlier rules, helps us to get a reduced instance without any vertex with an irrelevant color.

Rule 13 *Let $v \in V_0$ be the only vertex with some color c. Then remove v and its incident edges.*

Soundness is easy to see: as v has a fixed color, the same recolorings in the graph with and without v are convex. The combination of Rules 4, 12, and 13 causes the deletion of

all vertices with an irrelevant color: all such vertices first get a fixed color, then all vertices with the same irrelevant color are contracted to one vertex, and then this vertex is deleted. So, we can also use instead the following rule.

Rule 14 *Suppose c is an irrelevant color. Then remove all vertices with color c.*

4.4 Pieces of a stitching color

For a kernel of size $O(k^2)$, we still can have too many vertices with a stitching color. In order to reduce the number of such vertices we introduce the concept of a *piece of color c*. Consider a stitching color c, and consider the subforest of the forest, induced by the vertices with color c. If we remove from this subtree all vertices that are on a stitch, then the resulting components are the pieces, i.e., a *piece of color c* is a maximal subtree of vertices with color c that do not belong to a stitch. Assume that we have exhaustively applied all of the rules described so far, and therefore we do not have vertices with an irrelevant color.

Lemma 13. *Suppose W is the vertex set of a piece of stitching color c. Suppose there is a vertex $v \in W \cap V_0$. Then if there is a convex recoloring with at most k recolored vertices, then there is a convex recoloring that does not recolor any vertex in W.*

Proof. Suppose there is a convex recoloring Γ' with at most k recolored vertices. We can assume that there are no vertices with an irrelevant color, that Γ' is normalized, and that for each bad color c, the vertices v with $\Gamma'(v) = c$, either have $\Gamma(v) = c$, or are gluing or stitching for c. We also assume Γ' recolors a minimum number of vertices.

Suppose now that W is the vertex set of piece of stitching color c, and that $v \in W \cap V_0$. We claim that for all $w \in W$, $\Gamma'(w) = \Gamma(w) = c$. Suppose not. Take a vertex $w \in W$ with $\Gamma'(w) \neq \Gamma(w) = c$. Consider the path from w to v. Let x be the last vertex on this path with $\Gamma'(x) \neq c$. As $v \in V_0$, $\Gamma'(v) = c$, so $x \neq v$, and x is adjacent to a vertex y with $\Gamma'(y) = c$.

As x belongs to a piece, it is not stitching or gluing, and hence we have that $\Gamma'(x) = c_x$. Thus, if we change the color of x in Γ' to c, we still have a convex recoloring (c still has one block, and there are no other vertices with the same color as x in Γ'). Hence, there is a convex recoloring with fewer recolored vertices as Γ', contradiction. Thus, for all $w \in W$, $\Gamma'(w) = \Gamma(w)$. □

Lemma 13 shows directly that the following rule is sound.

Rule 15 *Let W be a vertex set of a piece of stitching color c, that contains at least one vertex in V_0. Then put all of the vertices of W into V_0.*

As a result of this rule and Rule 4, a piece containing a vertex with a fixed color will contain only one vertex. If there is a large piece, found by Rule 16, then it will be contracted to a single vertex by Rule 4.

Rule 16 *Suppose c is a stitching color, and there are α vertices with color c. Suppose there is no vertex in V_0 with color c. If W is the vertex set of a piece of color c, and $|W| > \alpha/2$, then put all of the vertices of W into V_0.*

Lemma 14. *Rule 16 is sound.*

Proof. We will show that if there is a convex recoloring with at most k recolored vertices, then there is one that does not recolor any vertex in W. Soundness then directly follows.

Let c, α, W be as in the rule. Suppose there is a convex recoloring Γ' with at most k recolored vertices. As before, we can assume that there are no vertices with an irrelevant color, that Γ' is normalized, and that for each bad color c, the vertices v with $\Gamma'(v) = c$, either have $\Gamma(v) = c$, or are gluing or stitching for c. We also assume Γ' recolors a minimum number of vertices.

Let $Q = \{v \in V \mid \Gamma(v) = c$ be the vertices with color c. Let X be the vertices in Q that are recolored, and $Y = Q - X$ be the vertices in X that are not recolored. Let $Z \subseteq X$ be the vertices in X that are recolored to a bad color. Note that vertices in Z must be gluing or stitching for some bad color.

As c is stitching (and hence not bad), Q induces a subtree. Each tree in the forest, obtained by removing the vertices from Z from Q is called a *superpiece*. Each piece of color c is contained in a superpiece: a piece does not contain stitching vertices, and hence no vertices recolored to a bad color.

In particular, the piece with vertex set W is contained in a superpiece, say with vertex set W'. Now, change Γ' to recoloring Γ'' in the following way: for all $v \in V - W$ and for all $v \in Z$, set $\Gamma''(v) = \Gamma'(v)$. For all $v \in Q - Z - W'$, set $\Gamma''(v) = c_v$, and for all $v \in W'$, set $\Gamma''(v) = c = \Gamma(v)$. In other words, we recolor all vertices in superpieces, except W' to the color of the form c_v, and do not recolor the vertices in superpiece W'.

Γ'' is convex. As all vertices v in W' have $\Gamma'(v) \in \{c, c_v\}$, and c_v is a color, that can only be given to v, the change from Γ' to Γ'' cannot increase the number of blocks for any color $\neq c$. Also, as W' induces a subtree, there is only one block with color c.

Γ'' recolors at most as many vertices as Γ'. To show this, we compare the number of vertices in Q which are recolored: note that if $\Gamma(v) \neq c$, then $\Gamma'(v) = \Gamma''(v)$, so we only need to analyze which vertices in Q are recolored. We consider two cases.

The first case is that there is a vertex v in W' which is not recolored by Γ'. Then, any vertex $w \in Q$ that is recolored by Γ'' either is recolored to a bad color (i.e., it belongs to Z and $\Gamma''(w) = \Gamma'(w)$), or belongs to another superpiece. In the latter case, the path from v to w contains a vertex z in Z, and now $\Gamma'(z) \neq c$ implies that $\Gamma'(w) \neq c$. So, all vertices recolored by Γ'' are also recolored by Γ'.

In the second case, all vertices in W' are recolored by Γ'. The number of vertices in Q, recolored by Γ' is hence at least $|W'| \geq |W| > \alpha/2$. The number of vertices in Q, recolored by Γ'' equals $|Q - W'| = |Q| - |W'| < \alpha - \alpha/2$. So Γ' recolors more vertices that Γ'', contradicting the minimality of Γ'.

As Γ'' is a convex recoloring which recolors at most as many vertices as Γ', and does not recolor any vertex in $W' \supseteq W$, the result follows. □

4.5 Kernel size

We now *tag* some vertices that have a stitching color. A tag is labeled with a bad color. Basically, when we have a stitch for a bad color c, we tag the vertices that count for the cost of the stitch with color c. Tagging is done as follows. For each stitch for bad color c, and for each stitching color c' with a vertex on the stitch with color c', consider the blocks of color c' obtained by removing the vertices on the stitch. If there is no vertex with color c' in V_0, then take a block with vertex set W such that the number of vertices in this piece is at least as large as the number of vertices in any other piece. Tag all vertices in $Q - W$ with color c. If there is a vertex v with color c' in V_0, then tag all vertices with color c', except those that are in the same block as v.

Comparing the tagging procedure with the definition of the cost of a stitch, we directly note that the number of vertices that is tagged equals the cost of the stitch. In particular, this implies that for each stitch of bad color c, we tag at most k vertices with c.

We now want to count the number of vertices with a stitching color. To do so, we first count the number of vertices with a stitching color that are tagged. To do this, we consider a bad color c, and count the number of vertices, with a stitching color, that are tagged with c.

Lemma 15. *Let c be a bad color with ℓ_c bunches. A reduced instance has at most $2k(\ell_c - 1)$ vertices that are tagged with c.*

Proof. If there is only one bunch with color c, then there are no stitches for c, and hence there are no vertices, tagged with c. So, in the remainder of the proof, we assume $\ell_c \geq 2$.

We define the *bunch-stitch* graph for color c as the graph, formed as follows. Take F. Contract each bunch of color c to a single vertex. Call these *bunch vertices*. Now, remove all vertices and incident edges that are not a bunch vertex or stitching for color c. Clearly, the bunch graph is a forest.

A *branch vertex* is a stitching vertex in the bunch-stitch graph of degree at least three. As each leaf of the bunch-stitch graph is a bunch vertex, there are at most ℓ_c leaves in the bunch-stitch graph, and hence there are at most $\ell_c - 1$ branch vertices. A path in the bunch-stitch graph such that both endpoints are a bunch vertex or a branch vertex, and all inner vertices are stitching vertices of degree two in the bunch-stitch graph is called a *thread*.

Note that there are at most $2\ell_c - 2$ threads. This can be seen as follows. If we contract all stitching vertex of degree two to a neighbor in the bunch-stitch graph, we obtain a forest where each edge corresponds to a thread. The vertices of the forest are the bunch and branch vertices, and hence there are at most $2\ell_c - 1$ vertices, so at most $2\ell_c - 2$ edges and hence threads.

We now view threads as paths in F, by replacing bunch vertices by a vertex in the corresponding bunch (adjacent to the next stitching vertex in the thread). We define the cost of a thread similar to the cost of a stitch, as follows. The *cost* of a thread from v to w is defined as the sum of the number of vertices with color unequal to c on this path, and the sum over all colors $c' \neq c$ such that

– c' is not bad and c' is not gluing, and
– there is at least one vertex with color c' on the path from v to w in T

of the following term: consider the sizes of the blocks with color c' in the forest, obtained by removing the path from v to w from T, and sum all these sizes except the largest when no block contains a vertex in V_0, or except the block that contains a vertex in V_0, if there is such a block.

For each thread, the cost is at most k, as each thread is contained in a stitch. So, the total cost of all threads is at most $2k(\ell_c - 1)$.

Finally, note that each vertex that is tagged with color c counts for the cost of at least one of the threads. So, the number of vertices, tagged with c is at most the sum of the costs of the threads, and hence at most $2k(\ell_c - 1)$. □

Lemma 16. *Let c be a stitching color. There is at most one piece of color c that contains a vertex that is not tagged with a bad color.*

Proof. Suppose v has stitching color c and is not tagged with a bad color. As all vertices on a stitch are tagged, v belongs to a piece of color c. Consider a vertex w in another piece. By the definition of piece, there is a vertex on a stitch on the path from v to w. As this stitch does not tag v, w is tagged by the stitch. □

Lemma 17. *In a reduced instance, there are at most $8k^2$ vertices with a stitching color.*

Proof. Write again ℓ_c for the number of bunches of bad color c. As the sum over all bad colors of $\ell_c - 1$ is at most $2k$, there are at most $4k^2$ vertices with a stitching color that are tagged.

For each stitching color c', there is at most one piece with vertices that are not tagged. Either this piece contains one vertex with fixed color, or no vertices with fixed color, but in the latter case, this piece contains at most half the number of vertices with color c'. Hence there are as most as many untagged vertices with color c' as tagged vertices with color c'. So, there are at most $4k^2$ vertices with color c' that are not tagged. □

Theorem 2. *In time polynomial in n and k, a kernel for* Annotated Convex Recoloring *can be found with $O(k^2)$ vertices.*

Proof. With standard techniques, one can observe that all rules can be carried out in time polynomial in n and k.

In the reduced instance, there are at most $2k$ bad colors, and at most $2k$ gluing colors. For each of these colors, there are at most $2k + 2$ vertices with that color. So, in total at most $4k^2 + 4k$ vertices have a bad or gluing color. There are at most $8k^2$ vertices with a stitching color, and no vertices with an irrelevant color, so in total we have at most $12k^2 + 4k$ vertices. □

5 A Kernel for Convex Recoloring

In the previous section, we have seen how to obtain a kernel with $O(k^2)$ vertices for the Annotated Convex Recoloring problem. In this section, we discuss how this can be modified to a kernel for Convex Recoloring, i.e., with some final steps, we can ensure that we do not have annotations (vertices with a fixed color), and that we have a tree (instead of a forest).

Lemma 18. *Let $v \in V_1$ be a vertex which has at least $k+1$ neighbors with the same color. Then, v is not recolored in a convex recoloring with at most k recolored vertices.*

Proof. Suppose v is recolored. Then, at least two neighbors w_1, w_2 of v are not recolored, but now the resulting coloring is not convex, as the path from w_1 to w_2 uses in the recoloring a vertex with a color, different from the color of w_1 and w_2. This is a contradiction. □

With Lemma 18, we can easily undo annotations: we add to each vertex $v \in V_0$, $k+1$ new neighbors with the same color as v, and then put v in V_1. This step, however, can possible yield more than a quadratic number of vertices. To obtain a quadratic kernel without annotated vertices, we add two new rules.

Rule 17 *Let $v \in V_0$ be a vertex with fixed color c. Let w be a neighbor of v, and suppose that the vertex set of the component of $F - v$ that contains w does not contain a vertex with color c, i.e., $c \notin S(w)$. Then remove the edge $\{v, w\}$ from F.*

Lemma 19. *Rule 17 is sound.*

Proof. Let F be the forest before applying the rule, and F' the forest after applying the rule. It is easy to see that each normalized convex recoloring of F that does not recolor vertices in V_0 is a normalized convex recoloring of F' that does not recolor vertices in V_0, and vice versa. □

Proposition 1. *If Rules 8 and 17 cannot be applied, then each vertex in V_0 has degree at most one.*

Proof. If $v \in V_0$ has degree at least two. Let T be the tree in F that contains v. If at least two subtrees of $T - v$ contain vertices with the same color as v, then Rule 8 can be applied. Otherwise, there is a subtree of $T - v$ that does not contain a vertex with the same color as v, and the edge from v to that subtree can be removed by Rule 17. □

We now can undo annotations in the following way.

- Suppose $v \in V_0$ has color c, and suppose there are α vertices with color c.
- If c is a broken color, then add $k+1$ new vertices with color c, and make these incident to v.
- If c is not a broken color, then add $\min\{k+1, \alpha - 1\}$ new vertices with color c, and make these incident to v.
- Unfix the color of v, i.e., put v in V_1.

Suppose F' is obtained from F by applying the step described above. Clearly, if there is a convex recoloring of F with at most k recolored vertices (not recoloring vertices in V_0), there is one of F'.

Now, suppose we have a convex recoloring Γ' of F' with at most k recolored vertices, that does not recolor vertices in V_0. We can assume Γ' is normalized. If we added $k+1$ neighbors with color c, then by Lemma 18, v is not recolored. Suppose c is not a broken color, and we added $\alpha - 1 < k+1$ neighbors of v with color c. We must have recolored at least $\alpha - 2$ of the new neighbors of v. Now, change Γ' as follows. For each w with $\Gamma'(w) = c$, give w the color c_w. As c was not broken, and Γ' is normalized, these vertices had color c in the original coloring, so we recolor here at most $\alpha - 1$ vertices. Then, give v and all its new neighbors color c: these are at least $\alpha - 1$ vertices that are no longer recolored. We obtain a convex recoloring Γ'' that recolors at most as many vertices as Γ', and Γ'' does not recolor v.

So, the steps described above allow us to undo all annotations. As there are $O(k)$ broken colors, we add $O(k^2)$ new vertices with a broken color, and for all other colors, we add at most as many new vertices as there were old vertices with such a color. Thus, we still have a kernel with $O(k^2)$ vertices.

With one simple final step, we can turn the forest into a tree. Take $k+2$ new vertices $z_0, z_1, \ldots, z_{k+1}$ with a new color c^*. Make $z_1, \ldots, z_{k+1}$ adjacent to z_0, and make z_0 adjacent to an arbitrary vertex in each tree in F. By Lemma 18, we cannot recolor z_0, and thus, the step gives an equivalent instance with a tree instead of a forest.

Theorem 3. *A kernel for* Convex Recoloring *can be obtained in time, polynomial in n and k.*

6 Linear Time FPT with Treewidth and MSOL

We describe an algorithm that solves the Convex Recoloring problem in $O(f(k) \cdot n)$ time. There are four main ingredients of the algorithm:

- the construction of an auxiliary graph, the *vertex-color* graph,
- the observation that for yes-instances of the Convex Recoloring problem this graph have bounded treewidth,
- a formulation of the Convex Recoloring problem for fixed k in Monadic Second Order Logic, and
- the use of a famous result of Courcelle [8], see also [2, 5].

For ease of presentation, we assume that there are no vertices with a fixed color, and that the input is a tree. The "trick" of using such an auxiliary graph seems to be widely applicable. For another example of this new technique, see [7].

Suppose we have a tree $T = (V, E)$, and a coloring $\Gamma : V \to \mathcal{C}$. The *vertex-color graph* has as vertex set $V \cup \mathcal{C}$, i.e., we have a vertex for each vertex in T, and a vertex for each color. The edge set of the vertex-color graph is $E \cup \{\{v, c\} \mid \Gamma(v) = c\}$, i.e., there are two types of edges: the edges of the tree T, and an edge between a vertex of T and

the vertex representing its color. Recall that $S(v)$ denotes the set of colors c for which there are vertices w and x, distinct from v, with v on the path from w to x in T, and $\Gamma(w) = \Gamma(x) = c$.

Lemma 20. *Suppose there is a convex recoloring with at most k recolored vertices. Then for each $v \in V$, $|S(v)| \leq k + 1$.*

Proof. Suppose Γ' is a convex recoloring with at most k recolored vertices of Γ. Consider a vertex $v \in V$, and a color $c \in S(v)$. Suppose $c \neq \Gamma'(v)$. There are vertices w, x with $\Gamma(w) = \Gamma(x) = c$, and the path from w to x uses v. As v is not recolored to c, either w or x must be recolored. So, there can be at most k colors $\neq \Gamma'(v)$ that belong to $S(v)$. □

Lemma 21. *Suppose there is a convex recoloring with at most k recolored vertices. Then the treewidth of the vertex-color graph is at most $k + 3$.*

Proof. Without loss of generality, we suppose that Γ is surjective, i.e., for all $c \in \mathcal{C}$, there is a $v \in V$ with $\Gamma(v) = c$. (If Γ is not surjective, then colors c with $\Gamma^{-1}(c) = \emptyset$ are isolated vertices in the vertex-color graph, and removing these does not change the treewidth.)

Take an arbitrary vertex $r \in V$ as root of T. For each $v \in V$, set $X_v = \{v, p(v), \Gamma(v)\} \cup S(v)$, where $p(v)$ is the parent of v. For $v = r$, we take $X_v = \{v, \Gamma(v)\} \cup S(v)$.

It is easy to verify directly that $(\{X_v \mid v \in V\}, T)$ is a tree decomposition of the vertex-color graph of width at most $k + 3$. First, note that $\bigcup_{v \in V} X_v = V \cup \mathcal{C}$, and that for all edges e in the vertex-color graph, there is a $v \in V$ with X_v containing both endpoints of e. For each $v \in V$, the set $\{w \in V \mid v \in X_w\}$ consists of v and its children in T. For each $c \in \mathcal{C}$, the set of all vertices on a path between two vertices of color c is connected, and hence $\{w \in V \mid c \in X_w\}$ is connected. Clearly, each set X_v has size at most $k + 4$, so the treewidth is at most $k + 3$. □

Theorem 4. *For each fixed k, there is a linear time algorithm that, given a vertex-colored tree T, decides if there is a convex recoloring of T with at most k recolored vertices.*

Proof. We show that for a fixed number of recolored vertices k, the Convex Recoloring problem can be formulated as a property of the vertex-color graph that can be stated in Monadic Second Order Logic. The result then directly follows by the result of Courcelle [8], that each such property can be decided in linear time on graphs with bounded treewidth. (See also [2, 5]. We use Lemma 21.)

We assume we are given the vertex-color graph $(V \cup \mathcal{C}, E')$, and have sets V and $\mathcal{C}$ distinguished. A recoloring Γ' that differs from Γ at most k vertices can be represented by these vertices $v_1, \ldots, v_k$ and by the corresponding values $c_1 := \Gamma'(v_1), \ldots, c_k := \Gamma'(v_k)$. Then, for a given color c, consider the set V_c defined as follows:

- If $v = v_i$ for some $1 \leq i \leq k$, then $v \in V_c$ if and only if $c_i = c$.
- If $v \notin \{v_1, \ldots, v_k\}$, then $v \in V_c$ if and only if (v, c) is an edge of the vertex-color graph.

Note, that V_c is the color class of c in the coloring Γ'. Hence Γ' is convex if and only if V_c is connected for each $c \in \mathcal{C}$. It is a standard fact, that MSOL can express connectedness, say by a formula $\varphi(X)$, where the set variable X represents V_c. Furthermore, the

definition of V_c given above can be expressed even in first-order logic, say by a formula $\psi(x_1, \ldots, x_k, y_1, \ldots, y_k, z, X)$, where additionally $x_1, \ldots, x_k$ represent $v_1, \ldots v_k$, $y_1, \ldots, y_k$ represent $c_1, \ldots c_k$, and z represents c. Then, for this k, the Convex recoloring problem can be expressed by the formula

$$\exists x_1, \ldots, x_k : \exists y_1, \ldots, y_k :$$
$$\left(\bigwedge_{1 \le i \le k} x_i \in V \wedge \bigwedge_{1 \le i < j \le k} x_i \neq x_j \wedge \bigwedge_{1 \le i \le k} y_i \in \mathcal{C} \ \wedge \forall z : \forall X : (\psi \rightarrow \varphi) \right).$$

The algorithms works as follows. Given a colored tree (T, Γ), it constructs the vertex-color graph and computes a tree-decomposition of width at most $k+3$. If this fails, the algorithm returns NO, in accordance with Lemma 21. Otherwise, using the tree-decomposition, it evaluates the above formula on the vertex-color graph. Each step uses at most linear time. □

7 Discussion and Open Problems

An obvious question is: does Convex Recoloring (for trees) admit a linear kernel?

Techniques to answer such questions are a matter of intense investigation, particularly because polynomial-time data reduction / kernelization is so useful in practical terms.

The results here are based on generalizing the problem to an annotated form (a strategy that has proved effective for many problems), and it is reassuring that the initial $Poly(k)$ kernelization with a bound of $O(k^{28})$ (based on twelve reduction rules, many different from those employed here), is improved here to $O(k^2)$, based on seventeen reduction rules, and a deeper attention to the structure associated to the parameter.

References

1. J. Alber, M. R. Fellows, and R. Niedermeier. Polynomial-time data reduction for dominating sets. *J. ACM*, 51:363–384, 2004.
2. S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12:308–340, 1991.
3. R. Bar-Yehuda, I. Feldman, and D. Rawitz. Improved approximation algorithm for convex recoloring of trees. In *Proceedings Third Workshop on Approximation and Online Algorithms WAOA 2005*, pages 55–68, 2005.
4. H. L. Bodlaender. A cubic kernel for feedback vertex set. In W. Thomas and P. Well, editors, *Proceedings 24th International Symposium on Theoretical Aspects of Computer Science, STACS 2007*, pages 320–331. Springer Verlag, Lecture Notes in Computer Science, vol. 4393, 2007.
5. R. B. Borie, R. G. Parker, and C. A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7:555–581, 1992.
6. K. Burrage, V. Estivill-Castro, M. R. Fellows, M. A. Langston, S. Mac, and F. A. Rosamond. The undirected feedback vertex set problem has a poly(k) kernel. In H. L. Bodlaender and M. A. Langston, editors, *Proceedings 2nd International Workshop on Parameterized and Exact Computation, IWPEC 2006*, pages 192–202. Springer Verlag, Lecture Notes in Computer Science, vol. 4169, 2006.
7. B. Chor, M. Fellows, M. Ragan, F. Rosamond, I. Razgon, and S. Snir. Connected coloring completion for general graphs: algorithms and complexity. In *Proceedings 13th Annual International Conference on Computing and Combinatorics, COCOON 2007*, pages 75–85. Springer Verlag, Lecture Notes in Computer Science, vol. 4598, 2007.

8. B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Information and Computation*, 85:12–75, 1990.
9. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
10. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
11. J. Gramm, A. Nickelsen, and T. Tantau. Fixed-parameter algorithms in phylogenetics. To appear in *The Computer Journal*.
12. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38:31–45, 2007.
13. S. Moran and S. Snir. Convex recolorings of strings and trees: Definitions, hardness results, and algorithms. In F. K. H. A. Dehne, A. López-Ortiz, and J.-R. Sack, editors, *Proceedings WADS 2005: 9th International Workshop on Algorithms and Data Structures*, pages 218–232. Springer Verlag, Lecture Notes in Computer Science, vol. 3608, 2005.
14. S. Moran and S. Snir. Efficient approximation of convex recolorings. In *Proceedings APPROX 2005: 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, published with Proceedings RANDOM 2005*, pages 192–208. Springer Verlag, Lecture Notes in Computer Science, vol. 3624, 2005.
15. R. Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.