Efficient Approximation of Convex Recolorings

Shlomo Moran†  Sagi Snir‡

February 1, 2008

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 arise 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 [19], 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 string.

In this paper we continue the work of [19], 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 $O(cn^2)$-time 3-approximation algorithm for convex recoloring of trees.

---

* A preliminary version of the results in this paper appeared in [18].
† Computer Science dept., Technion, Haifa 32000, Israel. moran@cs.technion.ac.il
‡ Department of Mathematics, 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 labelled 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 $C$ 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 make 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, 17, 12, 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 [19], 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 [19] is the non-uniform model, where the cost of coloring vertex $v$ by a color $d$ is an arbitrary nonnegative number $\text{cost}(v, d)$.

It was shown in [19] 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 penalties can be computed in \(O(cn)\) time, and once a penalty is computed, a recoloring whose cost is smaller than the penalty is computed in linear time. The 2-approximation follows by showing that for a string, the penalty is at most twice the cost of an optimal convex recoloring. This last result 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 [8], 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 \(\sum_{d \in \mathcal{C}} (n_b(C, d) - 1)\). 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 \(\text{Domain}(C)\). 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 \(V\), \(\text{carrier}(U)\) is the minimal subtree that contains \(U\). For a colored tree \((T, C)\) and a color \(d \in \mathcal{C}\), \(\text{carrier}_T(C, d)\) (or \(\text{carrier}(C, d)\) 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 \(\text{carrier}(C, d) \cap \text{carrier}(C, d') = \emptyset\). 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.

\[
\mathcal{X}_C(C') = \{v \in V : [v \in \text{Domain}(C)] \land [(v \notin \text{Domain}(C')) \lor (C(v) \neq C'(v))]\}.
\]

With each recoloring \(C'\) of \(C\) we associate a cost, denoted as \(\text{cost}_C(C')\) (or \(\text{cost}(C')\) when \(C\) is under-
Proof. Let $C$ model, the coloring expanding optimal convex recoloring of $C$ minimized. We shall prove that an optimal recoloring of $C$ 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.

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 show that for strings it is at least half the optimal cost, and then we use this fact to obtain 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:

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

Informally, when the vertices in $U$ induce a subtree, $\text{penalty}_{C, d}(U)$ 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 $C(v) = d$. 

The penalty of a given convex recoloring is sums of the penalties of every colored block: Let \( C' \) be a convex recoloring of \( C \). Then:

\[
\text{penalty}_{C}(C') = \sum_{d \in C} \text{penalty}_{C,d}(C' - 1(d))
\]

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** \( \text{penalty}(C') = 2\text{cost}(C') \)

**Proof.** From the definitions we have

\[
\text{penalty}(C') = \sum_{d \in C} \text{penalty}_{C,d}(C' - 1(d)) = 2\text{cost}(C')
\]

As can be seen in Figure 1, \( \text{penalty}(C') = 6 \) while \( \text{cost}(C') = 3 \).

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

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

**Corollary 3.2** For any recoloring \( C^* \) of \( C \),

\[
\sum_{d \in C} p^*_d \leq \sum_{d \in C} \text{penalty}_d(C') = 2\text{cost}(C').
\]

**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 \( (\sum_{d \in C} p^*_d)/2 \). For example, any convex recoloring of the tree in Figure 2 will recolor at least one of the big lateral blocks in the tree, while \( (\sum_{d \in C} p^*_d)/2 \) 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.
determined as follows: Let \( \hat{C} \) 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 \( v_i \in B_d \) 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 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 \( d_j \). The observation follows.

Thus it remains to prove

**Lemma 1** \( \text{cost}(\tilde{C}) < \sum_{d \in C} p_d' \).

**Proof.** Let \( v_i \) be a vertex which contributes to \( \text{cost}(\tilde{C}) \). Then \( C(v_i) = d \) and \( \tilde{C}(v_i) = d' \) for some distinct \( d', d \). By the algorithm, either \( v_i \in B_{d'} \), or \( v_i \) is free. In the first case \( v_i \) contributes to both \( p_d' \) and \( p_d'' \), and in the 2nd it contributes to \( p_d'' \). The inequality is strict since in each block \( F_j \) there is at least one vertex for which the former case holds.

---

### 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 \( \text{support}(w) = \{ v \in V : w(v) > 0 \} \).

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 the vertices in \( X \)). Thus, if \( C' \) is a convex recoloring of \((T, C)\), then \( X_C(C') \), 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 \( X(C') \subseteq X \) in case it is, can be done in \( O(n \cdot n_c) \) time. Also, the cost of a recoloring \( C' \) is \( w(X(C')) \).

Therefore, finding an optimal convex total recoloring of \( C \) is polynomially equivalent to finding an optimal cover \( X \), or equivalently a partial convex recoloring \( C' \) of \( C \) so that \( w(X(C')) = w(X) \) is minimized.

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 \((V, \Sigma \subseteq 2^V, w : V \rightarrow \mathbb{R}^+)\), and the goal is to find a subset \( X \in \Sigma \) such that \( w(X) \) is minimized, i.e. \( w(X) = \text{OPT}(V, \Sigma, w) = \min_{Y \in \Sigma} w(Y) \) (in our context \( V \) is the set of vertices, and \( \Sigma \) 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 \( w_1, w_2 \):

\[
\text{OPT}(V, \Sigma, w_1) + \text{OPT}(V, \Sigma, w_2) \leq \text{OPT}(V, \Sigma, w_1 + w_2)
\]

Now, given our initial weight function \( w \), we select \( w_1, w_2 \) s.t. \( w_1 + w_2 = w \) and \( |\text{support}(w_1)| < |\text{support}(w)| \). We first apply the algorithm to find an \( r \)-approximation to \((V, \Sigma, w_1)\) (in particular, if \( V \setminus \text{support}(w_1) \) is a cover, then it is an optimal cover to \((V, \Sigma, w_1)\)). Let \( X \) be the solution returned for \((V, \Sigma, w_1)\), and assume that \( w_1(X) \leq r \cdot \text{OPT}(V, \Sigma, w_1) \). If we could also guarantee that \( w_2(X) \leq r \cdot \text{OPT}(V, \Sigma, w_2) \) 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 \( w_2(X) \leq r \cdot \text{OPT}(V, \Sigma, w_2) \) is that \( w_2 \) is \( r \)-effective, that is: for every \( X \in \Sigma \) it holds that \( w_2(X) \leq r \cdot \text{OPT}(V, \Sigma, w_2) \) (note that if \( V \in \Sigma \), the above is equivalent to requiring that \( w_2(V) \leq r \cdot \text{OPT}(V, \Sigma, w_2) \)).
Theorem 5.2 \[\text{Given } X \in \Sigma \text{ s.t. } w_1(X) \leq r \cdot \text{OPT}(V, \Sigma, w_1). \text{ If } w_2 \text{ is } r\text{-effective, then } w(X) = w_1(X) + w_2(X) \leq r \cdot \text{OPT}(V, \Sigma, w).\]

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 to convex weighted string problem \((S, C, w)\):

1. If \(V \setminus \text{support}(w)\) is a cover then \(X \leftarrow V \setminus \text{support}(w)\). Else:
   2. Find 3 vertices \(x, y, z \in \text{support}(w)\) 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) \(X \leftarrow \text{3-string-APPROX}(S, C|_{\text{support}(w_1)}, w_1)\)

Note that if a (partial) coloring of a string is not convex then the condition in 2 must hold. 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 \(V \setminus \text{support}(w)\) 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 to convex weighted tree problem \((T, C, w)\):

1. If \(V \setminus \text{support}(w)\) is a cover then \(X \leftarrow V \setminus \text{support}(w)\). Else:
   2. Find two pairs of (not necessarily distinct) vertices \((x_1, x_2)\) and \((y_1, y_2)\) in \(\text{support}(w)\) s.t. \(C(x_1) = C(x_2) \neq (y_1) = C(y_2)\), and \(\text{carrier}([x_1, x_2]) \cap \text{carrier}([y_1, y_2]) \neq \emptyset\):
      (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) \(X \leftarrow \text{4-tree-APPROX}(S, C|_{\text{support}(w_1)}, w_1)\)

The algorithm is correct since if there are no two pairs as described in step 2 then \(V \setminus \text{support}(w)\) 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 \(4 \cdot \text{OPT}(T, C, w)\).

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
In this case we must have that \( w \) is not Case 1, and Case 2:

same arguments which implies the correctness of 3-string-APPROX implies that if \( \varepsilon \) lies on each of the three paths connecting these three pairs (see Figure 4). We set the \( \text{support} \) of \( (T', C', w_1) \) where \( |\text{support}(w_1)| < |\text{support}(w)| \). Depending on the given input, this 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 \( V \setminus \text{support}(w) \) is a cover then \( X \leftarrow V \setminus \text{support}(w) \). Else:

2. \((T', C', w_1) \leftarrow \text{REDUCE}(T, C, w)\). \( \setminus \) The function \( \text{REDUCE} \) guarantees that \( |\text{support}(w_1)| < |\text{support}(w)| \)

   (a) \( X' \leftarrow 3\text{-tree-APPROX}(T', C', w_1) \).

   (b) \( X \leftarrow \text{UPDATE}(X', T) \). \( \setminus \) The function \( \text{UPDATE} \) 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 \( \text{REDUCE} \) and \( \text{UPDATE} \), by considering few cases. In the first two cases we employ the local ratio technique.

Case 1: \( \text{support}(w) \) 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 \( \text{REDUCE}(T, C, w) = (T, C_{\text{support}(w_1)}, w_1) \), 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 \( \text{UPDATE}(X', T) = X' \).

Case 2: Not Case 1, and \( T \) contains a vertex \( v \) such that \( v \in \cap_{i=1}^{3} \text{carrier}(d_i, C) \) 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 \( \text{REDUCE}(T, C, w) = (T, C_{\text{support}(w)}, w_1) \), 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 $\text{carrier}(d, C)$. Let $d_0$ be a color for which the root $r_{d_0}$ is farthest from $r$. Let $\hat{T}$ be the subtree of $T$ rooted at $r_{d_0}$, and let $\check{T} = T \setminus \hat{T}$ (see Figure 5). By the definition of $r_{d_0}$, no vertex in $\check{T}$ is colored by $d_0$, and since Case 2 does not hold, there is a color $d'$ so that \{d_0\} $\subseteq C(V(\hat{T}))$ $\subseteq$ \{d_0, d'\}.

![Figure 5: Case 3: Not case 1 nor 2. $\hat{T}$ is the subtree rooted at $r_{d_0}$ and $\check{T} = T \setminus \hat{T}$.](image)

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

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

![Figure 6: Case 3a: No vertices of $\hat{T}$ are colored by $d'$.](image)

Subcase 3b: $r_{d_0} \in \text{carrier}(d_0, C) \cap \text{carrier}(d', C)$. See Figure 7.

Observe that in this case we have $w(r_{d_0}) = 0$ and $|\text{support}(w) \cap V(\check{T})| \geq 3$, since $V(\check{T})$ must contain at
least two vertices colored $d_0$ and at least one vertex colored $d'$. Figure 7 illustrates this case.

**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 $v \in V(\hat{T})$.

**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 $\text{cost}(C') \leq \text{cost}(\hat{C})$. Since $\hat{C}$ is expanding and optimal, at least one vertex in $\hat{T}$ is colored either by $d_0$ or by $d'$. Let $U$ be a set of vertices in $\hat{T}$ so that $\text{carrier}(U)$ is a maximal subtree all of whose vertices are colored by colors not in $\{d_0, d'\}$. Then $\text{carrier}(U)$ must have a neighbor $u$ in $\hat{T}$ s.t. $\hat{C}(u) \in \{d_0, d'\}$. Change the colors of the vertices in $U$ to $\hat{C}(u)$. This procedure can be repeated until all the vertices of $\hat{T}$ are colored by $d_0$ or by $d'$, without increasing the cost of the recoloring. 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 $\text{cost}(C') \leq \text{cost}(\hat{C})$.

The function $\text{REDUCE}$ 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 $C'|_{V(\hat{T})}$, the restriction of the coloring $C'$ to the vertices of $\hat{T}$, depends only on whether $\text{carrier}(d', C')$ intersects $V(\hat{T})$, and in this case if it contains the vertex $s$. Specifically, $C'|_{V(\hat{T})}$ must be one of the three colorings of $V(\hat{T})$, $C_{\text{high}}, C_{\text{medium}}$, and $C_{\text{min}}$, according to the following three scenarios:

1. $\text{carrier}(d', C') \cap V(\hat{T}) \neq \emptyset$ and $s \notin \text{carrier}(d', C')$. Then it must be the case that $C'$ colors all the vertices in $V(\hat{T})$ by $d_0$. This coloring of $\hat{T}$ is denoted as $C_{\text{high}}$.

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

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

We will show soon that the colorings $C_{\text{high}}, C_{\text{medium}}$, and $C_{\text{min}}$ above can be computed in linear time. The function $\text{REDUCE}$ in Subcase 3b modifies the tree $T$ by replacing $\hat{T}$ by a subtree $T_0$ with only 2 vertices, $r_{d_0}$ and $v_0$, which encodes the three colorings $C_{\text{high}}, C_{\text{medium}}, C_{\text{min}}$. Specifically, $\text{REDUCE}(T, C, w) = (T', C', w_1)$ where (see Figure 8):
Proof.  Figure 8 illustrates REDUCE for case 3b. In the figure, \( C_{\text{high}} \) requires overwriting all \( d' \) vertices and therefore costs 3, \( C_{\text{medium}} \) requires overwriting one \( d_0 \) vertex and costs 2 and \( C_{\text{min}} \) is the optimal coloring for \( \hat{T} \) with cost 1. The new subtree \( T_0 \) reflects these weight with \( w_1(r_{d_0}) = C_{\text{medium}} - C_{\text{min}} = 1 \) and \( w_1(v_0) = C_{\text{high}} - C_{\text{min}} = 2 \).

![Diagram](image)

Figure 8: REDUCE of case 3b: \( \hat{T} \) is replaced with \( T_0 \) where \( w_1(r_{d_0}) = C_{\text{medium}} - C_{\text{min}} = 1 \) and \( w_1(v_0) = C_{\text{high}} - C_{\text{min}} = 2 \).

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

Proof. We first show that \( \text{OPT}(T', C', w_1) \leq \text{OPT}(T, C, w) - \text{cost}(C_{\text{min}}) \). Let \( C^* \) be an optimal recoloring of \( C \) satisfying Observation 5.3 and let \( X^* = \chi(C^*) \). By the discussion above, we may assume that \( C^*|_{V(\hat{T})} \) has one of the forms \( C_{\text{high}}, C_{\text{medium}} \) or \( C_{\text{min}} \). Thus, \( X^* \cap V(\hat{T}) \) is either \( \chi(C_{\text{high}}), \chi(C_{\text{medium}}) \) or \( \chi(C_{\text{min}}) \). 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 \( C^*|_{V(\hat{T})} = C_{\text{high}} \) then \( C'(r_{d_0}) = C'(v_0) = d_0 \), and \( \text{cost}(C'|_{V(\hat{T})}) = w_1(v_0) \);
- If \( C^*|_{V(\hat{T})} = C_{\text{medium}} \) then \( C'(r_{d_0}) = C'(v_0) = d' \), and \( \text{cost}(C'|_{V(\hat{T})}) = w_1(r_{d_0}) \);
- If \( C^*|_{V(\hat{T})} = C_{\text{min}} \) then \( C'(r_{d_0}) = d_0, C'(v_0) = d' \), and \( \text{cost}(C'|_{V(\hat{T})}) = 0 \).
Note that in all three cases, $\text{cost}(C') = \text{cost}(C^*) - \text{cost}(C_{\text{min}})$.

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

$$w(X) = w_1(X') + \text{cost}(C_{\text{min}}) \leq 3\text{OPT}(T', C', w_1) + \text{cost}(C_{\text{min}})$$

$$< 3(\text{OPT}(T', C', w_1) + \text{cost}(C_{\text{min}})) = 3\text{OPT}(T, C, w)$$

### 5.1 A Linear Time Algorithm for Subcase 3b

In subcase 3b we need to compute $C_{\text{high}}, C_{\text{medium}}$ and $C_{\text{min}}$. The computation of $C_{\text{high}}$ is immediate. $C_{\text{medium}}$ and $C_{\text{min}}$ 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 $C_i$ the 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

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

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 $V \setminus \text{support}(w)$ is a cover then the returned solution is optimal. Else, in each of the cases, $\text{REDUCE}(T, C, w)$ reduces the input to $(T', C', w_1)$ such that $|\text{support}(w_1)| < |\text{support}(w)|$, hence the algorithm terminates within at most $n = |V|$ iterations. Also, as detailed in the previous subsections, the function $\text{UPDATE}$ guarantees that 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 $\text{REDUCE}$ 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:

- Can our approximation ratios for strings or trees be improved.
- This is a more focused variant of the previous item. A problem has a polynomial approximation scheme [11, 15], or is fully approximable [20], if for each \( \varepsilon \) it can be \( \varepsilon \)-approximated in \( p_\varepsilon(n) \) time for some polynomial \( p_\varepsilon \). Are the problems of optimal convex recoloring of trees or strings fully approximable, (or equivalently have a polynomial approximation scheme)?
- Alternatively, can any of the variant be shown to be APX-hard [?]
- 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 \( \text{cost}(v, d) \) (note that, formally, no initial coloring \( C \) is assumed in this cost model). In this model \( \text{cost}(C') \) is defined only for a total recoloring \( C' \), and is given by the sum \( \sum_{v \in V} \text{cost}(v, C'(v)) \). Finding non-trivial approximation results for this model is challenging.

### 7 Acknowledgments

We wish to thank Reuven Bar Yehuda and Mike Steel for many 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. Fernandez-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. Technical Report CS-2003-13, Technion, November 2003.

[19] S. Moran and S. Snir. Convex recoloring of strings and trees: Definitions, hardness results and algorithms. submitted, 2004.

[20] A. Paz and S. Moran. Non deterministic polynomial optimization problems and their approximability. 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.