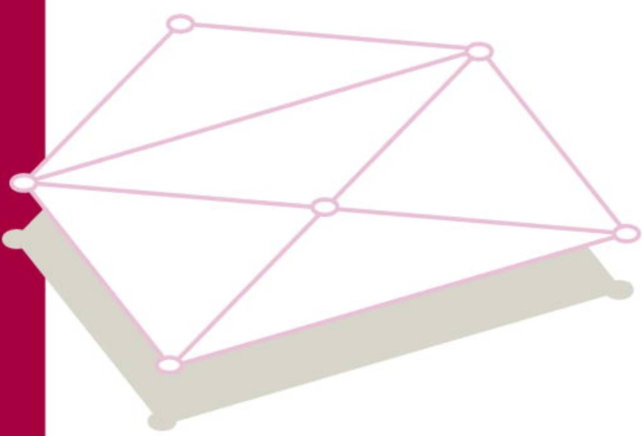
Volume 104, Number 2, 16 October 2007

ISSN 0020-0190

ELSEVIER

Information Processing Letters

Devoted to the rapid publication of short contributions to information processing



Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Information Processing Letters 104 (2007) 53–58
Information Processing Letters
www.elsevier.com/locate/ipl

A $2^{O(k)}$poly(n) algorithm for the parameterized Convex Recoloring problem

Igor Razgon

Computer Science Department, University College Cork, Ireland

Received 29 August 2006
Available online 23 May 2007
Communicated by F. Dehne

Abstract

In this paper we present a parameterized algorithm that solves the Convex Recoloring problem for trees in $O(256^k * \text{poly}(n))$. This improves the currently best upper bound of $O(k(k/\log k)^k * \text{poly}(n))$ achieved by Moran and Snir.

Keywords: Graph algorithms; Convex recoloring; Parameterized complexity

1. Introduction

Let T be a tree, C be a set of colors and $P: V(T) \to C$. We call the pair (T, P) a *colored tree* and P a *coloring* of T. The coloring P is *convex* if for each color $c \in C$, the set $P^{-1}(c)$ induces a subtree of T (in other words, the vertices corresponding to c induce a connected subgraph of T).

Let $S \subseteq V(T)$, $D: S \to C$ such that for each $v \in S$, $D(v) \neq P(v)$. Let P' be a coloring of T obtained from P by recoloring each $v \in S$ in $D(v)$. We say that (S, D) is a Convex Recoloring (CR) of (T, P) if P' is a convex coloring of T. The *size* of (S, D) is $|S|$.

In this Letter we consider a problem that gets as input a colored tree (T, P) and asks for a smallest CR of (T, P). A parameterized version of this problem gets in addition a parameter k and asks for existence of a CR of (T, P) of size at most k. We call this problem Convex Recoloring of Trees (CRT).

The CRT problem was introduced by Moran and Snir [5] as having applications in bioinformatics. They proved NP-hardness of the problem and showed its fixed-parameter tractability by presenting an $O(k(k/\log k)^k * \text{poly}(n))$ algorithm for this problem. The problem has also been considered in [3] where an $O(k^6)$ kernel has been obtained for this problem and in [2], where the size of the kernel has been reduced to $O(k^2)$. Variants of convex recoloring problem are studied in [1,4].

In this Letter we present a parameterized algorithm that solves the CRT problem in $O(256^k * \text{poly}(n))$ improving the result of Moran and Snir. To the best of our knowledge, this is the first algorithm solving the problem in $O(c^k * \text{poly}(n))$ where c is a constant.

1.1. Preliminaries

Let (T, P) be a colored tree. Let us call a monochromatic subtree of T with respect to P, a *color component*

E-mail address: i.razgon@cs.ucc.ie.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2007.05.007

of (T, P). A color c is *good* if there is only one color component of c. Otherwise c is a *bad* color. Let T' be a color component of (T, P) whose vertices are colored in c. A vertex $v \notin V(T')$ having a neighbor in $V(T')$ is *adjacent* to T'. Obviously, v is not colored in c.

Now we describe a greedy procedure of partitioning bad color components into *buckets*. Initially all the bad color components are *unmarked*. Let c_1 be a bad color component of color b. If any other component of color b lies at distance at least 3 from c_1, make bucket $B_1 = \{c_1\}$ and mark c_1. Otherwise, let v be a vertex *not* colored in b, which is adjacent to c_1 and at least one other component of color b. Let $c_1, \ldots, c_l$ be all the components of color b adjacent to v. Make bucket $B_1 = \{c_1, \ldots, c_l\}$ and mark the components $c_1, \ldots, c_l$.

Assume that i buckets ($i \geqslant 1$) have been already made. Let c_1 be an *unmarked* color component of some bad color b. If all other *unmarked* color components of b lie at distance at least 3 from c_1 then make bucket $B_{i+1} = \{c_1\}$ and mark c_1. Otherwise let v be a vertex *not* colored in b, which is adjacent to c_1 and another *unmarked* component colored in b. Let $c_1, \ldots, c_l$ be the components colored in b, which are adjacent to v. Make bucket $B_{i+1} = \{c_1, \ldots, c_l\}$ and mark all components $c_1, \ldots, c_l$. Proceed until there are no unmarked bad color components.

Let $B = \{B_1, \ldots, B_m\}$ be the set of buckets obtained by applying the above procedure. Let B_i be a bucket with at least two components. We call the vertex adjacent to all the components of B_i *the connecting vertex* of B_i. A vertex is *adjacent* to a bucket if it is adjacent to any of its components. The *path* between buckets B_k and B_l is the shortest path between a vertex adjacent to B_k and a vertex adjacent to B_l (this path may contain a single vertex only). We denote by $U(B_i)$ the set of all vertices that belong to the components of B_i. The following proposition is heavily used in the proofs of the present paper.

Proposition 1. *Let v_k and v_l be the vertices of two distinct buckets B_k and B_l, respectively. Then the path between v_k and v_l includes the path between B_k and B_l.*

Proof. By definition, the path between B_k and B_l is the shortest path between their adjacent vertices. Let u_k and u_l be these adjacent vertices. Consider a walk including the path from v_k to u_k, the path from u_k to u_l and the path from u_l to v_l. We prove the proposition by showing that this walk is actually a path. Assume by contradiction that a path from say v_k to u_k includes an intermediate external vertex and this external vertex belongs to a path from u_k to u_l. The only external vertex that might occur there is the connecting vertex u of B_k. In this case the path from u to u_k contains vertices of $U(B_k)$ as intermediate ones, while the path from u_k to u does not contain any of these vertices. In other words, we obtained a cycle, a contradiction. The possibility that the path from u_k to u_l and the path from u_l to v_l share intermediate vertices is ruled out analogously. Finally, if we assume that the path from v_k to u_k and the path from u_l to v_l share an intermediate vertex then, due to the disjointness of buckets, this intermediate vertex must be the connecting vertex v of both B_k and B_l. But in this case $v = u_k = u_l$ in contradiction to being v an *intermediate* vertex of the considered paths. □

1.2. *Structure of the paper*

Throughout the paper we assume that the input of the CRT problem is a colored tree (T, P) and a parameter k. We also assume that $B = \{B_1, \ldots, B_m\}$ is the set of buckets generated as shown in the previous subsection. The paper is organized as follows.

In Section 2 we prove that $|B| \leqslant 4k$ is a necessary condition for existence of a CR of size at most k. In the proof we use the fact that for a CR of size at most k, it is necessary that the number of bad colors is at most $2k$ [3]. In Section 3 we present a polynomial-time procedure that, given a pair of subsets of B, produces a convex recoloring. We then prove that for at least one pair of buckets, the procedure returns a smallest recoloring. It will follow the CRT problem can be solved by applying the procedure to each of $O(2^{2|B|})$ subsets of buckets. Taking into account that $|B| \leqslant 4k$, as proven in Section 2, we obtain a parameterized algorithm taking time of $O(2^{8k} * \mathrm{poly}(n)) = O(256^k * \mathrm{poly}(n))$.

2. The number of buckets is linear in the size of convex recoloring

Let (T, P) be input of the CRT problem and let l be the number of bad colors. Assume that $|B| = k_1 + \cdots + k_l$, where k_i is the number of buckets of color i.

Theorem 1. *Let (S, D) be a convex recoloring of (T, P). Then $|S| \geqslant (\sum_{i=1}^{l}(k_i - 1))/2$.*

Proof. Consider a bipartite graph $H = (B, S, E)$, where there is an edge between $B_i \in B$ and $v \in S$ if and only if v is a vertex of $U(B_i)$ or v is adjacent to B_i. We prove the theorem by showing that there is a correspondence $f : B' \to S$ with the following properties.

- B' is a subset of B containing $k_i - 1$ buckets of each color i.
- For each vertex $v \in S$, $|f^{-1}(v)| \leqslant 2$.

We construct the desired correspondence in two stages. On the first stage we proceed as follows. All the buckets are considered unmarked. We select an unmarked bucket B_i such that there is a vertex $v \in U(B_i) \cap S$, set $f(B_i) = v$ and mark B_i. We proceed the selection process until it is impossible to select a bucket satisfying the above conditions. Observe that there are no two marked buckets B_{i_1} and B_{i_2} that correspond to the same vertex v because in this case $v \in U(B_{i_1}) \cap U(B_{i_2})$ in contradiction to the disjointness of the buckets.

If after the first stage there is at most one unmarked bucket of each color, there desired correspondence is ready. Otherwise, we proceed constructing the correspondence based on the following two claims.

Claim 1. *Let B_i be a bucket of color b containing two or more components. Then at least one of the following two conditions holds*:

- $U(B_i) \cap S \neq \emptyset$.
- *The connecting vertex of B_i is recolored to b in D.*

Proof. Assume that the first condition does not hold. Then, if the connecting vertex of B_i is not colored in b, in the resulting coloring there are at least two vertices colored in b, the path between them is not entirely colored in b in contradiction to the convexity of the resulting coloring. □

Claim 2. *Let B_{i_1} and B_{i_2} be two distinct buckets of the same color b. Then at least one of the following conditions holds*:

- *Either $U(B_{i_1})$ or $U(B_{i_2})$ intersects with S.*
- *All the vertices of the path between the buckets are colored in b in the coloring obtained from the initial coloring P[1] as a result of recoloring vertices from S according to D.*

Proof. Assume that the first condition does not hold. Pick a vertex $v_1 \in U(B_{i_1})$ and a vertex $v_2 \in U(B_{i_2})$. Both these vertices are colored in b. Consequently, the path between them is colored in b in any convex coloring. By Proposition 1, the path between v_1 and v_2 includes the path between B_{i_1} and B_{i_2}. □

Now we present the second stage of constructing the correspondence. Let B_i be an unmarked bucket of color b such that there is another unmarked bucket of the same color. As the first stage has been finished no vertex of $U(B_i)$ belongs to S. Assume that B_i contains two or more components. Then, by Claim 1, the connecting vertex v of B_i belongs to S. Set the correspondence $f(B_i) = v$ and mark B_i. If B_i contains only one component then let B_j be another unmarked bucket of the same color. By Claim 2, the path between B_i and B_j is colored in b in any convex coloring. The vertex v of the path that is adjacent to B_i is not colored in b initially hence it belongs to S. Again, set the correspondence $f(B_i) = v$ and mark B_i. Proceed as shown above until for each color there is at most one unmarked bucket of this color.

Claim 3. *No two buckets marked on the second stage correspond to the same vertex.*

Proof. Assume that the statement is not true. Then there is a bucket B_i such that at the moment $f(B_i) = v$ is being set, another bucket B_j has been marked on the second stage and the correspondence $f(B_j) = v$ has been set. First of all, observe that B_i and B_j are of the same color b because v is colored in the initial colors of the buckets. Assume that both B_i and B_j contain two or more components. Then, according to the procedure of constructions of buckets, buckets B_i and B_j must be united to a single bucket as being of the same color and having the same connecting vertex. Indeed, assume, without loss of generality, that B_i is created *before* B_j. Then the components of B_i are *all* the components which are *unmarked* at the time of creation of B_i, colored in b, and adjacent to v. However, there are at least two additional components contained in B_j with *the same* properties, a contradiction. This argumentation shows that the case where both B_i and B_j contain two or more components cannot occur.

A similar argumentation works for the case where only B_i or only B_j contain two or more components. Assuming, without loss of generality, that B_i contains two or more components, we see that if B_i is created first, the component of B_j must be joined to B_i as being adjacent to the connecting vertex of B_i. If B_j is created first, the creating procedure cannot allow B_j to contain only one component because there is at least unmarked component "sharing" an adjacent vertex with the component of B_j. The latter argumentation also works for

[1] Recall that (T, P) is the input of the problem.

the case where both B_i and B_j contain only one component. Whoever is created first, cannot be allowed to contain only one component because this component has a common adjacent vertex with another unmarked component.

We obtained contradiction in all possible cases, hence the claim is true.

Let B' be the set of all marked buckets. It follows that for each vertex v of S, there is at most one bucket corresponding to v is marked at the first stage and at most one bucket corresponding to v is marked on the second stage (Claim 3). In other words, $|f^{-1}(v)| \leqslant 2$. Hence $|S| \geqslant |B'|/2 = (\sum_{i=1}^{l}(k_i - 1))/2$. □

Corollary 1. *If for the given tree there is a convex recoloring of size at most k then the number of buckets is at most* $4k$.

Proof. Assume that the given tree has a convex recoloring of size at most k. Then, by Theorem 1, $k \geqslant (k_1 + \cdots + k_l - l)/2$ (recall that l is the number of bad colors, $k_1 + \cdots + k_l$ is the number of buckets). It follows that the number of buckets is at most $2k + l$. It has been shown by Bodlaender et al. [3] that l is at most $2k$. □

3. The algorithm

Let SB_1 and SB_2 be two subsets of B (recall that B is a subset of buckets) such that $SB_2 \subseteq SB_1$. We present a procedure that $Recolor(SB_1, SB_2)$ that returns a recoloring (S, D). We then prove that this recoloring is convex. Further, we prove that for at least one pair (SB_1, SB_2), the resulting recoloring is smallest. Taking into account that the *Recolor* procedure takes a polynomial time and that the number of distinct pairs (SB_1, SB_2) is at most $2^{2|B|}$, we obtain an algorithm that solves the CRT problem in $O(2^{2|B|}\mathrm{poly}(n))$. Considering that a convex recoloring of size at most k can exist only if the number of buckets is at most $4k$, we get a parameterized algorithm for the CRT problem that takes time of $O(2^{8k}\mathrm{poly}(n)) = O(256^k\mathrm{poly}(n))$.

Let us define $FixedColored(SB_1, SB_2, b)$, where b is a bad color as a subset of vertices of $V(T)$ containing the following vertices.

(i) For each pair $\{B_{i_1}, B_{i_2}\}$ of distinct buckets of SB_1 colored in b in (T, P), the shortest path between B_{i_1} and B_{i_2}. Obviously, vertices of this category are contained in $FixedColored(SB_1, SB_2, b)$ only if there are at least two buckets of color b in SB_1.
(ii) For each bucket B_i of SB_2 having two or more color components, the connecting vertex of B_i.
(iii) The minimal subset S of vertices such that the vertices of the above two categories together with S induce a single subtree.

We assume that $FixedColored(SB_1, SB_2, c) = \emptyset$ for each good color c.

Below we describe a procedure $Recolor(SB_1, SB_2)$.

$Recolor(SB_1, SB_2)$

- **Initial recoloring**
  (i) **if** $FixedColored(SB_1, SB_2, b_1)$ has a nonempty intersection with $FixedColored(SB_1, SB_2, b_2)$ for some bad colors b_1 and b_2 **then** return '*INFEASIBLE*'.
  (ii) Let (T, P') be a colored tree obtained from (T, P) by setting the color of vertices of $FixedColored(SB_1, SB_2, b)$ to b for each bad color b.
- **Identifying the additional set of vertices to be recolored**
  (i) For each color c such that $FixedColored(SB_1, SB_2, c) \neq \emptyset$, let $g(c)$ be the color component of c in (T, P'), which contains all the vertices of $FixedColored(SB_1, SB_2, c)$.
  (ii) For each color c such that $FixedColored(SB_1, SB_2, c) = \emptyset$, let $g(c)$ be the color component of c in (T, P'), which has the greatest size.
  (iii) For each color c, let $r(c)$ be the set of vertices contained in all the components of color c in (T, P') except $g(c)$.
  (iv) Let RC be the union of all $r(c)$.
- **The additional recoloring**
  (i) color each connected subtree induced by the vertices of RC to the color of any vertex of $T \setminus RC$ adjacent to that subtree.
  (ii) Let (T, P'') be the colored tree obtained from (T, P') by the above coloring of RC.
  (iii) Return $(S, P'' \setminus P)$, where S is the domain of $P'' \setminus P$.

Lemma 1. *If* $Recolor(SB_1, SB_2)$ *does not return 'INFEASIBLE', it returns a convex recoloring of* (T, P).

Proof. We will prove that (T, P'') has at most one color component per color. Observe that the vertices of $T \setminus RC$ are colored by P'' in the same color as by P'. Moreover, note that the coloring of vertices of $T \setminus RC$ by P' is convex because, by definition of RC, it contains all but one color components of (T, P') for each color. Hence, if (T, P'') is non-convex, the "non-convexity" is introduced by coloring of vertices of RC in P''. Observe that any connected component induced by the vertices of *RC* is colored in the color of a vertex of $T \setminus RC$ *adjacent* to that component. That is, one can show by induction that coloring of each new connected component induced by *RC does not* produce a bad color, hence there are no bad colors at the end of the coloring process. □

Now we shall prove that for at least one feasible input (SB_1, SB_2) (such that $Recolor(SB_1, SB_2)$ does not return '*INFEASIBLE*'), the *Recolor* procedure returns the smallest recoloring.

Theorem 2. *$Recolor(SB_1, SB_2)$ returns the smallest recoloring for at least one feasible input (SB_1, SB_2).*

Proof. Let (S_1, D_1) be a smallest convex recoloring of (T, P). Let SB_1 be the subset of buckets of B not all vertices of which are recolored by (S_1, D_1). Let SB_2 be the subset of SB_1 containing each bucket B' with 2 or more components such that the connecting vertex of B' is recolored into the initial color of the vertices of B'. We will show that (SB_1, SB_2) is a feasible input of the *Recolor* procedure. Further, let (S, D) be the recoloring returned by $Recolor(SB_1, SB_2)$. We will prove that $|S| \leqslant |S_1|$, from which the theorem will follow.

The key property used for the proof is expressed by the following lemma.

Lemma 2. *Let (T, P_2) be the colored tree obtained from (T, P) by recoloring the vertices of S_1 according to D_1. Then for each bad color b in (T, P), all vertices of $FixedColored(SB_1, SB_2, b)$ are colored in b in (T, P_2).*

Proof. We prove the lemma for each category of vertices of $FixedColored(SB_1, SB_2, b)$.

Let $v \in FixedColored(SB_1, SB_2, b)$ be a vertex of the first category. By selection of SB_1 and Proposition 1 a vertex v lies on the path between vertices $v_1 \in U(B_{i_1})$ and $v_2 \in U(B_{i_2})$, which are not recolored. Hence, v must be colored in b by P_2.

Let v be a vertex of the second category of $FixedColored(SB_1, SB_2, b)$. By selection of SB_2, v is the connecting vertex of some bucket $B_i \in SB_2$, hence v gets color b in P_2 by definition of SB_2 in the beginning of the proof of the present theorem.

Now, as the vertices of the first two categories of $FixedColored(SB_1, SB_2, b)$, are colored in b in (T, P_2), the vertices lying in the paths between them are also colored in b. But those vertices constitute the set of vertices of the third category of $FixedColored(SB_1, SB_2, b)$. □

An immediate corollary from Lemma 2 is that (SB_1, SB_2) is a feasible input of the *Recolor* procedure. Really, if not then $FixedColored(SB_1, SB_2, b_1) \cap FixedColored(SB_1, SB_2, b_2) \neq \emptyset$ for some bad colors b_1 and b_2. Consequently the vertices that belong to the above intersection are colored in two colors by P_2, a contradiction.

Thus, $Recolor(SB_1, SB_2)$ returns a recoloring (S, D) and we have to prove that $|S| \leqslant |S_1|$. Let F be the set of all vertices v such that $v \in FixedColored(SB_1, SB_2, b)$ for some bad color b but *not* colored in b in (T, P). By the description of the *Recolor* procedure and Lemma 2 $F \subseteq S \cap S_1$. Moreover, as a result of recoloring of vertices of F, (T, P) is transformed in (T, P'). It follows that (T, P'') is obtained from (T, P') by recoloring the vertices of $S \setminus F$, while (T, P_2) is obtained from (T, P') by recoloring the vertices of $S_1 \setminus F$. Let $P'(c)$ be the subset of vertices of T colored in c by P'. Clearly, the union of all $P'(c)$ is $V(T)$. We prove the theorem by showing that for each color c, $|(S \setminus F) \cap P'(c)| \leqslant |(S_1 \setminus F) \cap P'(c)|$. We consider only colors c such that $P'(c)$ induces two or more subtrees of T: in case of one subtree $|(S \setminus F) \cap P'(c)| = 0$ and the desired statement trivially follows.

Let c be a color such that $FixedColored(SB_1, SB_2, c) = \emptyset$.

Lemma 3. *All the components of c in $T(P')$ but at most one are fully recolored in $T(P_2)$.*

Proof. Note that in the considered case P' does not color new vertices in c in addition to those colored in c in (T, P), that is $P'(c) \subseteq P(c)$. Thus the lemma immediately follows in case all vertices of $P(c)$ are recolored in P_2. Assume that this is not so. Then, the fact that $FixedColored(SB_1, SB_2) = \emptyset$ implies that SB_1 contains exactly one bucket B' whose vertices are colored in c in (T, P) and that exactly one component g' of B' is not fully recolored. If the vertices of g' that are not recolored by (T, P') constitute a single component, we are done. Otherwise, the vertices of g' are separated into several components by vertices of F recolored into colors different from c. Let v_1 and v_2 be two vertices of g' that belong to different color components of c in (T, P'). The path between them passes through a vertex v of F whose color is different from c and preserved in P_2. Consequently, to avoid convexity violation either v_1 or v_2 is recolored in P_2. □

Lemma 3 shows that $S_1 \setminus F$ includes all the components of c but at most one, while $S \setminus F$ includes, by definition, all the components of c but the *largest* one. Clearly, $|(S \setminus F) \cap P'(c)| \leqslant |S_1 \setminus F \cap P'(c)$ in this case.

Let c be a color such that $FixedColored(SB_1, SB_2, c) \neq \emptyset$.

Lemma 4. *Each color component of c in (T, P) which is not fully recolored in P_2 is adjacent to a vertex of $FixedColored(SB_1, SB_2, c)$.*

Proof. Let g' be a color component of c in (T, P) which is not fully recolored by P_2. Clearly g' is a component of a bucket B' of SB_1. If $B' \in SB_2$ then the connecting vertex of B' belongs to $FixedColored(SB_1, SB_2, c)$, hence the lemma is valid for this case. Assume that $B' \in SB_1 \setminus SB_2$. Then, to ensure that $FixedColored(SB_1, SB_2, c) \neq \emptyset$, SB_1 must contain another bucket B'' colored in c in (T, P). By definition, $FixedColored(SB_1, SB_2, c)$ contains the path between B' and B''. Let v_1 be the vertex of this path adjacent to B'. Taking into account that $B' \in SB_1 \setminus SB_2$, v_1 is not the connecting vertex of B', hence it is adjacent to exactly one component g'' of B'. If $g'' = g'$ we are done. Otherwise, g'' is fully recolored in P_2 in order to avoid the connecting vertex of B' to be recolored in c. Let v_2 be the vertex of g'' adjacent to v_1. Let v_3 be a vertex of g'. Let Z be the path from v_3 to v_2 in P. Observe that the connecting vertex of B' is the only vertex of Z that does not belong to B'. That is, v_1 does not belong to Z. It follows that Z together with $\{v_1, v_2\}$ constitutes the path Z' from v_3 to v_1. In other words, Z' is the path between two vertices colored in c in P_2 which passes through a vertex not colored in c in P_2, namely v_2, in contradiction to the convexity of (T, P_2).

Let g be the component induced by $P'(c)$ that contains $FixedColored(SB_1, SB_2, c)$. According to the description of $Recolor(SB_1, SB_2)$, $(S \setminus F) \cap P'(c) = P'(c) \setminus V(g)$. We are going to show that any vertex of $P'(c) \setminus V(g)$ belongs to $(S_1 \setminus F) \cap P'(c)$ which will finish the proof of the theorem. Assume that it is not so regarding some $v \in P'(c) \setminus V(g)$. Observe that $v \in P(c) \cap P'(c)$ because all the vertices of $P'(c) \setminus P(c)$ belong to $V(g)$ by construction. By Lemma 4, v belongs to a color component of c in (T, P) adjacent to a vertex v_1 of $FixedColor(SB_1, SB_2, c)$. In other words, all the vertices of the path Z from v to v_1, except v_1 itself, have color c in (T, P). By construction, $v_1 \in V(g)$. Taking into account that $v \notin V(g)$, at least one vertex v' of Z is recolored by P' and hence preserves its role in P_2. Thus, Z is the path between two vertices colored in c by P_2 which passes through a vertex not coloring in c in P_2 in contradiction to the convexity of P_2. □

An immediate corollary of Theorem 2 is an algorithm that solves the CRT problem in $O(2^{2|B|}poly(n))$. Try all possible pairs of buckets (SB_1, SB_2) such that $SB_2 \subseteq SB_1$ and select the smallest returned recoloring. By Lemma 1, this recoloring is convex. By Theorem 2, this recoloring is optimal. According to Theorem 1, the algorithm can return "NO" at the preprocessing stage if the number of buckets is greater than $4k$. As a result, the complexity of the algorithm is bounded to $O(2^{8k}poly(n)) = O(256^k poly(n))$.

References

[1] E. Bachoore, H. Bodlaender, Convex recoloring of leaf-colored trees, Technical Report UU-CS-2006-10, Department of Information and Computing Sciences, Utrecht University, 2006.

[2] H. Bodlaender, M. Fellows, M. Langston, M. Ragan, F. Rosamond, M. Weyer, Quadratic kernelization for convex recoloring, in: COCOON 2007, in press.

[3] H. Bodlaender, M. Fellows, M. Langston, M. Ragan, F. Rosamond, M. Weyer, Kernelization for convex recoloring, in: ACID 2006, King's College London, pp. 23–35.

[4] B. Chor, M. Fellows, M. Ragan, F. Rosamond, S. Snir, I. Razgon, Connected coloring completion for general graphs: Algorithms and complexity, in: COCOON 2007, in press.

[5] S. Moran, S. Snir, Convex recolorings of strings and trees: Definitions, hardness results and algorithms, in: WADS 2005, Springer-Verlag, pp. 218–232.