Efficient Approximation of Convex Recolorings *

Shlomo Moran† Sagi Snir‡

June 14, 2005

Abstract

A coloring of a tree is convex if the vertices that pertain to any color induce a connected subtree; a partial coloring (which assigns colors to some of the vertices) is convex if it can be completed to a convex (total) coloring. Convex coloring of trees arises in areas such as phylogenetics, linguistics, etc. e.g., a perfect phylogenetic tree is one in which the states of each character induce a convex coloring of the tree. Research on perfect phylogeny is usually focused on finding a tree so that few predetermined partial colorings of its vertices are convex.

When a coloring of a tree is not convex, it is desirable to know "how far" it is from a convex one. In [18], a natural measure for this distance, called *the recoloring distance* was defined: the minimal number of color changes at the vertices needed to make the coloring convex. This can be viewed as minimizing the number of "exceptional vertices" w.r.t. to a closest convex coloring. The problem was proved to be NP-hard even for colored strings.

In this paper we continue the work of [18], and present a 2-approximation algorithm of convex recoloring of strings whose running time $O(cn)$, where c is the number of colors and n is the size of the input, and an $O(cn^2)$ 3-approximation algorithm for convex recoloring of trees.

*A preliminary version of the results in this paper appeared in [19].

†Computer Science dept., Technion, Haifa 32000, Israel. moran@cs.technion.ac.il

‡Mathematics dept. University of California, Berkeley, CA 94720, USA. ssagi@math.berkeley.edu

1 Introduction

A phylogenetic tree is a tree which represents the course of evolution for a given set of species. The leaves of the tree are labeled with the given species. Internal vertices correspond to hypothesized, extinct species. A *character* is a biological attribute shared among all the species under consideration, although every species may exhibit a different *character state*. Mathematically, if X is the set of species under consideration, a character on X is a function C from X into a set $\mathcal{C}$ of character states. A character on a set of species can be viewed as a *coloring* of the species, where each color represents one of the character's states. A natural biological constraint is that the reconstructed phylogeny have the property that each of the characters could have evolved without reverse or convergent transitions: In a reverse transition some species regains a character state of some old ancestor whilst its direct ancestor has lost this state. A convergent transition occurs if two species possess the same character state, while their least common ancestor possesses a different state.

In graph theoretic terms, the lack of reverse and convergent transitions means that the character is *convex* on the tree: for each state of this character, all species (extant and extinct) possessing that state induce a single *block*, which is a maximal monochromatic subtree. Thus, the above discussion implies that in a phylogenetic tree, each character is likely to be convex or "almost convex". This makes convexity a fundamental property in the context of phylogenetic trees to which a lot of research has been dedicated throughout the years. The *Perfect Phylogeny* (PP) problem, whose complexity was extensively studied (e.g. [13, 16, 1, 17, 7, 23]), seeks for a phylogenetic tree that is simultaneously convex on each of the input characters. *Maximum parsimony* (MP) [10, 21] is a very popular tree reconstruction method that seeks for a tree which minimizes the parsimony score defined as the number of mutated edges summed over all characters (therefore, PP is a special case of MP). [12] introduce another criterion to estimate the distance of a phylogeny from convexity. They define the *phylogenetic number* as the maximum number of connected components a single state induces on the given phylogeny (obviously, phylogenetic number one corresponds to a perfect phylogeny).

Convexity is a desired property in other areas of classification, beside phylogenetics. For instance, in [6, 5] a method called *TNoM* is used to classify genes, based on data from gene expression extracted from two types of tumor tissues. The method finds a separator on a binary vector, which minimizes the number of "1" in one side and "0" in the other, and thus defines a convex vector of minimum Hamming distance to the given binary vector. In [14], distance from convexity is used (although not explicitly) to show strong connection between strains of Tuberculosis and their human carriers.

In a previous work [18], we defined and studied a natural distance from a given coloring to a convex one: the *recoloring distance*. In the simplest, unweighted model, this distance is the minimum number of color changes at the vertices needed to make the given coloring convex (for strings this reduces to Hamming distance from a closest convex coloring). This model was extended to a weighted model, where changing the color of a vertex v costs a nonnegative weight $w(v)$. The most general model studied in [18] is the *non-uniform* model, where the cost of coloring vertex v by a color d is an arbitrary nonnegative number $cost(v, d)$.

It was shown in [18] that finding the recoloring distance in the unweighted model is NP-hard even for strings (trees with two leaves), and few dynamic programming algorithms for exact solutions of few variants of the problem were presented.

In this work we present two polynomial time, constant ratio approximation algorithms, one for strings and one for trees. Both algorithms are for the weighted (uniform) model. The algorithm for strings is based on a lower bound technique which assigns penalties to colored trees. The 2-approximation is based on the fact that for a string, the penalty is at most twice the cost of an optimal convex recoloring. This fact does not hold for trees, where a different technique is used. The algorithm for trees is based on a recursive construction that uses a variant of the local ratio technique [3, 4], which allows adjustments of the underlying tree topology during the recursive process.

The rest of the paper is organized as follows. In the next section we present the notations and define the models used. In Section 3 we define the notion of penalty which provides lower bounds on the optimal cost of convex recoloring of any tree. In Section 4, we present the 2-approximation algorithm for the string. In Section 5 we briefly explain the local ratio technique, and present the 3-approximation algorithm for the tree. We conclude and point out future research directions in Section 6.

2 Preliminaries

A colored tree is a pair (T, C) where $T = (V, E)$ is a tree with vertex set $V = \{v_1, \ldots, v_n\}$, and C is a *coloring* of T, i.e. - a function from V onto a set of colors $\mathcal{C}$. For a set $U \subseteq V$, $C|_U$ denotes the restriction of C to the vertices of U, and $C(U)$ denotes the set $\{C(u) : u \in U\}$. For a subtree $T' = (V(T'), E(T'))$ of T, $C(T')$ denotes the set $C(V(T'))$. A *block* in a colored tree is a maximal set of vertices which induces a monochromatic subtree. A *d-block* is a block of color d. The number of d-blocks is denoted by $n_b(C, d)$, or $n_b(d)$ when C is clear from the context. A coloring C is said to be *convex* if $n_b(C, d) = 1$ for every color $d \in \mathcal{C}$. The number of *d-violations* in the coloring C is $n_b(C, d) - 1$, and the total number of *violations* of C is $\sum_{c \in \mathcal{C}}(n_b(C, d) - 1)$. Thus a coloring C is convex iff the total number of violations of C is zero (in [9] the above sum, taken over all characters, is used as a measure of the distance of a given phylogenetic tree from perfect phylogeny).

The definition of convex coloring is extended to *partially colored* trees, in which the coloring C assigns colors to some subset of vertices $U \subseteq V$, which is denoted by $Domain(C)$. A partial coloring is said to be convex if it can be extended to a total convex coloring (see [22]). Convexity of partial and total coloring have simple characterization by the concept of *carriers*: For a subset U of V, $carrier(U)$ is the minimal subtree that contains U. For a colored tree (T, C) and a color $d \in C$, $carrier_T(C, d)$ (or $carrier(C, d)$ when T is clear) is the carrier of $C^{-1}(d)$. We say that C has the *disjointness property* if for each pair of colors $\{d, d'\}$ it holds that $carrier(C, d) \cap carrier(C, d') = \emptyset$. It is easy to see that a total or partial coloring C is convex iff it has the disjointness property (in [8] convexity is actually defined by the disjointness property).

When some (total or partial) input coloring (C, T) is given, any other coloring C' of T is viewed as a *recoloring* of the input coloring C. We say that a recoloring C' of C *retains* (the color of) a

vertex v if $C(v) = C'(v)$, otherwise C' *overwrites* v. Specifically, a recoloring C' of C overwrites a vertex v either by changing the color of v, or just by *uncoloring* v. We say that C' retains (overwrites) a set of vertices U if it retains (overwrites resp.) every vertex in U. For a recoloring C' of an input coloring C, $\mathcal{X}_C(C')$ (or just $\mathcal{X}(C')$) is the set of the vertices overwritten by C', i.e.

$$\mathcal{X}_C(C') = \{v \in V : [v \in Domain(C)] \bigwedge [(v \notin Domain(C')\ ) \vee (C(v) \neq C'(v)\ )]\}.$$

With each recoloring C' of C we associate a *cost*, denoted as $cost_C(C')$ (or $cost(C')$ when C is understood), which is the number of vertices overwritten by C', i.e. $cost_C(C') = |\mathcal{X}_C(C')|$. A coloring C^* is an *optimal convex recoloring of* C, or in short an *optimal recoloring of* C, and $cost_C(C^*)$ is denoted by $OPT(T, C)$, if C^* is a convex coloring of T, and $cost_C(C^*) \leq cost_C(C')$ for any other convex coloring C' of T.

The above cost function naturally generalizes to the *weighted* version: the input is a triplet (T, C, w), where $w : V \to \mathbb{R}^+ \cup \{0\}$ is a weight function which assigns to each vertex v a nonnegative weight $w(v)$. For a set of vertices X, $w(X) = \sum_{v \in X} w(v)$. The cost of a convex recoloring C' of C is $cost_C(C') = w(\mathcal{X}(C'))$, and C' is an optimal convex recoloring if it minimizes this cost.
The above unweighted and weighted cost models are *uniform*, in the sense that the cost of a recoloring is determined by the set of overwritten vertices, regardless the specific colors involved. [18] defines also a more subtle *non uniform model*, which is not studied in this paper.

Let AL be an algorithm which receives as an input a weighted colored tree (T, C, w) and outputs a convex recoloring of (T, C, w), and let $AL(T, C, w)$ be the cost of the convex recoloring output by AL. We say that AL is an *r-approximation* algorithm for the convex tree recoloring problem if for all inputs (T, C, w) it holds that $AL(T, C, w)/OPT(T, C, w) \leq r$ [11, 15, 24].

We complete this section with a definition and a simple observation which will be useful in the sequel. Let (T, C) be a colored tree. A coloring C^* is an *expanding* recoloring of C if in each block of C^* at least one vertex v is retained (i.e., $C(v) = C^*(v)$).

Observation 2.1 *let $(T = (V, E), C, w)$ be a weighted colored tree, where $w(V) > 0$. Then there exists an expanding optimal convex recoloring of C.*

Proof. Let C' be an optimal recoloring of C which uses a minimum number of colors (i.e. $|C'(V)|$ is minimized). We shall prove that C' is an expanding recoloring of C.

Since $w(V) > 0$, the claim is trivial if C' uses just one color. So assume for contradiction that C' uses at least two colors, and that for some color d used by C', there is no vertex v s.t. $C(v) = C'(v) = d$. Then there must be an edge (u, v) such that $C'(u) = d$ but $C'(v) = d' \neq d$. Therefore, in the uniform cost model, the coloring C'' which is identical to C' except that all vertices colored d are now colored by d' is an optimal recoloring of C which uses a smaller number of colors - a contradiction. ■

In view of Observation 2.1 above, we assume in the sequel (sometimes implicitly) that the given optimal convex recolorings are expanding.

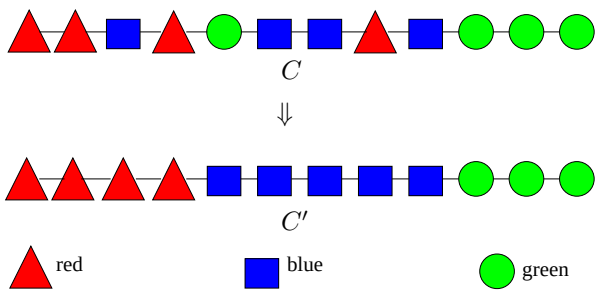

Figure 1: C' is a convex recoloring for C which defines the following penalties: $p_{green}(C') = 1$, $p_{red}(C') = 2$, $p_{blue}(C') = 3$

3 Lower Bounds via Penalties

In this section we present a general lower bound on the recoloring distance of weighted colored trees. Although for a general tree this bound can be fairly poor, in the next section we present an algorithm for convex recoloring of strings, which always finds a convex recoloring whose cost is at most twice this lower bound, and hence it is a 2-approximation algorithm for strings.

Let (T, C, w) be a weighted colored tree. For a color d and $U \subseteq V(T)$ let:

$$penalty_{C,d}(U) = w(U \cap \overline{C^{-1}(d)}) + w(\overline{U} \cap C^{-1}(d))$$

Informally, when the vertices in U induce a subtree, $penalty_{C,d}(U)$ is the total weight of the vertices which must be overwritten to make U the unique d-block in the coloring: a vertex v must be overwritten either if $v \in U$ and $C(v) \neq d$, or if $v \notin U$ and $C(v) = d$.

$penalty_C(C')$, the penalty of a convex recoloring C' of C, is the sum of the penalties of all the colors, with respect to the color blocks of C':

$$penalty_C(C') = \sum_{d \in \mathcal{C}} penalty_{C,d}(C'^{-1}(d))$$

Figure 1 depicts the calculation of a penalty associated with a convex recoloring C' of C.

In the sequel we assume that the input colored tree (T, C) is fixed, and omit it from the notations.

Claim 3.1 $penalty(C') = 2cost(C')$

Proof. From the definitions we have

$$\begin{aligned} penalty(C') &= \sum_{d \in \mathcal{C}} w\left(\{v \in V : C'(v) = d \text{ and } C(v) \neq d\} \cup \{v \in V : C'(v) \neq d \text{ and } C(v) = d\}\right) \\ &= 2w(\{v \in V : C'(v) \neq C(v)\}) = 2cost(C') \end{aligned}$$

■

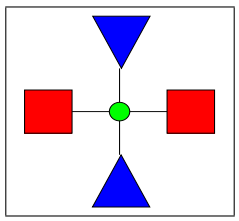

Figure 2: A convex recoloring must overwrite at least one of the large lateral blocks (a triangle or a rectangle).

As can be seen in Figure 1, $penalty(C') = 6$ while $cost(C') = 3$.

For each color d, p^*_d is the penalty of a block which minimizes the penalty for d:

$$p^*_d = \min\{penalty_d(V(T')) : T' \text{ is a subtree of } T\}$$

Corollary 3.2 *For any recoloring C' of C,*

$$\sum_{d\in\mathcal{C}} p^*_d \leq \sum_{d\in\mathcal{C}} penalty_d(C') = 2cost(C').$$

Proof. The inequality follows from the definition of p^*_d, and the equality from Claim 3.1. ∎

Corollary 3.2 above provides a lower bound on the cost of convex recoloring of trees. It can be shown that this lower bound can be quite poor for trees, that is: $OPT(T, C)$ can be considerably larger than $(\sum_{d\in\mathcal{C}} p^*_d)/2$. For example, any convex recoloring of the tree in Figure 2, must overwrite at least one of the (large) lateral blocks in the tree, while $(\sum_{d\in\mathcal{C}} p^*_d)/2$ in that tree is the weight of the (small) central vertex (the circle). However in the next section we show that this bound can be used to obtain a polynomial time 2-approximation for convex recoloring of strings.

4 A 2-Approximation Algorithm for Strings

Let a weighted colored string (S, C, w), where $S = (v_1, \ldots, v_n)$, be given. For $1 \leq i \leq j \leq n$, $S[i, j]$ is the substring $(v_i, v_{i+1}, \ldots, v_j)$ of S. The algorithm starts by finding for each d a substring $B_d = S[i_d, j_d]$ for which $penalty_d(S[i_d, j_d]) = p^*_d$. It is not hard to verify that B_d consists of a subsequence of consecutive vertices in which the difference between the total weight of d-vertices and the total weight of other vertices (i.e. $w(B_d \cap C^{-1}(d)) - w(B_d \setminus C^{-1}(d))$) is maximized, and thus B_d can be found in linear time. We say that a vertex v is *covered by color d* if it belongs to B_d. v is *covered* if it is covered by some color d, and it is *free* otherwise.

We describe below a linear time algorithm which, given the blocks B_d, defines a convex coloring $\hat{C}$ so that $cost(\hat{C}) \leq \sum_d p^*_d$. Hence, by Corollary 3.2, $\hat{C}$ is a 2-approximation to a minimal convex recoloring of C.

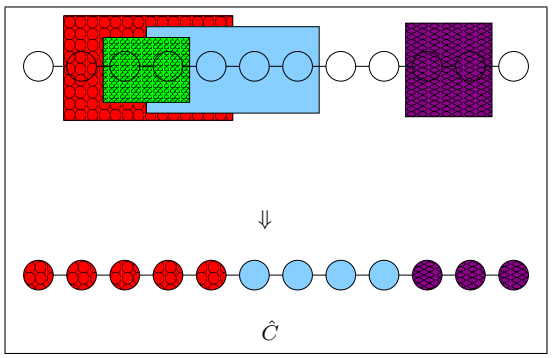

Figure 3: The upper part of the figure shows the optimal blocks on the string and the lower part shows the coloring returned by the algorithm.

$\hat{C}$ is constructed by performing one scan of S from left to right. The scan consists of at most c stages, where stage j defines the $j-th$ block of $\hat{C}$, to be denoted F_j, and its color, d_j, as follows.

Let d_1 be the color of the leftmost covered vertex (note that v_1 is either free or covered by d_1). d_1 is taken to be the color of the first (leftmost) block of $\hat{C}$, F_1, and $\hat{C}(v_1)$ is set to d_1. For $i > 1$, $\hat{C}(v_i)$ is determined as follows: Let $\hat{C}(v_{i-1}) = d_j$. Then if $v_i \in B_{d_j}$ or v_i is free, then $\hat{C}(v_i)$ is also set to d_j. Else, v_i must be a covered vertex. Let d_{j+1} be one of the colors that cover v_i. $\hat{C}(v_i)$ is set to d_{j+1} (and v_i is the first vertex in F_{j+1}).

Observation 4.1 *$\hat{C}$ is a convex coloring of S.*

Proof. Let d_j be the color of the $j-th$ block of $\hat{C}$, F_j, as described above. The convexity of $\hat{C}$ follows from the the following invariant, which is easily proved by induction: For all $j \geq 1$, $\cup_{k=1}^{j} F_k \supseteq \cup_{k=1}^{j} B_{d_k}$. This means that, for all j, no vertex to the right of F_j is covered by d_j, and hence no such vertex is colored by d_j. ■

Thus it remains to prove

Lemma 1 *$cost(\hat{C}) \leq \sum_{d \in \mathcal{C}} p^*_d$.*

Proof. $cost(\hat{C}) = \sum\{w(v_i) : C(v_i) \neq \hat{C}(v_i)\}$. Thus $w(v_i)$ is added to $cost(\hat{C})$ only if $C(v_i) = d$ and $\hat{C}(v_i) = d'$ for some distinct d', d. By the algorithm, $\hat{C}(v_i) = d'$ only if $v_i \in B_{d'}$ or v_i is free. In the first case $w(v_i)$ is accounted for in $p^*_{d'}$, and in the second case it is accounted for in p^*_d. In both cases $w(v_i)$ is added to sum on the righthand side, which proves the inequality. ■

5 A 3-Approximation Algorithm for Tree

In this section we present a polynomial time algorithm which approximates the minimal convex coloring of a weighted tree by factor three. The input is a triplet (T, C, w), where w is a nonnegative

weight function and C is a (possibly partial) coloring whose domain is the set $support(w) = \{v \in V : w(v) > 0\}$.

We first introduce the notion of covers w.r.t. colored trees. A set of vertices X is a *convex cover* (or just a cover) for a colored tree (T, C) if the (partial) coloring $C_X = C|_{[V \setminus X]}$ is convex (i.e., C can be transformed to a convex coloring by overwriting only vertices in X). Thus, if C' is a convex recoloring of (T, C), then $\mathcal{X}_C(C')$, the set of vertices overwritten by C', is a cover for (T, C). Moreover, deciding whether a subset $X \subseteq V$ is a cover for (T, C), and constructing a total convex recoloring C' of C such that $\mathcal{X}(C') \subseteq X$ in case it is, can be done in $O(n \cdot n_c)$ time. Therefore, finding an optimal convex total recoloring of (T, C) is polynomially equivalent to finding an optimal convex cover for (T, C).

Our approximation algorithm makes use of the local ratio technique, which is useful for approximating optimization covering problems such as vertex cover, dominating set, minimum spanning tree, feedback vertex set and more [4, 2, 3]. We hereafter describe it briefly:
The input to the problem is a triplet $(V, f : 2^V \to \{0, 1\}, w : V \to \mathbb{R}^+)$. Let $\Sigma = \{U \subseteq V : f(U) = 1\}$. The goal is to find a subset $X \in \Sigma$ such that $w(X)$ is minimized, i.e. $w(X) = OPT(V, \Sigma, w) = \min_{Y \in \Sigma} w(Y)$ (in our context V is the set of vertices, and Σ is the set of covers). The local ratio principle is based on the following observation (see e.g. [3]):

Observation 5.1 *For every two weight functions w_1, w_2:*

$$OPT(V, \Sigma, w_1) + OPT(V, \Sigma, w_2) \leq OPT(V, \Sigma, w_1 + w_2)$$

Now, given our initial weight function w, we select w_1, w_2 s.t. $w_1 + w_2 = w$ and $|supprt(w_1)| < |support(w)|$. We first apply the algorithm to find an r-approximation to (V, Σ, w_1) (in particular, if $V \setminus support(w_1)$ is a cover, then it is an optimal cover to (V, Σ, w_1)). Let X be the solution returned for (V, Σ, w_1), and assume that $w_1(X) \leq r \cdot OPT(V, \Sigma, w_1)$. If we could also guarantee that $w_2(X) \leq r \cdot OPT(V, \Sigma, w_2)$ then by Observation 5.1 we are guaranteed that X is also an r-approximation for $(V, \Sigma, w_1 + w_2 = w)$. The original property, introduced in [4], which was used to guarantee that $w_2(X) \leq r \cdot OPT(V, \Sigma, w_2)$ is that w_2 is *r-effective*, that is: for every $X \in \Sigma$ it holds that $w_2(X) \leq r \cdot OPT(V, \Sigma, w_2)$ (note that if $V \in \Sigma$, the above is equivalent to requiring that $w_2(V) \leq r \cdot OPT(V, \Sigma, w_2)$).

Theorem 5.2 *[4] Given $X \in \Sigma$ s.t. $w_1(X) \leq r \cdot OPT(V, \Sigma, w_1)$. If w_2 is r-effective, then $w(X) = w_1(X) + w_2(X) \leq r \cdot OPT(V, \Sigma, w)$.*

We start by presenting two applications of Theorem 5.2 to obtain a 3-approximation algorithm for convex recoloring of strings and a 4-approximation algorithm for convex recoloring of trees.

3-string-APPROX:
Given an instance of the convex weighted string problem (S, C, w):

1. If $V \setminus support(w)$ is a cover then $X \leftarrow V \setminus support(w)$. Else:
2. Find 3 vertices $x, y, z \in support(w)$ s.t. $C(x) = C(z) \neq C(y)$ and y lies between x and z.
   (a) $\varepsilon \leftarrow \min\{w(x), w(y), w(z)\}$
   (b) $w_2(v) = \begin{cases} \varepsilon & \text{if } v \in \{x, y, z\} \\ 0 & \text{otherwise.} \end{cases}$
   (c) $w_1 \leftarrow w - w_2$
   (d) $X \leftarrow$ 3-string-APPROX$(S, C|_{support(w_1)}, w_1)$

Note that a (partial) coloring of a string is not convex iff the condition in 2 holds. It is also easy to see that w_2 is 3-effective, since any cover Y must contain at least one vertex from any triplet described in condition 2, hence $w_2(Y) \geq \varepsilon$ while $w_2(V) = 3\varepsilon$.

The above algorithm cannot serve for approximating convex tree coloring since in a tree the condition in 2 might not hold even if $V \setminus support(w)$ is not a cover. In the following algorithm we generalize this condition to one which must hold in any non-convex coloring of a tree, in the price of increasing the approximation ratio from 3 to 4.

4-tree-APPROX:
Given an instance of the convex weighted tree problem (T, C, w):

1. If $V \setminus support(w)$ is a cover then $X \leftarrow V \setminus support(w)$. Else:
2. Find two pairs of (not necessarily distinct) vertices (x_1, x_2) and (y_1, y_2) in $support(w)$ s.t. $C(x_1) = C(x_2) \neq C(y_1) = C(y_2)$, and $carrier(\{x_1, x_2\}) \cap carrier(\{y_1, y_2\}) \neq \emptyset$:
   (a) $\varepsilon \leftarrow \min\{w(x_i), w(y_i)\}$, $i = \{1, 2\}$
   (b) $w_2(v) = \begin{cases} \varepsilon & \text{if } v \in \{x_1, x_2, y_1, y_2\} \\ 0 & \text{otherwise.} \end{cases}$
   (c) $w_1 \leftarrow w - w_2$
   (d) $X \leftarrow$ 4-tree-APPROX$(S, C|_{support(w_1)}, w_1)$

The algorithm is correct since if there are no two pairs as described in step 2, then $V \setminus support(w)$ is a cover. Also, it is easy to see that w_2 is 4-effective. Hence the above algorithm returns a cover with weight at most $4 \cdot OPT(T, C, w)$.

We now describe algorithm 3-tree-APPROX. Informally, the algorithm uses an iterative method, in the spirit of the local ratio technique, which approximates the solution of the input (T, C, w) by reducing it to (T', C', w_1) where $|support(w_1)| < |support(w)|$. Depending on the given input, this

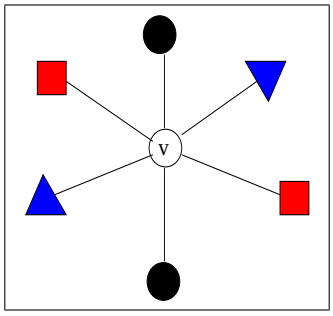

Figure 4: Case 2: a vertex v is contained in 3 different carriers.

reduction is either of the local ratio type (via an appropriate 3-effective weight function) or, the input graph is replaced by a smaller one which preserves the optimal solutions.

3-tree-APPROX(T, C, w)
On input (T, C, w) of a weighted colored tree, do the following:

1. If $V \setminus support(w)$ is a cover then $X \leftarrow V \setminus support(w)$. Else:
2. $(T', C', w_1) \leftarrow REDUCE(T, C, w)$. \The function $REDUCE$ guarantees that $|support(w_1)| < |support(w)|$
   (a) $X' \leftarrow$ 3-tree-APPROX(T', C', w_1).
   (b) $X \leftarrow UPDATE(X', T)$. \The function $UPDATE$ guarantees that if X' is a 3-approximation to (T', C', w_1), then X is a 3-approximation to (T, C, w).

Next we describe the functions $REDUCE$ and $UPDATE$, by considering few cases. In the first two cases we employ the local ratio technique.

Case 1: $support(w)$ contains three vertices x, y, z such that y lies on the path from x to z and $C(x) = C(z) \neq C(y)$.

In this case we use the same reduction of 3-string-APPROX: Let $\varepsilon = \min\{w(x), w(y), w(z)\} > 0$. Then $REDUCE(T, C, w) = (T, C|_{support(w_1)}, w_1)$, where $w_1(v) = w(v)$ if $v \notin \{x, y, z\}$, else $w_1(v) = w(v) - \varepsilon$. The same arguments which implies the correctness of 3-string-APPROX implies that if X' is a 3-approximation for (T', C', w_1), then it is also a 3-approximation for (T, C, w), thus we set $UPDATE(X', T) = X'$.

Case 2: Not Case 1, and T contains a vertex v such that $v \in \cap_{i=1}^{3} carrier(d_i, C)$ for three distinct colors d_1, d_2 and d_3 (see Figure 4).

In this case we must have that $w(v) = 0$ (else Case 1 would hold), and there are three *designated pairs* of vertices $\{x_1, x_2\}, \{y_1, y_2\}$ and $\{z_1, z_2\}$ such that $C(x_i) = d_1$, $C(y_i) = d_2, C(z_i) = d_3 (i = 1, 2)$, and v lies on each of the three paths connecting these three pairs (see Figure 4). We set $REDUCE(T, C, w) = (T, C|_{support(w_1)}, w_1)$, where w_1 is defined as follows.

Let $\varepsilon = \min\{w(x_i), w(y_i), w(z_i) : i = 1, 2\}$. Then $w_1(v) = w(v)$ if v is not in one of the designated

pairs, else $w_1(v) = w(v) - \varepsilon$. Finally, any cover for (T, C) must contain at least two vertices from the set $\{x_i, y_i, z_i : i = 1, 2\}$, hence $w - w_1 = w_2$ is 3-effective, and by the local ratio theorem we can set $UPDATE(X', T) = X'$.

Case 3: Not Cases 1 and 2.

Root T at some vertex r and for each color d let r_d be the root of the subtree $carrier(d, C)$. Let d_0 be a color for which the root r_{d_0} is farthest from r. Let $\bar{T}$ be the subtree of T rooted at r_{d_0}, and let $\hat{T} = T \setminus \bar{T}$ (see Figure 5). By the definition of r_{d_0}, no vertex in $\hat{T}$ is colored by d_0, and since Case 2 does not hold, there is a color d' so that $\{d_0\} \subseteq C(V(\bar{T})) \subseteq \{d_0, d'\}$.

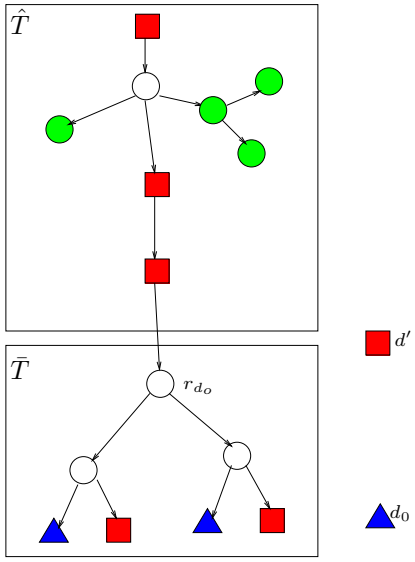

Figure 5: Case 3: Not case 1 nor 2. $\bar{T}$ is the subtree rooted at r_{d_0} and $\hat{T} = T \setminus \bar{T}$.

Subcase 3a: $C(V(\bar{T})) = \{d_0\}$ (see Figure 6).

In this case, $carrier(d_0, C) \cap carrier(d, C) = \emptyset$ for each color $d \neq d_0$, and for each optimal solution X it holds that $X \cap V(\bar{T}) = \emptyset$. We set $REDUCE(T, C, w) \leftarrow (\hat{T}, C|_{V(\hat{T})}, w|_{V(\hat{T})})$. The 3-approximation X' to (T', C', w_1) is also a 3-approximation to (X, C, w), thus $UPDATE(X', T) = X'$.

We are left with the last case, depicted in Figure 7.

Subcase 3b: $C(V(\bar{T})) = \{d_0, d'\}$.

In this case we have that $r_{d_0} \in carrier(d_0, C) \cap carrier(d', C)$ and $w(r_{d_0}) = 0$ (since Case 1 does not hold). Note that, in this case, $V(\bar{T})$ must contain at least two vertices colored by d_0 and at least one vertex colored by d', altogether at least three colored vertices.

Informally, $REDUCE(T, C, w)$ modifies the tree T by replacing the subtree $\bar{T}$ by a smaller subtree $\bar{T}_0$, which contains only two vertices, and which encodes three possible recolorings of $\bar{T}$, one of which must be used in an optimal recoloring of T. Thus the tree T', resulted from replacing $\bar{T}$ by $\bar{T}_0$ in the tree T, has smaller support than T.

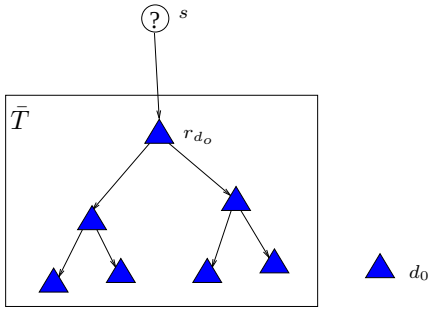

Figure 6: Case 3a: No vertices of $\hat{T}$ are colored by d'.

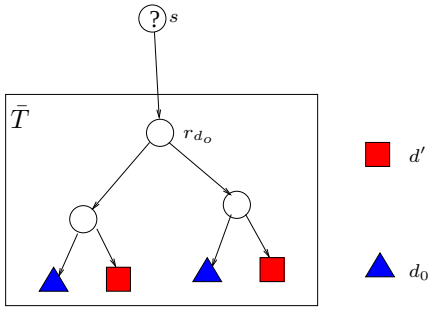

Figure 7: Case 3b: $r_{d_0} \in T_{d_0} \cap carrier(d')$;

Observation 5.3 *There is an optimal convex coloring C' which satisfies the following: $C'(v) \neq d_0$ for any $v \in V(\hat{T})$, and $C'(v) \in \{d_0, d'\}$ for any $v \in V(\bar{T})$.*

Proof. Let $\hat{C}$ be an expanding optimal convex recoloring of (T, C). We will show that there is an optimal coloring C' satisfying the lemma such that $cost(C') \leq cost(\hat{C})$. Since $\hat{C}$ is expanding and optimal, at least one vertex in $\bar{T}$ is colored either by d_0 or by d'. Let U be a set of vertices in $\bar{T}$ so that $carrier(U)$ is a maximal subtree all of whose vertices are colored by colors not in $\{d_0, d'\}$. Then $carrier(U)$ must have a neighbor u in $\bar{T}$ s.t. $\hat{C}(u) \in \{d_0, d'\}$. Changing the colors of the vertices in U to $\hat{C}(u)$ does not increase the cost of the recoloring. This procedure can be repeated until all the vertices of $\bar{T}$ are colored by d_0 or by d'. A similar procedure can be used to change the color of all the vertices in $\hat{T}$ to be different from d_0. It is easy to see that the resulting coloring C' is convex and $cost(C') \leq cost(\hat{C})$. ■

The function $REDUCE$ in Subcase 3b is based on the following observation: Let C' be any optimal recoloring of T satisfying Observation 5.3, and let s be the parent of r_{d_0} in T. Then $C'|_{V(\bar{T})}$, the restriction of the coloring C' to the vertices of $\bar{T}$, depends only on whether $carrier(d', C')$ intersects $V(\hat{T})$, and in this case if it contains the vertex s. Specifically, $C'_{V(\bar{T})}$ must be one of the three colorings of $V(\bar{T})$, C_{high}, C_{medium} and C_{min}, according to the following three scenarios:

1. $carrier(d', C') \cap V(\hat{T}) \neq \emptyset$ and $s \notin carrier(d', C')$. Then it must be the case that C' colors

all the vertices in $V(\bar{T})$ by d_0. This coloring of $\bar{T}$ is denoted as C_{high}.

2. $carrier(d', C') \cap V(\hat{T}) \neq \emptyset$ and $s \in carrier(d', C')$. Then $C'|_{\bar{T}}$ is a coloring of minimal possible cost of $\bar{T}$ which either equals C_{high} (i.e. colors all vertices by d_0), or otherwise colors r_{d_0} by d'. This coloring of $\bar{T}$ is called C_{medium}.

3. $carrier(d', C') \cap V(\hat{T}) = \emptyset$. Then $C'|_{\bar{T}}$ must be an optimal convex recoloring of $\bar{T}$ by the two colors d_0, d'. This coloring of $\bar{T}$ is called C_{min}.

We will show soon that the colorings C_{high}, C_{medium} and C_{min} above can be computed in linear time. The function $REDUCE$ in Subcase 3b modifies the tree T by replacing $\bar{T}$ by a subtree $\bar{T}_0$ with only 2 vertices, r_{d_0} and v_0, which encodes the three colorings $C_{high}, C_{medium}, C_{min}$. Specifically, $REDUCE(T, C, w) = (T', C', w_1)$ where (see Figure 8):

- T' is obtained from T by replacing the subtree $\bar{T}$ by the subtree $\bar{T}_0$ which contains two vertices: a root r_{d_0} with a single descendant v_0.
- $w_1(v) = w(v)$ for each $v \in V(\hat{T})$. For r_{d_0} and v_0, w_1 is defined as follows: $w_1(r_{d_0}) = cost(C_{medium}) - cost(C_{min})$ and $w_1(v_0) = cost(C_{high}) - cost(C_{min})$.
- $C'(v) = C(v)$ for each $v \in V(\hat{T})$; if $w_1(r_{d_0}) > 0$ then $C'(r_{d_0}) = d_0$ and if $w_1(v_0) > 0$ then $C'(v_0) = d'$. (If $w_1(u) = 0$ for $u \in \{r_{d_0}, v_0\}$, then $C'(u)$ is undefined).

Figure 8 illustrates $REDUCE$ for case 3b. In the figure, C_{high} requires overwriting all d' vertices and therefore costs 3, C_{medium} requires overwriting one d_0 vertex and costs 2 and C_{min} is the optimal coloring for $\bar{T}$ with cost 1. The new subtree $\bar{T}_0$ reflects these weight with $w_1(r_{d_0}) = C_{medium} - C_{min} = 1$ and $w_1(v_0) = C_{high} - C_{min} = 2$.

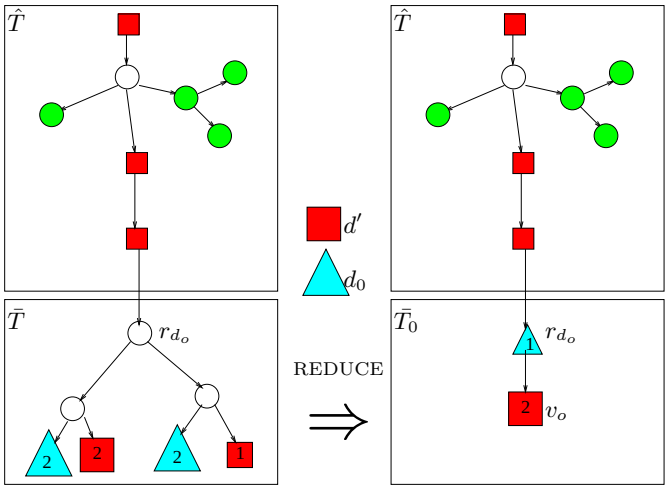

Figure 8: $REDUCE$ of case 3b: $\bar{T}$ is replaced with $\bar{T}_0$ where $w_1(r_{d_0}) = C_{medium} - C_{min} = 1$ and $w_1(v_0) = C_{high} - C_{min} = 2$.

Claim 5.4 $OPT(T', C', w_1) = OPT(T, C, w) - cost(C_{min})$.

Proof. We first show that $OPT(T', C', w_1) \leq OPT(T, C, w) - cost(C_{min})$. Let C^* be an optimal recoloring of C satisfying Observation 5.3, and let $X^* = \mathcal{X}(C^*)$. By the discussion above, we may assume that $C^*|_{V(\bar{T})}$ has one of the forms C_{high}, C_{medium} or C_{min}. Thus, $X^* \cap V(\bar{T})$ is either $\mathcal{X}(C_{high}), \mathcal{X}(C_{medium})$ or $\mathcal{X}(C_{min})$. We map C^* to a coloring C' of T' as follows: for $v \in V(\hat{T})$, $C'(v) = C^*(v)$. C' on r_{d_0} and v_0 is defined as follows:

- If $C^*|_{V(\bar{T})} = C_{high}$ then $C'(r_{d_0}) = C'(v_0) = d_0$, and $cost(C'|_{V(\bar{T})}) = w_1(v_0)$;
- If $C^*|_{V(\bar{T})} = C_{medium}$ then $C'(r_{d_0}) = C'(v_0) = d'$, and $cost(C'|_{V(\bar{T})}) = w_1(r_{d_0})$;
- If $C^*|_{V(\bar{T})} = C_{min}$ then $C'(r_{d_0}) = d_0$, $C'(v_0) = d'$, and $cost(C'|_{V(\bar{T})}) = 0$.

Note that in all three cases, $cost(C') = cost(C^*) - cost(C_{min})$.

The proof of the opposite inequality $OPT(T, C, w) - cost(C_{min}) \leq OPT(T', C', w_1)$ is similar. ■

Corollary 5.5 *C^* is optimal recoloring of (T, C, w) iff C' is an optimal recoloring of (T', C', w_1).*

We now can define the $UPDATE$ function for Subcase 3b: Let $X' = 3\text{-}tree\text{-}APPROX(T', C', w_1)$. Then X' is a disjoint union of the sets $\hat{X}' = X' \cap V(\hat{T})$ and $\bar{X}'_0 = X' \cap V(\bar{T}_0)$. Moreover, $\bar{X}'_0 \in \{\{r_{d_0}\}, \{v_0\}, \emptyset\}$. Then $X \leftarrow UPDATE(X') = \hat{X}' \cup \bar{X}'$, where $\bar{X}'$ is $\mathcal{X}(C_{high})$ if $\bar{X}'_0 = \{r_{d_0}\}$, is $\mathcal{X}(C_{medium})$ if $\bar{X}'_0 = \{v_0\}$, and is $\mathcal{X}(C_{min})$ if $\bar{X}'_0 = \emptyset$. Note that $w(X) = w(X') + cost(C_{min})$. The following inequalities show that if $w_1(X')$ is a 3-approximation to $OPT(T', C', w_1)$, then $w(X)$ is a 3-approximation to $OPT(T, C, w)$:

$$\begin{aligned} w(X) &= w_1(X') + cost(C_{min}) \leq 3OPT(T', C', w_1) + cost(C_{min}) \\ &< 3(OPT(T', C', w_1) + cost(C_{min})) = 3OPT(T, C, w) \end{aligned}$$

5.1 A Linear Time Algorithm for Subcase 3b

In Subcase 3b we need to compute C_{high}, C_{medium} and C_{min}. The computation of C_{high} is immediate. C_{medium} and C_{min} can be computed by the following simple, linear time algorithm that finds a minimal cost convex recoloring of a bi-colored tree, under the constraint that the color of a given vertex r is predetermined to one of the two colors.

Let the weighted colored tree (T, C, w) and the vertex r be given, and let $\{d_1, d_2\} = C(T)$. For $i \in \{1, 2\}$, let C_i be a minimal cost convex recoloring which sets the color of r to d_i (note that a coloring with minimum cost in $\{C_1, C_2\}$ is an optimal convex recoloring of (T, C)). We illustrate the computation of C_1 (the computation of C_2 is similar):

Compute for every edge $e = (u \rightarrow v)$ a cost defined by

$$cost(e) = w(\{v' : v' \in T(v) \text{ and } C(v') = d_1\}) + w(\{v' : v' \in [T \setminus T(v)] \text{ and } C(v') = d_2\})$$

where $T(v)$ is the subtree rooted at v. This can be done by one post order traversal of the tree. Then, select the edge $e^* = (u_0 \rightarrow v_0)$ which minimizes this cost, and set $C_1(w) = d_2$ for each $w \in T(v_0)$, and $C_1(w) = d_1$ otherwise.

5.2 Correctness and complexity

We now summarize the discussion of the previous section to show that the algorithm terminates and return a cover X which is a 3-approximation for (T, C, w).

Let $(T = (V, E), C, w)$ be an input to 3-tree-APPROX. if $V \setminus support(w)$ is a cover then the returned solution is optimal. Else, in each of the cases, $REDUCE(T, C, w)$ reduces the input to (T', C', w_1) such that $|support(w_1)| < |support(w)|$, hence the algorithm terminates within at most $n = |V|$ iterations. Also, as detailed in the previous subsections, the function $UPDATE$ guarantees that if X' is a 3-approximation for (T', C', w_1) then X is a 3-approximation to (T, C, w). Thus after at most n iterations the algorithm provides a 3-approximation to the original input.

Checking whether Case 1, Case 2, Subcase 3a or Subcase 3b holds at each stage requires $O(cn)$ time for each of the cases, and computing the function $REDUCE$ after the relevant case is identified requires linear time in all cases. Since there are at most n iterations, the overall complexity is $O(cn^2)$. Thus we have

Theorem 5.6 *Algorithm 3-tree-APPROX is a polynomial time 3-approximation algorithm for the minimum convex recoloring problem.*

6 Discussion and Future Work

In this work we showed two approximation algorithms for colored strings and trees, respectively. The 2-approximation algorithm relies on the technique of penalizing a colored string and the 3-approximation algorithm for the tree extends the local ratio technique by allowing dynamic changes in the underlying graph.

Few interesting research directions which suggest themselves are:

- A problem has a *polynomial approximation scheme* [11, 15], or is *fully approximable* [20], if for each $\varepsilon > 0$ it can be $(1 + \varepsilon)$-approximated in $p_\varepsilon(n)$ time for some polynomial p_ε. Are the problems of optimal convex recoloring of trees or strings fully approximable, (or equivalently have a polynomial approximation scheme)?
- Alternatively, can any of these two problems be shown to be APX-hard [24]?
- The algorithms presented here apply only to uniform models. The *non uniform model*, motivated by weighted maximum parsimony [21], assumes that the cost of assigning color d to vertex v is given by an arbitrary nonnegative number $cost(v, d)$ (note that, formally, no initial coloring C is assumed in this cost model). In this model $cost(C')$ is defined only for a total recoloring C', and is given by the sum $\sum_{v \in V} cost(v, C'(v))$. Finding non-trivial approximation results for this model is challenging.

7 Acknowledgments

We would like to thank Reuven Bar Yehuda, Arie Freund and Dror Rawitz for very helpful discussions.

References

[1] R. Agrawala and D. Fernandez-Baca. Simple algorithms for perfect phylogeny and triangulating colored graphs. *International Journal of Foundations of Computer Science*, 7(1):11–21, 1996.

[2] V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM J. on Discrete Mathematics*, 12:289–297, 1999.

[3] R. Bar-Yehuda. One for the price of two: A unified approach for approximating covering problems. *Algorithmica*, 27:131–144, 2000.

[4] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.

[5] A. Ben-Dor, N. Friedman, and Z. Yakhini. Class discovery in gene expression data. In *RECOMB*, pages 31–38, 2001.

[6] M. Bittner and et.al. Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, 406(6795):536–40, 2000.

[7] H.L. Bodlaender, M.R. Fellows, and T. Warnow. Two strikes against perfect phylogeny. In *ICALP*, pages 273–283, 1992.

[8] A. Dress and M.A. Steel. Convex tree realizations of partitions. *Applied Mathematics Letters,*, 5(3):3–6, 1992.

[9] D. Fernndez-Baca and J. Lagergren. A polynomial-time algorithm for near-perfect phylogeny. *SIAM Journal on Computing*, 32(5):1115–1127, 2003.

[10] W. M. Fitch. A non-sequential method for constructing trees and hierarchical classifications. *Journal of Molecular Evolution*, 18(1):30–37, 1981.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[12] L.A. Goldberg, P.W. Goldberg, C.A. Phillips, Z Sweedyk, and T. Warnow. Minimizing phylogenetic number to find good evolutionary trees. *Discrete Applied Mathematics*, 71:111–136, 1996.

[13] D. Gusfield. Efficient algorithms for inferring evolutionary history. *Networks*, 21:19–28, 1991.

[14] A. Hirsh, A. Tsolaki, K. DeRiemer, M. Feldman, and P. Small. From the cover: Stable association between strains of mycobacterium tuberculosis and their human host populations. *PNAS*, 101:4871–4876, 2004.

[15] D. S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problem*. PWS Publishing Company, 1997.

[16] S. Kannan and T. Warnow. Inferring evolutionary history from DNA sequences. *SIAM J. Computing*, 23(3):713–737, 1994.

[17] S. Kannan and T. Warnow. A fast algorithm for the computation and enumeration of perfect phylogenies when the number of character states is fixed. *SIAM J. Computing*, 26(6):1749–1763, 1997.

[18] S. Moran and S. Snir. Convex recoloring of strings and trees: Definitions, hardness results and algorithms. In *WADS*, 2005.

[19] S. Moran and S. Snir. Efficient approximation of convex recolorings. In *APPROX: the 8th. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2005.

[20] A. Paz and S. Moran. Non deterministic polynomial optimization probems and their approximabilty. *Theoretical Computer Science*, 15:251–277, 1981. Abridged version: Proc. of the 4th ICALP conference, 1977.

[21] D. Sankoff. Minimal mutation trees of sequences. *SIAM Journal on Applied Mathematics*, 28:35–42, 1975.

[22] C. Semple and M.A. Steel. *Phylogenetics*. Oxford University Press, 2003.

[23] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9(1):91–116, 1992.

[24] V. Vazirani. *Approximation Algorithms*. Springer, Berlin, germany, 2001.