

## On the Linear-Cost Subtree-Transfer Distance between Phylogenetic Trees<sup>1</sup>

B. DasGupta,<sup>2</sup> X. He,<sup>3</sup> T. Jiang,<sup>4</sup> M. Li,<sup>5</sup> and J. Tromp<sup>6</sup>

**Abstract.** Different phylogenetic trees for the same group of species are often produced either by procedures that use diverse optimality criteria [16] or from different genes [12] in the study of molecular evolution. Comparing these trees to find their *similarities* and *dissimilarities* (i.e., *distance*) is thus an important issue in computational molecular biology. Several distance metrics including the *nearest neighbor interchange* (nni) distance and the *subtree-transfer* distance have been proposed and extensively studied in the literature. This article considers a natural extension of the subtree-transfer distance, called the *linear-cost* subtree-transfer distance, and studies the complexity and efficient approximation algorithms for this distance as well as its relationship to the nni distance. The linear-cost subtree-transfer model seems more suitable than the (unit-cost) subtree-transfer model in some applications. The following is a list of our results:

1. The linear-cost subtree-transfer distance is in fact *identical* to the nni distance on unweighted phylogenies.
2. There is an algorithm to compute an optimal linear-cost subtree-transfer sequence between unweighted phylogenies in  $O(n \cdot 2^{O(d)})$  time, where  $d$  denotes the linear-cost subtree-transfer distance. Such an algorithm is useful when  $d$  is small.
3. Computing the linear-cost subtree-transfer distance between two weighted phylogenetic trees is NP-hard, provided we allow multiple leaves of a tree to share the same label (i.e., the trees are not necessarily uniquely labeled).
4. There is an efficient approximation algorithm for computing the linear-cost subtree-transfer distance between weighted phylogenies with performance ratio 2.

**Key Words.** Evolutionary trees, Approximation algorithms, Lower bounds.

**1. Introduction.** The evolution history of organisms is often conveniently represented by trees, called *phylogenetic trees* or simply *phylogenies*. Such a tree has uniquely labeled

<sup>1</sup> The results reported here also form a subset of the results that appeared in *Proc. 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1997, pp. 427–436, New Orleans. The remaining results of the conference paper which do not appear in this paper will be published separately. B. DasGupta was supported by a CGAT (Canadian Genome Analysis and Technology) grant. The work was done while he was at the University of Waterloo and McMaster University. X. He was supported in part by CGAT and NSF Grant 9205982. His work was done while visiting at the University of Waterloo. T. Jiang was supported in part by NSERC Operating Grant OGP0046613 and CGAT and the work was done while he was visiting at the University of Washington. M. Li was supported by NSERC Operating Grant OGP0046506 and CGAT. J. Tromp was supported by an NSERC International Fellowship and CGAT and the work was done while he was with the University of Waterloo.

<sup>2</sup> Department of Computer Science, Rutgers University, Camden, NJ 08102, USA. bhaskar@crab.rutgers.edu.

<sup>3</sup> Department of Computer Science, SUNY-Buffalo, Buffalo, NY 14260, USA. xinhe@cs.buffalo.edu.

<sup>4</sup> Department of Computer Science, McMaster University, Hamilton, Ontario, Canada L8S 4K1. jiang@maccs.mcmaster.ca.

<sup>5</sup> Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. mli@math.uwaterloo.ca.

<sup>6</sup> CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands. tromp@cwi.nl.

Received May 8, 1997; revised February 20, 1998. Communicated by D. Gusfield and M.-Y. Kao.



Fig. 1. T

leaves and  
is known, a  
different of  
developed,  
distance [10  
usually dep  
result, in p  
thus of inte  
method on  
is also rou  
phylogenie

Several  
in the liter  
tance [19],  
two subtrees  
nni operati  
 $D_{nni}(T_1, T_2)$   
erations rec  
computing  
recently. It  
logarithmic  
in [3] and [

The pro  
context. W  
sequences  
the course  
described b  
models hav  
of phyloge  
of the sequ  
transferrin  
event betw  
corresponc  
the compu  
number of  
authors sh  
simple app

It is rel  
occur with  
that have

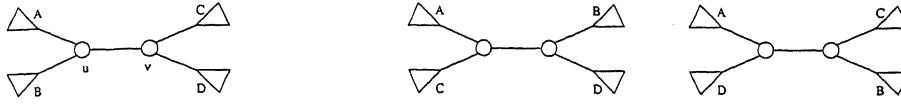


Fig. 1. The two possible nni operations on an internal edge  $(u, v)$ : exchange  $B \leftrightarrow C$  or  $B \leftrightarrow D$ .

leaves and unlabeled internal nodes, can be *unrooted* or *rooted* if the evolutionary origin is known, and usually has internal nodes of degree 3. Over the past few decades, many different objective criteria and algorithms for reconstructing phylogenies have been developed, including (not exhaustively) parsimony [6], [9], [22], compatibility [17], distance [10], [21], and maximum likelihood [6], [7], [1]. The outcomes of these methods usually depend on the data and the amount of computational resources applied. As a result, in practice they often lead to different trees on the same set of species [16]. It is thus of interest to compare phylogenies produced by different methods, or by the same method on different data, for similarity and discrepancy. The comparison of phylogenies is also routinely performed in simulation studies where people analyze reconstructed phylogenies against the true ones.

Several metrics for measuring the distance between phylogenies have been proposed in the literature. Among these measures, the *nearest neighbor interchange* (nni) distance [19], [20], [25] has perhaps received the most attention. An nni operation swaps two subtrees that are separated by an internal edge  $(u, v)$ , as shown in Figure 1. The nni operation is said to *operate* or *perform* on this internal edge. The nni distance,  $D_{nni}(T_1, T_2)$ , between two trees  $T_1$  and  $T_2$  is defined as the minimum number of nni operations required to transform one tree into the other. The computational complexity of computing the nni distance has puzzled the research community for nearly 25 years until recently. It is shown in [3] that the nni distance is NP-hard to compute. Some efficient logarithmic-ratio approximation algorithms for the nni distance have also been proposed in [3] and [18].

The problem of computing distance between phylogenies also arises in a different context. When the data is in the form of some molecular sequences of organisms and the sequences have been subject to events such as *recombination* or *gene conversion* during the course of evolution, the evolutionary history of the sequences cannot be adequately described by a single tree. In an attempt to solve this problem, more general evolutionary models have been proposed including the network model [24] and a model using a list of phylogenetic trees [12], [13]. In the latter, every tree corresponds to a specific region of the sequences, and each tree can be obtained from the preceding tree on the list by transferring some subtrees from one place to another. Figure 2 shows a recombination event between two sequences and Figure 3 shows a *subtree-transfer* operation and its corresponding recombination event. The parsimony model in [12] and [13] requires the computation of the subtree-transfer distance between two trees, i.e., the minimum number of subtrees we need to move to transform one tree into the other. In [15] the authors show that computing the subtree-transfer distance is NP-complete and give a simple approximation algorithm with approximation ratio 3.

It is relevant in practice to discriminate among subtree-transfer operations as they occur with different frequencies. For example, it is reasonable to assume that sequences that have only diverged recently give rise to more recombinations than sequences that

er by procedures  
ecular evolution.  
nportant issue in  
nterchange (nni)  
e literature. This  
subtree-transfer  
ce as well as its  
in the (unit-cost)

ted phylogenies.  
een unweighted  
stance. Such an

rees is NP-hard,  
not necessarily

ifer distance be-

r represented  
quely labeled

ual ACM-SIAM  
f the conference  
rted by a CGAT  
e University of  
1982. His work  
ERC Operating  
of Washington.  
upported by an  
e University of

rab.rutgers.edu.  
Yalo.edu.  
ida L8S 4K1.

ida N2L 3G1.

Kao.

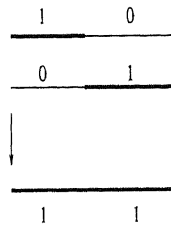


Fig. 2. Recombination event between two sequences. The genetic material (thick lines), that is in one sequence after recombination, was in two sequences just before the recombination.

diverged many generations ago [13], [14]. In this case we can charge each subtree-transfer operation a cost equal to the distance (number of nodes passed) that the subtree has moved in the current tree. The *linear-cost* subtree-transfer distance,  $D_{st}(T_1, T_2)$ , between two trees  $T_1$  and  $T_2$  is then the minimum total cost required to transform  $T_1$  into  $T_2$  by subtree-transfer operations. Clearly, both subtree-transfer and linear-cost subtree-transfer models can also be used as alternative measures for comparing phylogenies generated by different phylogeny reconstruction methods.

A phylogeny may also have *weights* on its edges, where an edge weight (more popu-

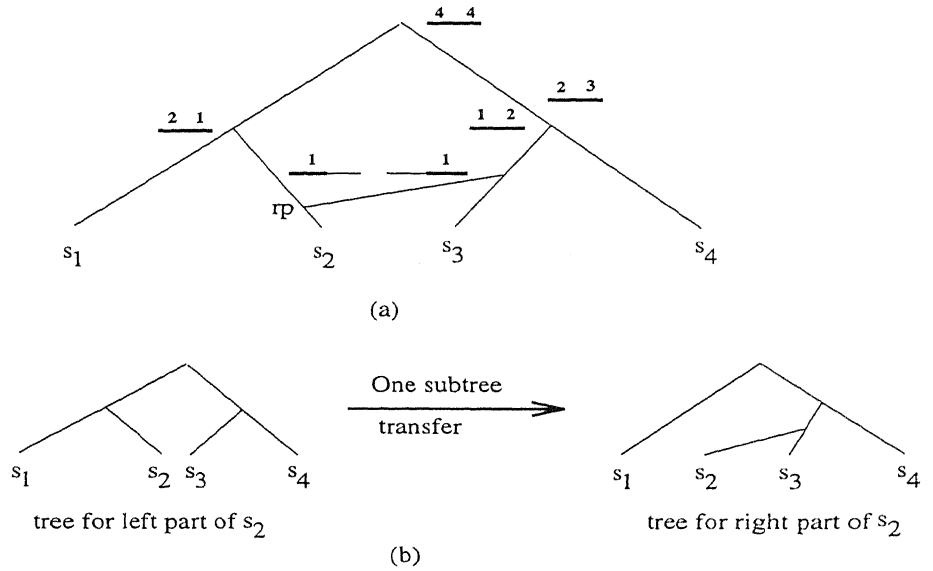


Fig. 3. [13] Recombination event at point  $rp$  in (a) corresponds to transferring subtree  $s_2$  in (b). The genetic material (thick lines), that is in one sequence after recombination, was in two sequences just before the recombination. The two sets of numbers (on the thick lines) correspond to the two evolutionary histories (as shown in (b)) of two parts of the sequences. For example, in the evolutionary tree for the second parts of the sequences (rightmost tree in (b)), a common ancestor of  $s_2, s_3, s_4$  is found going back in time; hence the second number of the thick line in the second row is 3.

larly known as *branch length* in genetics) could represent the evolutionary distance along the edge. Many phylogeny reconstruction methods, including the distance and maximum likelihood methods, actually produce weighted phylogenies. Comparison of weighted phylogenies has recently been studied in [16]. The distance measure adopted is based on the difference in the partitions of the leaves induced by the edges in both trees, and has the drawback of being somewhat insensitive to the tree topologies [8]. Just like the nni model [4], the linear-cost subtree-transfer model can be naturally extended to weighted phylogenies: a moving subtree is charged for the weighted distance it travels. Intuitively this measure is more sensitive to the tree topologies than the one in [16].

In this paper we study the computational complexity and approximation algorithms for linear-cost subtree-transfer distance on both unweighted and weighted phylogenies. The rest of the paper is organized as follows. In Section 1.1 we show that the linear-cost subtree-transfer distance is in fact *identical* to the nni distance on unweighted phylogenies. As a result, the complexity and approximation results for the nni distance reported in [3] and [4] directly apply to the linear-cost subtree-transfer distance on unweighted phylogenies too. Section 2 presents an algorithm to compute an optimal linear-cost subtree-transfer sequence on unweighted phylogenies in time  $O(n \cdot 2^{O(d)})$ , where  $d$  stands for the linear-cost subtree-transfer distance between the trees involved. In Section 3 we formalize the extension of the linear-cost subtree-transfer distance on weighted phylogenies and prove the following results:

- We show that computing the linear-cost subtree-transfer distance between two weighted trees is NP-hard, provided we allow multiple leaves of a tree to share the same label (i.e., the trees are not necessarily uniquely labeled).
- We devise an approximation algorithm for the linear-cost subtree-transfer distance between weighted trees with performance ratio 2.

The results presented in this paper form a part of the results in [3]. The remaining results in [3] deal with the proof of NP-hardness of computing the nni distance for (uniquely) labeled trees, as well as extending the nni distance for weighted phylogenies, and will be published separately [4].

Unless otherwise mentioned explicitly, the following definitions are used uniformly throughout the rest of the paper. All the trees in this paper are trees with *internal nodes* of degree 3 and with *unique* labels on leaves. We will mention it explicitly if a tree has nonuniquely labeled leaves or unlabeled leaves. An edge of a tree is *external* if it is incident on a leaf, otherwise it is *internal*. Finally, two weighted trees are considered equal iff there is an isomorphism between them preserving topology and edge weights (and leaf labels, if they are labeled).

1.1. *Nni and Subtree-Transfer on Unweighted Phylogenies.* Surprisingly, although they are studied in parallel for very different reasons, we demonstrate here that the linear-cost subtree-transfer distance and the nni distance are very closely related for unweighted phylogenies.

LEMMA 1. *The linear-cost subtree-transfer distance is identical to the nni distance on unweighted phylogenies.*

PROOF. Observe that an nni move is just a restricted subtree-transfer where a subtree is only moved across a single edge. (In Figure 1 the first exchange can alternatively be seen as moving node  $v$  together with subtree  $C$  past node  $u$  toward subtree  $A$ , or vice versa.) On the other hand, when all internal nodes have degree 3, a subtree-transfer over a distance  $d$  can always be simulated by a series of  $d$  nni moves. Hence the linear-cost subtree-transfer distance is in fact *identical* to the nni distance on unweighted phylogenies.  $\square$

As a result, all the results on computing the nni distance reported in [3] and [4] directly apply to the linear-cost subtree-transfer problem on unweighted phylogenies also. In particular, this means that, for any two unweighted phylogenies  $T_1$  and  $T_2$ :

- Computing  $D_{st}(T_1, T_2)$  is NP-hard.
- $D_{st}(T_1, T_2)$  can be approximated within a logarithmic factor in polynomial time.

**2. An Efficient Exact Algorithm for Small Subtree-Transfer Distance.** The result in this section concerns computing  $D_{st}(T_1, T_2)$  exactly, where  $T_1$  and  $T_2$  are unweighted phylogenies. In practice, the trees to be compared usually have small subtree-transfer distances between them and it is of interest to devise efficient algorithms for computing an optimal subtree-transfer sequence when the  $D_{st}(T_1, T_2)$  is small, say at most  $d$ . An  $n^{O(d)}$  algorithm for this problem is trivial. With careful inspection, one can derive an algorithm that runs in  $O(n^{O(1)} \cdot d^{O(d^2)})$  time. It turns out that by using the results in [23] and [18], we can improve this asymptotically to  $O(n \cdot 2^{2^{1d/2}})$  time.

DEFINITION 1. Let  $T_1$  and  $T_2$  be the two trees being compared. An edge  $e_1 \in T_1$  is *good* if there is another edge  $e_2 \in T_2$  such that  $e_1$  and  $e_2$  partition the leaf labels of  $T_1$  and  $T_2$  identically;  $e_1$  is *bad* otherwise.

The proof of the following lemma can be found in [5] which deals with computing strict consensus trees.

LEMMA 2. [5] *Let  $T_1$  and  $T_2$  be two trees, each with  $n$  leaves. Then the set of good edges of  $T_1$  (with respect to  $T_2$ ) can be enumerated in  $O(n)$  time.*

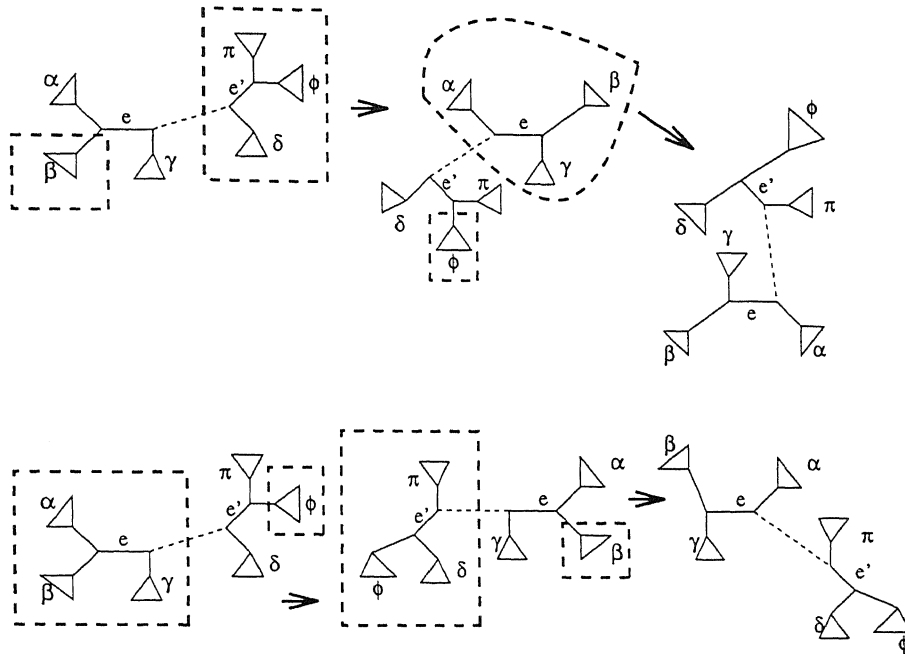
We also need the following rather straightforward observation.

OBSERVATION 3. Let  $e$  and  $e'$  be two edges of a binary tree  $T$  which are not adjacent to each other. Then performing an nni operation  $\sigma$  across  $e$  followed by an nni operation  $\sigma'$  across  $e'$  is the same as performing the nni operation  $\sigma'$  across  $e'$  followed by the nni operation  $\sigma$  across  $e$ .

See Figure 4 for an illustration of the Observation 3.

THEOREM 4. *Suppose that  $D_{st}(T_1, T_2) \leq d$ . An optimal sequence of subtree-transfer operations transforming  $T_1$  into  $T_2$  can be computed in  $O(n \cdot 2^{2^{1d/2}})$  time.*

PROOF. Since the linear-cost subtree-transfer distance is identical to the nni distance on unweighted trees, we choose, for convenience of understanding, to describe how to



**Fig. 4.** Illustration of Observation 3. The subtrees swapped in each nni operation are shown by thick dashed rectangles. The final trees in the two nni sequences are the same.  $\alpha, \beta, \gamma, \delta, \pi$  and  $\phi$  are subtrees. The portion of the tree connecting  $e$  to  $e'$  is shown by a dashed line.

find an optimal nni sequence (which is in fact an optimal subtree-transfer sequence). We know that  $T_1$  contains at least one (and at most  $d$ ) bad edge. Moreover, assume that these bad edges form  $t$  connected components  $B_1, \dots, B_t$  ( $1 \leq t \leq d$ ). As observed in [18], for an optimal nni transformation, sometimes one or more nni operations are needed across a good internal edge of  $T_1$ . Consider the set of at most  $d - 1$  good edges in  $T_1$  across which at least one nni operation is performed in an optimal nni sequence. This set of good edges forms at most  $d - 1$  connected components in  $T_1$ . Consider any one such connected component  $S$ . Since good edges in  $T_1$  and  $T_2$  partition the trees in a similar manner, it is very easy to see that there must be at least one connected component  $B_i$  sharing a vertex with  $S$ .

Using this observation, we can devise the following algorithm:

**Algorithm NNI-d**

For every choice of integers  $k_1, \dots, k_t \geq 0, 1 \leq \sum_{i=1}^t k_i \leq d$  do  
 For every choice of connected subgraphs  $A_1, \dots, A_t$  of  $T_1$  such that  $A_i$  has at most  $k_i$  internal edges<sup>7</sup> and contains the component  $B_i$  do

<sup>7</sup> Nni operations cannot obviously be performed across external edges.

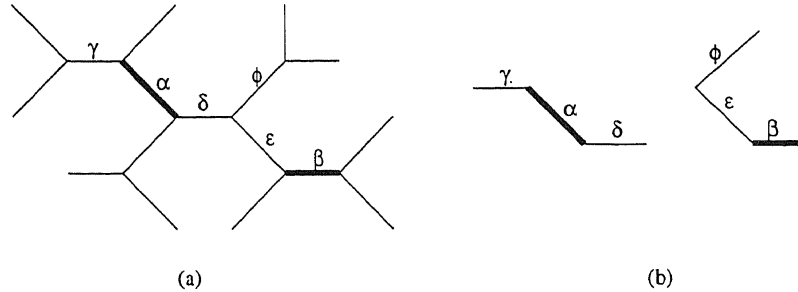


Fig. 5. Illustration of how Algorithm NNI-d works ( $d = 6, k_1 = k_2 = 3, t = 2$ ).

Examine all sequences of nni transformations across edges of all  $A_i$ 's such that no more than  $k_i$  nni operations are performed across the edges of  $A_i$ . Among all sequences examined, select the one of shortest length that transforms  $T_1$  into  $T_2$ .

Figure 5 illustrates how the algorithm works. Figure 5(a) shows two bad edges  $\alpha, \beta$  in  $T_1$  (shown by thick lines) forming two connected components ( $t = 2$ ). In Figure 5(b) we show one choice of two connected subgraphs containing  $k_1$  and  $k_2$  edges, and including the edges  $\alpha$  and  $\beta$ , respectively. For each connected subgraph, Algorithm NNI-d computes all possible nni sequences such that at most three nni are performed across edges of each connected subgraph.

Now we analyze the running time of the above algorithm. The following countings are crucial for the analysis.

- There are at most

$$\sum_{i=1}^d \binom{i+t-1}{i} \leq \sum_{i=1}^d \binom{i+d-1}{i} \leq \sum_{i=1}^d 2^{i+d-1} < 2^{2d}$$

choices for the integers  $k_1, \dots, k_t$ .

- Note that any connected subgraph of  $k$  edges including a fixed edge ( $k > 1$ ) can be represented by a rooted binary tree on  $k + 2$  nodes (the root corresponding to the middle of the fixed edge), hence there are at most  $C_{k+2} = (1/(k+3)) \binom{2k+4}{k+2} < 2^{2k}$  such subgraphs [2, page 262]. For  $k = 1$ , there is exactly  $1 < 2^{2k}$  such subgraph. Hence, it follows that the total number of choices for the subgraphs  $A_1, \dots, A_t$  (for any particular value of  $k_1, \dots, k_t$ ) is at most  $2^{\sum_{i=1}^t (2k_i)} \leq 2^{2d}$ .

Consider a particular choice of subgraphs  $A_1, A_2, \dots, A_t$  (with  $k_1, k_2, \dots, k_t$  edges, respectively). Let  $M_1, M_2, \dots, M_s$  be the  $s$  connected components of  $A_1 \cup A_2 \cup \dots \cup A_t$ . Assume that  $M_i$  has  $\ell_i$  edges ( $\sum_{i=1}^s \ell_i = \sum_{i=1}^t k_i \leq d$ ). Notice that we are required to perform at most  $\ell_i$  nni operations across the edges of  $M_i$ . Let  $m_i \leq \ell_i$  be the number of nni operations performed across the edges of  $M_i$ . Extend each  $M_i$  to  $M'_i$  by adding edges from  $T$  such that every degree 1 or degree 2 node of  $M_i$  (that is not a leaf of  $T$ ) is

of degree 3 in  $M'_i$ . Notice that  $M'_i$  has at most  $\ell_i + 3$  leaves. Lemma 1 of [18] states that the number of trees within an nni distance of  $m > 1$  from any given tree with  $n$  leaves is at most  $3^{n-2}2^{4m}$  (for  $m = 1$ , the number of such trees is obviously at most  $n - 3$ ). Hence, the total number of distinct nni operations we will need to consider for each connected subgraph  $M_i$  is at most  $3^{(\ell_i+3)-2}2^{4\ell_i} = 3^{\ell_i+1}2^{4\ell_i} < 2^{13\ell_i/2}$  if  $1 < m_i \leq \ell_i$ , and at most  $\ell_i < 2^{13\ell_i/2}$  if  $m_i = 1$ . By Observation 3, nni operations across the edges in  $M_i$  can be performed independently of the nni operations across the edges of  $M_j$  for  $i \neq j$ . Hence, the total number of nni operations across the edges of all  $M_i$ 's is at most  $2^{(13/2)\sum_{i=1}^d \ell_i} \leq 2^{13d/2}$ . Combining everything, the total number of nni operations we will need to consider is at most

$$\begin{array}{l} \text{Number of choices for the} \\ \text{integers } k_1, k_2, \dots, k_t \end{array} \times \begin{array}{l} \text{Number of choices of} \\ A_1, A_2, \dots, A_t \text{ for each} \\ \text{choice of } k_1, k_2, \dots, k_t \end{array} \times \begin{array}{l} \text{Number of nni op-} \\ \text{erations across the} \\ \text{edges of } M_1, M_2, \dots, M_t \end{array}$$

which is at most  $2^{2d} \times 2^{2d} \times 2^{13d/2} = 2^{21d/2}$ .

The set of all good edges of  $T_1$  can be found in  $O(n)$  time using Lemma 2, and this time bound is also sufficient to find the connected components of good edges. Using the adjacency-list representation of trees, updating a tree during a single nni operation can be easily done in  $O(1)$  time, and whether two trees are isomorphic can be easily checked in  $O(n)$  time. Hence, this algorithm finds an optimal nni sequence in  $O(n \cdot 2^{21d/2})$  time.  $\square$

**3. Linear-Cost Subtree-Transfer Distance on Weighted Phylogenies.** In this section we investigate the linear-cost subtree-transfer model on weighted phylogenies. Recall that the linear-cost subtree-transfer distance is identical to the nni distance on unweighted phylogenies. Below we formalize the linear-cost subtree-transfer model on weighted phylogenies.

Consider (unrooted) trees in which each edge  $e$  has a weight  $w(e) \geq 0$ . To ensure feasibility of transforming a tree into another, we require the total weight of all edges to equal one. A subtree-transfer is defined as follows. Select a subtree  $S$  of  $T$  at a given node  $u$  and select an edge  $e_4 \notin S$ . Split the edge  $e_4$  into two edges  $e_6$  and  $e_7$  with weights  $w(e_6)$  and  $w(e_7)$  ( $w(e_6), w(e_7) \geq 0, w(e_6) + w(e_7) = w(e_4)$ ), and move  $S$  to the common endpoint of  $e_6$  and  $e_7$ . Finally, merge the two remaining edges  $e_1$  and  $e_2$  adjacent to  $u$  into one edge  $e_5$  with weight  $w(e_5) = w(e_1) + w(e_2)$ . The cost of this subtree-transfer is the total weight of all the edges over which  $S$  is moved. Figure 6 gives an example. The subtree  $S$  is transferred to split the edge  $e_4$  to  $e_6$  and  $e_7$  such that  $w(e_6), w(e_7) \geq 0$  and  $w(e_6) + w(e_7) = w(e_4)$ ; finally, the two edges  $e_1$  and  $e_2$  are merged to  $e_5$  such that  $w(e_5) = w(e_1) + w(e_2)$ . The cost of transferring  $S$  is  $w(e_2) + w(e_3) + w(e_6)$ .

**3.1. Some Definitions and a Useful Lower Bound.** In this section we introduce some notations and a lower bound on the subtree-transfer distance which will be useful in subsequent proofs. For any tree  $T$ , let  $E(T)$  (resp.  $V(T)$ ) denote the edge set (resp. node set) of  $T$  and let  $L(T)$  denote the set of leaf nodes of  $T$ . An external edge of  $T$  incident on a leaf node  $a$  is denoted by  $e_T(a)$ . Let  $E_{\text{int}}(T)$  and  $E_{\text{ext}}(T)$  denote the set of internal and



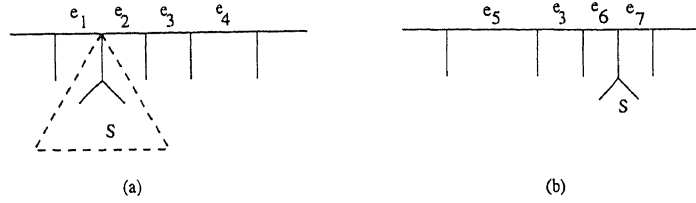


Fig. 6. Subtree-transfer on weighted phylogenies. Tree (b) is obtained from tree (a) with one subtree-transfer.

external edges of  $T$ , respectively. For a subset  $E' \subseteq E(T)$ , define  $w(E') = \sum_{e \in E'} w(e)$ . Define  $W_{\text{int}}(T) = w(E_{\text{int}}(T))$  and  $W_{\text{ext}}(T) = w(E_{\text{ext}}(T))$ .

Consider the transformation of tree  $T_1$  to tree  $T_2$  (hence  $L(T_1) = L(T_2)$ ). We partition  $E_{\text{ext}}(T_1)$  into three subsets as follows:

$$\begin{aligned} E_{\text{ext}, T_1 > T_2}(T_1) &= \{e_{T_1}(a) \mid w(e_{T_1}(a)) > w(e_{T_2}(a))\}, \\ E_{\text{ext}, T_1 = T_2}(T_1) &= \{e_{T_1}(a) \mid w(e_{T_1}(a)) = w(e_{T_2}(a))\}, \\ E_{\text{ext}, T_1 < T_2}(T_1) &= \{e_{T_1}(a) \mid w(e_{T_1}(a)) < w(e_{T_2}(a))\}, \\ W_{\text{ext}, T_1 > T_2}(T_1) &= \sum_{e_{T_1}(a) \in E_{\text{ext}, T_1 > T_2}(T_1)} w(e_{T_1}(a)) - w(e_{T_2}(a)). \end{aligned}$$

Similarly,  $E_{\text{ext}}(T_2)$  can be partitioned into  $E_{\text{ext}, T_1 > T_2}(T_2)$ ,  $E_{\text{ext}, T_1 = T_2}(T_2)$ , and  $E_{\text{ext}, T_1 < T_2}(T_2)$ .  $W_{\text{ext}, T_1 < T_2}(T_2)$  is defined analogously.

LEMMA 5.  $W_{\text{int}}(T_1) + W_{\text{ext}, T_1 > T_2}(T_1) = W_{\text{int}}(T_2) + W_{\text{ext}, T_1 < T_2}(T_2)$ .

PROOF. Since  $w(E_{\text{ext}, T_1 > T_2}(T_1)) = w(E_{\text{ext}, T_1 > T_2}(T_2)) + W_{\text{ext}, T_1 > T_2}(T_1)$ , we have

$$\begin{aligned} W_{\text{int}}(T_1) + w(E_{\text{ext}, T_1 = T_2}(T_1)) + w(E_{\text{ext}, T_1 < T_2}(T_1)) \\ + w(E_{\text{ext}, T_1 > T_2}(T_2)) + W_{\text{ext}, T_1 > T_2}(T_1) = w(T_1) = 1. \end{aligned}$$

Similarly, we have

$$\begin{aligned} W_{\text{int}}(T_2) + w(E_{\text{ext}, T_1 = T_2}(T_2)) + w(E_{\text{ext}, T_1 > T_2}(T_2)) \\ + w(E_{\text{ext}, T_1 < T_2}(T_1)) + W_{\text{ext}, T_1 < T_2}(T_2) = w(T_2) = 1. \end{aligned}$$

Since  $w(E_{\text{ext}, T_1 = T_2}(T_1)) = w(E_{\text{ext}, T_1 = T_2}(T_2))$ , the lemma follows from the above two equations.  $\square$

We next define the notion of good edge pairs in the following:

DEFINITION 2. Let  $e_1 \in E_{\text{int}}(T_1)$  and  $e_2 \in E_{\text{int}}(T_2)$ . Let  $T_1'$  and  $T_1''$  be the two subtrees of  $T_1$  partitioned by  $e_1$ . Let  $T_2'$  and  $T_2''$  be the two subtrees of  $T_2$  partitioned by  $e_2$ . The edges  $e_1$  and  $e_2$  are called a *good edge pair* of  $T_1$  and  $T_2$  iff the following two conditions hold:

1.  $L(T_1') = L(T_2')$  and  $L(T_1'') = L(T_2'')$ .

2. One of the following two conditions holds:

- (a)  $w(E(T_1')) \leq w(E(T_2')) < w(E(T_1')) + w(e_1)$ ; or  
 (b)  $w(E(T_2')) \leq w(E(T_1')) < w(E(T_2')) + w(e_2)$ .

The following lemma provides a lower bound on  $D_{st}(T_1, T_2)$  when  $T_1$  and  $T_2$  do not share good edge pairs.

LEMMA 6. *If  $T_1$  and  $T_2$  share no good edge pairs, then:*

- (1)  $D_{st}(T_1, T_2) \geq W_{int}(T_1) + W_{ext, T_1 > T_2}(T_1)$ .  
 (2)  $D_{st}(T_1, T_2) \geq W_{int}(T_2) + W_{ext, T_1 < T_2}(T_2)$ .

PROOF. We only prove (1). The proof of (2) follows from (1) and Lemma 5. For each edge  $e \in E(T_1)$ , we determine the minimum portion of  $e$  over which some subtrees of  $T_1$  must be transferred in order to transform  $T_1$  to  $T_2$ . First consider an edge  $e_1 \in E_{int}(T_1)$ . By the assumption of the lemma, there is no edge  $e_2$  in  $T_2$  such that  $e_1$  and  $e_2$  are a good pair. There are two cases:

Case 1. The partition of  $L(T_1)$  induced by  $e_1$  is different from the partition of  $L(T_2)$  induced by any edge in  $T_2$ . Then, in order to transform  $T_1$  to  $T_2$ , some leaf nodes of  $T_1$  must be transferred across the entire length of  $e_1$ .

Case 2. The partition of  $L(T_1)$  induced by  $e_1$  is the same as the partition of  $L(T_2)$  induced by an edge  $e_2$  in  $T_2$ . Let  $T_1'$  and  $T_1''$  be the two subtrees of  $T_1$  partitioned by  $e_1$ . Let  $T_2'$  and  $T_2''$  be the two subtrees of  $T_2$  partitioned by  $e_2$ , where  $L(T_1') = L(T_2')$  and  $L(T_1'') = L(T_2'')$ .

Case 2.1.  $w(E(T_2')) \geq w(E(T_1')) + w(e_1)$ . In this case, in order to transform  $T_1'$  to  $T_2'$ , some subtree in  $T_1'$  must be transferred across the entire length of  $e_1$ .

Case 2.2.  $w(E(T_1')) \geq w(E(T_2')) + w(e_2)$ . This implies  $w(E(T_1'')) + w(e_1) \leq w(E(T_2''))$ . In order to transform  $T_1''$  to  $T_2''$ , some subtree in  $T_1''$  must be transferred across the entire length of  $e_1$ .

In either case, some subtree of  $T_1$  must be transferred across the entire length of  $e_1$  with cost  $w(e_1)$ .

Next consider an edge  $e_{T_1}(a) \in E_{ext, T_1 > T_2}(T_1)$ . In order to transform  $e_{T_1}(a)$  to  $e_{T_2}(a)$ , a subtree of  $T_1$  must be transferred across a portion of  $e_{T_1}(a)$  of length  $w(e_{T_1}(a)) - w(e_{T_2}(a))$ . Thus:

$$D_{st}(T_1, T_2) \geq \sum_{e \in E_{int}(T_1)} w(e) + \sum_{e \in E_{ext, T_1 > T_2}(T_1)} [w(e_{T_1}(a)) - w(e_{T_2}(a))] = W_{int}(T_1) + W_{ext, T_1 > T_2}(T_1) \quad \square$$

REMARK. Assume that the given trees  $T_1$  and  $T_2$  are not uniquely labeled (i.e., a label may appear in more than one leaf). Extend the definition of good edge pairs by treating  $L(T)$  as the multiset of leaves for a tree  $T$  and considering the conditions  $L(T_1') = L(T_2')$  or  $L(T_1'') = L(T_2'')$  to hold if the corresponding multisets are identical. Assume that all the leaves are incident on zero-weight edges (i.e.,  $W_{int}(T_1) = W_{int}(T_2)$  and  $W_{ext}(T_1) = W_{ext}(T_2) = 0$ ), and that  $T_1$  and  $T_2$  share no good edge pairs. Then, in a manner very similar

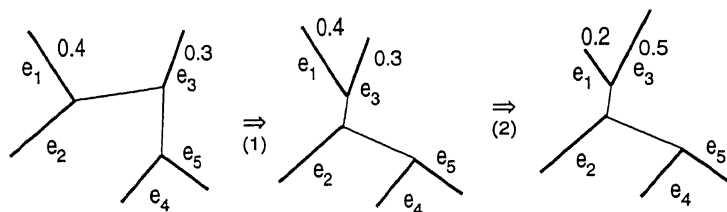


Fig. 7. The operation  $move(e_1, 0.2, e_3)$ . (1)  $e_2, e_4, e_5$  are assembled into a tree  $S$ ; (2)  $S$  is moved along  $e_1$  by a length of 0.2.

to the proof of Lemma 6, one can show that  $D_{st}(T_1, T_2) \geq W_{int}(T_1)$  (and that some subtree is transferred over every internal edge of  $T_1$  to transform  $T_1$  to  $T_2$ ).

We say that nodes connected by zero-weight edges are equivalent and call the resulting equivalence classes *supernodes*. Let  $e_1, \dots, e_k$  be all positive weight edges incident to a supernode  $o$ . With zero cost, we can reconnect the edges  $e_1, \dots, e_k$  by any subtree, consisting of only zero-weight edges. In particular, the following observation will be useful in our subsequent descriptions.

**OBSERVATION.** Let  $o$  be a supernode of  $T$ . Let  $e_1, \dots, e_k$  be all positive weight edges incident on  $o$ . Pick any  $e_i$  and  $e_j$ . We can assemble  $\{e_1, \dots, e_k\} - \{e_i, e_j\}$  into a single subtree  $S$  with zero cost; and then transfer  $S$  along  $e_i$  by a distance  $d \leq w(e_i)$ . The effect of this operation is that the edges  $e_1, \dots, e_k$  are still incident on a supernode, and a portion of  $e_i$  of length  $d$  is moved into  $e_j$ . The total cost of this operation is  $d$ . We denote this operation by  $move(e_i, d, e_j)$ . This operation can be implemented in  $O(k)$  time using the adjacency-list representation of the tree (where the weight of the edge is also stored in the adjacency list).

Figure 7 shows an example of this operation. In the figure the thin lines denote zero-weight edges and heavy lines denote positive-weight edges.

A tree  $T$  is called a *superstar* if all of its internal edges have zero weight. In other words, all external edges of a superstar  $T$  are incident to a single supernode.

**3.2. An NP-hardness Result.** It is open whether the linear-cost subtree-transfer problem is NP-hard for weighted phylogenies. However, we can show that the problem is NP-hard for weighted trees with nonuniquely labeled leaves.

**THEOREM 7.** Let  $T_1$  and  $T_2$  be two weighted trees with (not necessarily uniquely) labeled leaves. Then computing  $D_{st}(T_1, T_2)$  is NP-hard.

Our proof is by a reduction from the *Exact Cover by 3-Sets (X3C)* problem, which is defined as follows:

*Instance:*  $S = \{s_1, \dots, s_m\}$ , where  $m = 3q$ , and  $C_1, \dots, C_n$ , where  $C_i = \{s_{i_1}, s_{i_2}, s_{i_3}\} \subseteq S$ .

*Question:* Are there  $q$  disjoint sets  $C_{i_1}, \dots, C_{i_q}$  such that  $\bigcup_{j=1}^q C_{i_j} = S$ ?

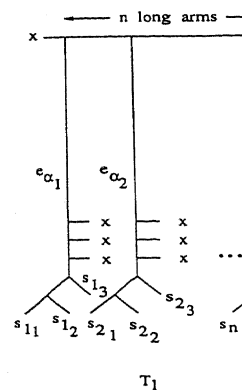


Fig. 8. Trees  $T_1$  and  $T_2$  use leaves. The notations for set weights are as follows:  $w(e_{\alpha_1}) = w(e_{\alpha_2}) = \dots = \nu$

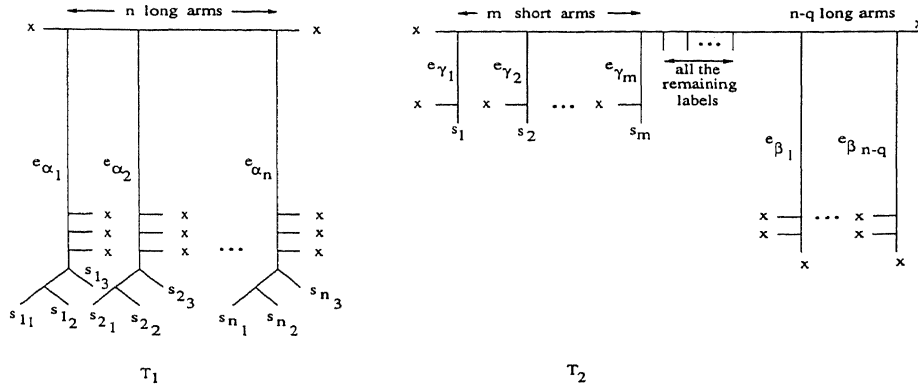
X3C is known to be NP-hard. The leaf labels (not necessarily unique) are used to describe subtree-transfers of to

**PROOF OF THEOREM 7.** Let  $T_1$  and  $T_2$  be two weighted trees with (not necessarily uniquely) labeled leaves. Each long (resp. short) leaf of  $T_1$  (resp.  $T_2$ ) is labeled with a leaf label. For notation, let  $w$  be the weight in the long arm of the  $i$ th long arm of  $T_2$ , there are  $n$  long arms in  $T_2$ , there are  $3n$  leaves of  $T_2$  is as follows:

- At the bottom of the long arm of  $T_2$ , there are  $3$  short arms.
- The remaining  $3n$  short arms (in any order) will be labeled with leaf labels.

Note that  $W_{int}(T_1) = W_{int}(T_2)$  if the trees  $T_1$  and  $T_2$  are constructed as above. This is the key to the NP-hardness proof.

**LEMMA 8.**  $D_{st}(T_1, T_2) = W_{int}(T_1)$



**Fig. 8.** Trees  $T_1$  and  $T_2$  used in the proof of Theorem 7. The leaf labels are shown beside the corresponding leaves. The notations for some of the internal edges are shown beside the corresponding edges. The edge weights are as follows:  $w(e_{\alpha_1}) = w(e_{\alpha_2}) = \dots = w(e_{\alpha_n}) = w(e_{\beta_1}) = w(e_{\beta_2}) = \dots = w(e_{\beta_{n-q}}) = 1/n$ ,  $w(e_{\gamma_1}) = w(e_{\gamma_2}) = \dots = w(e_{\gamma_m}) = 1/3n$ , and all other edges have zero weights.

X3C is known to be NP-complete [11]. We will construct two trees  $T_1$  and  $T_2$  with leaf labels (not necessarily unique), such that transforming from  $T_1$  into  $T_2$  requires subtree-transfers of total cost exactly one iff an exact cover of  $S$  exists.

**PROOF OF THEOREM 7.** Assume that an instance,  $S = \{s_1, s_2, \dots, s_m\}$  (with  $m = 3q$ ) and  $C_1, C_2, \dots, C_n$  (with  $|C_i| = 3$ ), of the X3C problem is given. We construct two weighted *labeled* (but not uniquely labeled) trees as shown in Figure 8.  $T_1$  has  $n$  long arms,  $\alpha_1, \dots, \alpha_n$ .  $T_2$  has  $n - q$  long arms,  $\beta_1, \dots, \beta_{n-q}$ , and  $m$  short arms,  $\gamma_1, \dots, \gamma_m$ . Each long (resp. short) arm consists of an edge of weight  $1/n$  (resp.  $1/3n$ ), with three leaves (resp. one leaf) labeled by the same label  $x$  ( $x \notin S$ ), connected to it as shown in Figure 8. For notational convenience, let  $e_{\alpha_i}$  (resp.  $e_{\beta_i}$ ,  $e_{\gamma_i}$ ) denote the edge of nonzero weight in the long arm  $\alpha_i$  (resp. in the long arm  $\beta_i$ , in the short arm  $\gamma_i$ ). In  $T_1$ , at the bottom of the  $i$ th long arm  $\alpha_i$ , we attach a subtree  $t_i$  consisting of three leaves, as shown in Figure 8, labeled by the three elements  $s_{i_1}$ ,  $s_{i_2}$ , and  $s_{i_3}$  of  $C_i$ . At the bottom of each long arm of  $T_2$ , there are no additional subtrees attached. The labeling of the remaining leaves of  $T_2$  is as follows:

- At the bottom of the  $i$ th short arm  $\gamma_i$ , we attach a leaf labeled by  $s_i$ .
- The remaining  $3n - m$  leaf labels (each leaf label is an element of  $S$ ) are associated (in any order) with the  $3n - m$  leaves in the middle of  $T_2$  between the long and the short arms.

Note that  $W_{\text{int}}(T_1) = W_{\text{int}}(T_2) = 1$  and  $W_{\text{ext}}(T_1) = W_{\text{ext}}(T_2) = 0$ . Also, notice that the trees  $T_1$  and  $T_2$  are not uniquely labeled. The following lemma proves the correctness of the NP-hardness reduction.

**LEMMA 8.**  $D_{\text{st}}(T_1, T_2) = 1$  iff there is a solution of the X3C problem.

The following lemma is needed in the proof of Lemma 8.

LEMMA 9.  $D_{\text{st}}(T_1, T_2) \geq 1$ . Moreover, if  $D_{\text{st}}(T_1, T_2) = 1$ , then, over any portion of any of the edges  $e_{\alpha_i}$ , exactly one subtree-transfer takes place.

PROOF. We first verify that every edge  $e_{\alpha_i}$  of  $T_1$  is not a good edge pair with any edge of  $T_2$ . Consider the edge  $e_{\alpha_i}$  ( $1 \leq i \leq n$ ). This edge partitions  $T_1$  into two trees  $T_1'$  and  $T_1''$ , where  $L(T_1')$  consists of six leaves labeled with  $s_{i_1}, s_{i_2}, s_{i_3}, x, x, x$  and  $L(T_1'')$  consists of the remaining leaves of  $T_1$ . Also, note that  $w(E(T_1')) = 0$  and  $w(E(T_1'')) = 1 - 1/n$ . Consider any edge  $e \in T_2$  partitioning  $T_2$  into two trees  $T_2'$  and  $T_2''$ . Since both the conditions  $L(T_1') = L(T_2')$  and  $L(T_1'') = L(T_2'')$  must be satisfied for  $e_{\alpha_i}$  to be a good edge pair with  $e$ , the only possibility for the edge  $e$  is to be the zero-weight edge between  $e_{\gamma_3}$  and  $e_{\gamma_4}$ . However, in that case,  $w(E(T_2')) = 1/n$  and  $w(E(T_2'')) = 1 - 1/n$ . Then, clearly both  $w(E(T_2')) > w(E(T_1'))$  and  $w(E(T_2'')) = w(E(T_1'')) + w(e_{\alpha_i})$  are true. Hence,  $e_{\alpha_i}$  is not a good edge pair with  $e$ .

Hence, from the Remark following the proof of Lemma 6, some subtree is transferred over every internal edge  $e_{\alpha_i}$  of  $T_1$ , and we get

$$D_{\text{st}}(T_1, T_2) \geq \sum_{i=1}^n w(e_{\alpha_i}) = 1.$$

The remaining part of the lemma is now straightforward. Assume that over a portion  $\gamma$  of some  $e_{\alpha_i}$ , more than one subtree is transferred ( $0 < w(\gamma) \leq w(e_{\alpha_i})$ ). Then  $D_{\text{st}}(T_1, T_2) \geq \sum_{i=1}^n w(e_{\alpha_i}) + w(\gamma) = 1 + w(\gamma) > 1$ .  $\square$

PROOF OF LEMMA 8. Suppose that there is an exact cover of  $S$ , say (without loss of generality)  $C_1, \dots, C_q$ . Then we transform  $T_1$  to  $T_2$  in the following manner:

- First we consider the corresponding long arms  $\alpha_1, \dots, \alpha_q$  in  $T_1$  and move the leaves of each subtree  $t_j$  ( $j = 1, \dots, q$ ) up in the following way as shown in Figure 9. Without loss of generality, we describe the procedure for  $\alpha_1$  only. Leave one of the three leaves, say  $s_{1_3}$ , with a leaf  $x$  at the bottom, and move the subtree containing the other two leaves  $s_{1_1}$  and  $s_{1_2}$  together with two leaves labeled  $x$  up by a distance  $1/3n$  (remember that we can use zero-weight edges to assemble many subtrees at a given node into one subtree with zero cost). Now leave one of these two leaves, say  $s_{1_1}$ , with a leaf labeled  $x$  there, and move the subtree containing leaves  $s_{1_2}$  and  $s_{1_3}$  together with two leaves labeled  $x$  up by a distance  $1/3n$ . Finally, move the subtree containing the leaf  $s_{1_1}$  and a leaf labeled  $x$  and the subtree containing the leaf  $s_{1_3}$  and a leaf labeled  $x$  together up by a distance  $1/3n$ . After this, we have created all the short arms of  $T_2$ , but not necessarily in the correct order. After some rearrangements of the short arms of total cost zero (since we move subtrees across zero-weight edges), we can create the short arms of  $T_2$  in the correct order.
- For each  $C_l$  not in the cover, we simply move the subtree containing the three leaves of the subtree  $t_l$  up by a distance  $1/n$  (see Figure 10). This already creates the remaining long arms of  $T_2$ . Now, with some extra rearrangements of total cost zero, we can create, in correct order, the leaves in the middle part of  $T_2$  between the long and the short arms.

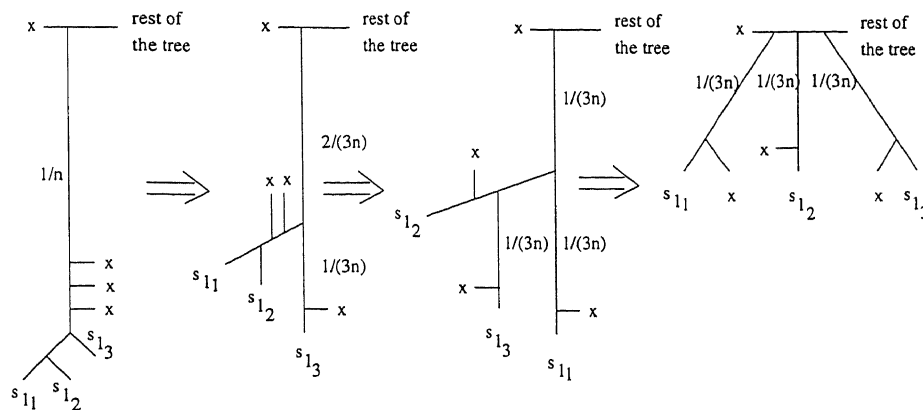


Fig. 9. Moving leaves of the long arm  $\alpha_1$ .

Hence, our conversion of  $T_1$  to  $T_2$  is complete. The total cost of conversion is  $q \cdot 1/n + (n - q) \cdot 1/n = 1$ , as promised.

Conversely, assume that there is no exact cover of  $S$ . Then, by Lemma 9, if  $D_{st}(T_1, T_2) = 1$ , then over any portion of any of the edges  $e_{\alpha_i}$ , at most one subtree transfer takes place. However, in that case, the only possible way to create the  $m$  short arms of  $T_2$  is to use exactly  $q$  long arms in  $T_1$ , which means there was an exact cover of  $S$ .  $\square$

REMARK. Since the X3C problem is NP-complete even if each element occurs in at most three sets [11], it follows that Theorem 7 holds even if every label, except only one label (label  $x$  in the proof), occurs in at most three leaves.

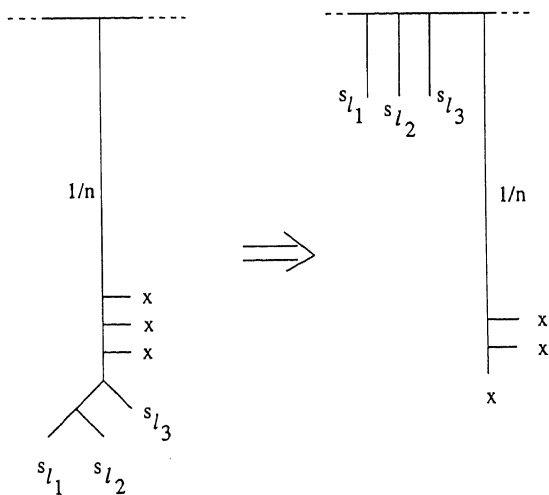


Fig. 10. Moving leaves of the long arm  $\alpha_l$  for every  $C_l$  not in cover.

3.3. *An Approximation Algorithm.* In this section, we prove the following theorem.

**THEOREM 10.** *For any two (uniquely-labeled) weighted phylogenies  $T_1$  and  $T_2$ ,  $D_{\text{st}}(T_1, T_2)$  can be approximated to within a factor of 2 in  $O(n^2)$  time.*

We are now ready to describe our algorithm. First we consider the special case when  $T_1$  and  $T_2$  do not have any good edge pairs. Algorithm DST, as described below, approximates  $D_{\text{st}}(T_1, T_2)$  to within a factor of 2. The algorithm transforms  $T_1$  into a superstar  $T'_1$  (by moving the weight of internal edges into external edges). Similarly, the algorithm transforms  $T_2$  into a superstar  $T'_2$ . The transformations are chosen to make  $T'_1$  coincide with  $T'_2$ . To transform  $T_1$  to  $T_2$ , we first transform  $T_1$  to  $T'_1 (= T'_2)$  and then transform this to  $T_2$ . Let  $T'_1$  (resp.  $T'_2$ ) denote the tree during the transformation of  $T_1$  (resp.  $T_2$ ).  $T'_1$  (resp.  $T'_2$ ) is initialized to be  $T_1$  (resp.  $T_2$ ).

#### Algorithm DST

Step 0. Initialize  $T'_1 = T_1$  and  $T'_2 = T_2$ .

Step 1. While  $T'_1$  is not a superstar yet and there is an external edge  $e_{T'_1}(a) = (a, u)$  in  $T'_1$  such that  $w(e_{T'_1}(a)) < w(e_{T'_2}(a))$ , do:

- Let  $e_1$  be any positive weight internal edge of  $T'_1$  incident on the supernode containing  $u$ . Let  $d = \min\{w(e_1), [w(e_{T'_2}(a)) - w(e_{T'_1}(a))]\}$ .
- Perform the operation  $\text{move}(e_1, d, e_{T'_1}(a))$  in  $T'_1$ . (Note: after this move operation, either the entire length of  $e_1$  is moved into  $e_{T'_1}(a)$  or  $w(e_{T'_1}(a)) = w(e_{T'_2}(a))$ .)

(Note: after the loop terminates, either  $T'_1$  is a superstar or  $w(e_{T'_1}(a)) \geq w(e_{T'_2}(a))$  for all leaf nodes  $a$ . Also we perform subtree-transfer only on internal edges of  $T_1$ .)

Step 2. Similar to Step 1, with the roles of  $T'_1$  and  $T'_2$  swapped.

Step 3. We transform  $T'_1$  and  $T'_2$  into two superstars such that  $w(e_{T'_1}(a)) = w(e_{T'_2}(a))$  for all leaf nodes  $a$ . There are two possible cases as follows.

Case 3.1.  $w(e_{T'_1}(a)) = w(e_{T'_2}(a))$  for all leaf nodes  $a$ . Perform the following loop to transform both  $T'_1$  and  $T'_2$  into superstars. During the execution of the loop, we maintain the condition  $w(e_{T'_1}(a)) = w(e_{T'_2}(a))$  for all leaf nodes  $a$  (this condition implies that  $T'_1$  is a superstar iff  $T'_2$  is a superstar).

Repeat

Pick any edge  $e_{T'_1}(a) = (a, u_1)$  in  $T'_1$ . Suppose that the corresponding edge  $e_{T'_2}(a)$  in  $T'_2$  is  $(a, u_2)$ . Let  $e_1$  be any positive weight internal edge of  $T'_1$  incident on the super-node containing  $u_1$ . Let  $e_2$  be any positive weight internal edge of  $T'_2$  incident on the supernode containing  $u_2$ . Let  $d = \min\{w(e_1), w(e_2)\}$ . In  $T'_1$  perform the operation  $\text{move}(e_1, d, e_{T'_1}(a))$ . In  $T'_2$  perform the operation  $\text{move}(e_2, d, e_{T'_2}(a))$ . (After this, we have moved the entire length of either  $e_1$  or  $e_2$  into external edges.)

Until both  $T'_1$  and  $T'_2$  are superstars.

(Note: during this step we perform subtree-transfer only on internal edges of  $T_1$  and  $T_2$ .)

Case 3.2. There exists a leaf node  $a$  such that  $w(e_{T_1}(a)) \neq w(e_{T_2}(a))$ . This can happen only if both  $T_1'$  and  $T_2'$  are superstars already. We need to make  $w(e_{T_1}(a)) = w(e_{T_2}(a))$  for all leaf nodes  $a$ . This is done as follows. Partition  $L(T_1')$  into three subsets  $A$ ,  $B$ , and  $C$  as follows:  $A$  (resp.  $B$ ,  $C$ ) is the set of leaf nodes  $a$  (resp.  $b$ ,  $c$ ) such that  $w(e_{T_1}(a)) = w(e_{T_2}(a))$  (resp.  $w(e_{T_1}(b)) < w(e_{T_2}(b))$ ,  $w(e_{T_1}(c)) > w(e_{T_2}(c))$ ).

Repeat

Pick any edge  $e_{T_1}(b)$  with  $b \in B$  and  $e_{T_1}(c)$  with  $c \in C$ . Let  $d = \min\{[w(e_{T_1}(c)) - w(e_{T_2}(c))], [w(e_{T_2}(b)) - w(e_{T_1}(b))]\}$ . In  $T_1'$ , perform  $move(e_{T_1}(c), d, e_{T_1}(b))$ . Then:

- If  $d = w(e_{T_2}(b)) - w(e_{T_1}(b))$ , remove  $b$  from  $B$  and put  $b$  into  $A$ .
- If  $d = w(e_{T_1}(c)) - w(e_{T_2}(c))$ , remove  $c$  from  $C$  and put  $c$  into  $A$ .
- If  $d = w(e_{T_1}(c)) - w(e_{T_2}(c)) = w(e_{T_2}(b)) - w(e_{T_1}(b))$ , remove  $b$  from  $B$ ; remove  $c$  from  $C$ ; put both  $b$  and  $c$  into  $A$ .

Until  $B = C = \emptyset$ .

Step 4. Now both  $T_1'$  and  $T_2'$  are superstars and  $w(e_{T_1}(a)) = w(e_{T_2}(a))$  for all leaf nodes  $a$ . We adjust the topology of the supernodes of  $T_1'$  and  $T_2'$  so that  $T_1'$  and  $T_2'$  are identical.

LEMMA 11. Assume that  $T_1$  and  $T_2$  do not share any good edge pairs. Then Algorithm DST approximates  $D_{st}(T_1, T_2)$  to within a factor of 2 in  $O(n^2)$  time.

PROOF. We analyze the cost and running time of each step of the algorithm. We use the adjacency-list representation of a tree. Steps 0 and 4 incur no costs and can be easily implemented in  $O(n)$  time. During Steps 1, 2, and 3.1, we only transfer subtrees across internal edges of  $T_1$  and  $T_2$ . Over any portion of such an edge  $e$ , at most one subtree-transfer operation occurs. So the total cost of these steps is bounded above by  $W_{int}(T_1) + W_{int}(T_2)$ . Moreover, it is easy to see that at most  $O(n)$  moves are performed during Steps 1, 2, and 3.1, and since each move operation can be implemented in  $O(n)$  time, the total time for all these steps is at most  $O(n^2)$ .

Next, consider Step 3.2. Before the repeat loop is entered, for any  $c \in C$ , we have:

- $w(e_{T_1}(c)) = w(e_{T_2}(c))$ . (This is because no additional weight is moved to the edge  $e_{T_1}(c)$  during Steps 1 and 2.)
- $w(e_{T_2}(c)) \geq w(e_{T_1}(c))$ .

During Step 3.2 we only transfer subtrees across the edges  $e_{T_1}(c)$  for  $c \in C$ . Fix such an edge. Note that any portion of  $e_{T_1}(c)$  is traversed at most once during Step 3.2. Once the length of  $e_{T_1}(c)$  is reduced to  $w(e_{T_2}(c))$ ,  $c$  is removed from  $C$ . So the portion of  $e_{T_1}(c)$  traversed during Step 3.2 is  $w(e_{T_1}(c)) - w(e_{T_2}(c)) = w(e_{T_1}(c)) - w(e_{T_2}(c)) \leq w(e_{T_1}(c)) - w(e_{T_2}(c))$ . So the total cost of Step 3.2 is at most  $\sum_{c \in C} [w(e_{T_1}(c)) - w(e_{T_2}(c))]$ .



$\sum_{c \in C} [w(e_{T_1}(c)) - w(e_{T_2}(c))] \leq W_{\text{ext}, T_1 > T_2}(T_1)$ . Also, we perform at most  $O(n)$  move operations during Step 3.2, and hence this step can also be implemented in  $O(n^2)$  time.

Thus the total cost of the algorithm is bounded above by  $W_{\text{int}}(T_1) + W_{\text{int}}(T_2) + W_{\text{ext}, T_1 > T_2}(T_1)$ , which is at most  $2D_{\text{st}}(T_1, T_2)$  by Lemma 6.  $\square$

Next we consider the general case when  $T_1$  and  $T_2$  may share some good edge pairs. First we show how to find all good edge pairs efficiently.

**LEMMA 12.** *Let  $T_1$  and  $T_2$  be two trees, each with  $n$  leaves. Then the set of good edges of  $T_1$  (with respect to  $T_2$ ) can be enumerated in  $O(n^2)$  time.*

**PROOF.** First we calculate, for every edge  $e$  of either  $T_1$  or  $T_2$ ,  $w(E(T_{11}))$  and  $w(E(T_{12}))$  where  $T_{11}$  and  $T_{12}$  are two subtrees at the two endpoints of  $e$ . This can be trivially done in  $O(n^2)$  time. Next we ignore condition 2 of the definition of good edge pairs (Definition 2), and find all those edge pairs of  $T_1$  and  $T_2$  which satisfy only condition 1 of this definition. This can be done in  $O(n)$  time by Lemma 12. Finally, for every such edge pair which satisfies condition 1 of this definition, we check if it satisfies condition 2 of the definition also. This takes  $O(n^2)$  time.  $\square$

We now show how to apply Algorithm DST to achieve an approximation ratio of 2 when  $T_1$  and  $T_2$  may share some good edge pairs. Let  $K$  be the number of good edge pairs in  $T_1$  and  $T_2$ . Our algorithm is by induction on  $K$ . If  $K = 0$ , Algorithm DST works by Lemma 11. Suppose  $K > 0$ . Let  $e_1 = (u_1, v_1) \in E(T_1)$  and  $e_2 = (u_2, v_2) \in E(T_2)$  be a good edge pair. Let  $T'_1$  and  $T''_1$  be the two subtrees of  $T_1$  partitioned by  $e_1$ . Let  $T'_2$  and  $T''_2$  be the two subtrees of  $T_2$  partitioned by  $e_2$ , where  $L(T'_1) = L(T'_2)$  and  $L(T''_1) = L(T''_2)$ .

Assume  $w(E(T'_1)) \leq w(E(T'_2)) < w(E(T'_1)) + w(e_1)$ . (The other case can be handled in a similar way.) Add a new edge  $(u_1, x)$  to  $T'_1$  and assign  $w((u_1, x)) = w(E(T'_2)) - w(E(T'_1))$ . Add a new edge  $(x, v_1)$  to  $T''_1$  and assign  $w((x, v_1)) = w(e_1) - w((u_1, x))$ . Add a new edge  $(u_2, x)$  to  $T'_2$  and assign  $w((u_2, x)) = 0$ . Add a new edge  $(x, v_2)$  to  $T''_2$  and assign  $w((x, v_2)) = w(e_2)$ . (See Figure 11). Note that the weights of all new edges are nonnegative.

Now we have  $L(T'_1) = L(T'_2)$  and  $w(T'_1) = w(T'_2)$ . We can normalize the weights of  $T'_1$  and  $T'_2$  such that their sum is one. By induction hypothesis, we can transform  $T'_1$  to  $T'_2$  with cost at most  $2D_{\text{st}}(T'_1, T'_2)$ . Similarly, we can transform  $T''_1$  to  $T''_2$  with cost at most

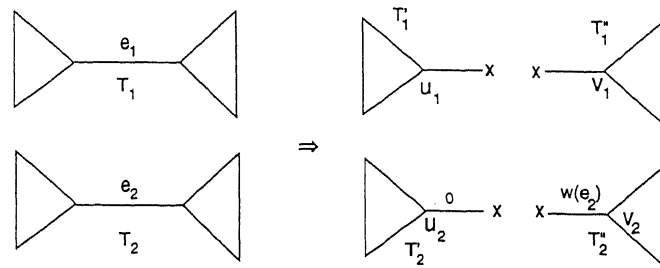


Fig. 11. Cut each of  $T_1$  and  $T_2$  into two smaller trees.

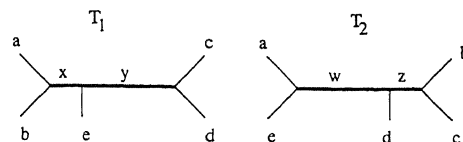


Fig. 12. A counterexample to the entangle conjecture.

$2D_{\text{st}}(T_1'', T_2'')$ . Combining the two transfer sequences, we can transform  $T_1$  to  $T_2$  with cost at most  $2D_{\text{st}}(T_1, T_2)$ . The complete algorithm takes  $O(n^2)$  time. This completes the proof of Theorem 10.

REMARK. Naturally, one may try to investigate if the performance ratio of 2 in Theorem 10 can be further improved. For this purpose, note that in some cases, the lower bounds of Lemma 6 are rather weak, for instance if two trees have four leaves each, differently partitioned over the single internal edge of weight one, with all four external edges having zero weight. The transformation cost in this case is two, whereas the above only shows a lower bound of one. The internal edges of these two trees could be said to be *entangled*, since they partition the leaves in sets none of which is contained in another. So one must bring the various leaves together first, and after repartitioning, move them apart again. This led us to the following conjecture: *Disjoint pairs of entangled edges contribute at least their sum of weights to the optimal cost*. However, the trees in Figure 12 provide a counterexample (external edges have zero weight), in that the edge pairs  $\{x, w\}$  and  $\{y, z\}$  are both entangling, yet the distance between  $T_1$  and  $T_2$  is less than the sum weight of these four edges.

**4. Discussion and Open Problems.** These results have been obtained as a part of our larger project of building a comprehensive software package for comparing phylogenetic trees.

One may wonder why we could obtain a factor 2 approximation for the linear-cost subtree-transfer distance on weighted phylogenies, where we could get only a  $\log n$  factor approximation for unweighted phylogenies. However, notice that all intermediate trees in the unweighted case are also binary trees, whereas in the weighted case, intermediate trees of high degree may be produced (e.g., by allowing zero length edges). In other words, in the weighted case, the topology of an intermediate tree may be considerably different from the given trees, and in fact, we do utilize this to get a factor 2 approximation. Consequently, the distance may vary considerably depending on whether we are considering unweighted or weighted phylogenies. For example, consider unweighted trees with  $n$  labeled leaves, and weighted trees with  $n$  labeled leaves where the weight of every internal edge is  $1/(n-3)$  and the rest of the edges have zero weights. Assume also the two (unweighted or weighted) trees involved in the distance calculation share no good edge pairs (Definition 1 or 2, as appropriate). In the unweighted case, it is known that there are two trees which are at a distance of  $\Omega(n \log n)$  [23]. However, in the weighted case, our factor 2 approximation algorithm and the lower bounds in Lemma 6 imply that any two trees are at a distance of at most  $O(1)$ .

Several open questions still remain and may be worth pursuing further:

1. Is the linear-cost subtree-transfer problem NP-hard when the trees are (uniquely) labeled **and** weighted?
2. Can we approximate the linear-cost subtree-transfer distance for weighted phylogenies with a ratio better than 2?

**Acknowledgments.** We thank J. Felsenstein and J. Hein for explaining to us the biological motivations for comparing weighted phylogenies and studying the linear-cost subtree-transfer distance, respectively, Takashi Yokomori, Shengke Yu, and Louxin Zhang for discussions on related topics, and Shengke Yu for implementing the user interface. We would also like to thank all the anonymous reviewers for their extremely helpful comments and pointing out reference [5] to us.

### References

- [1] D. Barry and J. A. Hartigan, Statistical analysis of hominoid molecular evolution, *Statist. Sci.*, 2 (1987), 191–210.
- [2] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, Cambridge, MA, 1990.
- [3] B. DasGupta, X. He, T. Jiang, M. Li, J. Tromp, and L. Zhang, On distances between phylogenetic trees, *Proc. 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1997, pp. 427–436.
- [4] B. DasGupta, X. He, T. Jiang, M. Li, J. Tromp, and L. Zhang, On computing the nearest neighbor interchange distance, Preprint, 1997.
- [5] W. H. E. Day, Optimal algorithms for comparing trees with labeled leaves, *J. Classification*, 2 (1985), 7–28.
- [6] A. W. F. Edwards and L. L. Cavalli-Sforza, The reconstruction of evolution, *Ann. Human Genetics*, 27 (1964), 105. (Also in *Heredity*, 18, 553.)
- [7] J. Felsenstein, Evolutionary trees for DNA sequences: a maximum likelihood approach. *J. Mol. Evol.*, 17 (1981), 368–376.
- [8] J. Felsenstein, Personal communication, 1996.
- [9] W. M. Fitch, Toward defining the course of evolution: minimum change for a specified tree topology, *Syst. Zool.*, 20 (1971), 406–416.
- [10] W. M. Fitch and E. Margoliash, Construction of phylogenetic trees, *Science*, 155 (1967), 279–284.
- [11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA, 1979.
- [12] J. Hein, Reconstructing evolution of sequences subject to recombination using parsimony, *Math. Biosci.*, 98 (1990), 185–200.
- [13] J. Hein, A heuristic method to reconstruct the history of sequences subject to recombination, *J. Mol. Evol.*, 36 (1993), 396–405.
- [14] J. Hein, Personal email communication, 1996.
- [15] J. Hein, T. Jiang, L. Wang, and K. Zhang, On the complexity of comparing evolutionary trees, *Discrete Appl. Math.*, 71 (1996), 153–169.
- [16] M. Kuhner and J. Felsenstein, A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Mol. Biol. Evol.* 11(3) (1994), 459–468.
- [17] W. J. Le Quesne, The uniquely evolved character concept and its cladistic application, *Syst. Zool.*, 23 (1974), 513–517.
- [18] M. Li, J. Tromp, and L. Zhang, Some notes on the nearest neighbor interchange distance, *J. Theoret. Biol.*, 182 (1996), 463–467.

- [19] G. W. Moore, M. Goodman, and J. Barnabas, An iterative approach from the standpoint of the additive hypothesis to the dendrogram problem posed by molecular data sets, *J. Theoret. Biol.*, 38 (1973), 423-457.
- [20] D. F. Robinson, Comparison of labeled trees with valency three, *J. Combin. Theory Ser. B*, 11 (1971), 105-119.
- [21] N. Saitou and M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, *Mol. Biol. Evol.*, 4 (1987), 406-425.
- [22] D. Sankoff, Minimal mutation trees of sequences, *SIAM J. Appl. Math.*, 28 (1975), 35-42.
- [23] D. Sleator, R. Tarjan, and W. Thurston, Short encodings of evolving structures, *SIAM J. Discrete Math.*, 5 (1992), 428-450.
- [24] A. von Haseler and G. A. Churchill, Network models for sequence evolution, *J. Mol. Evol.*, 37 (1993), 77-85.
- [25] M. S. Waterman and T. F. Smith, On the similarity of dendrograms, *J. Theoret. Biol.*, 73 (1978), 789-800.