A New Editing based Distance between Unordered Labeled Trees *

Kaizhong Zhang

Department of Computer Science
University of Western Ontario
London, Ont. N6A 5B7
CANADA
e-mail: kzhang@csd.uwo.ca

Abstract. This paper considers the problem of computing a new editing based distance between unordered labeled trees. The problem of approximate unordered tree matching is also considered. We present algorithms solving these problems in sequential time $O(|T_1| \times |T_2| \times \max\{deg(T_1), deg(T_2)\} \times \log_2(max\{deg(T_1), deg(T_2)\}))$. Our previous result shows that computing the editing distance between unordered labeled trees is NP-complete.

1 Introduction

Unordered labeled trees are trees whose nodes are labeled and in which only ancestor relationships are significant (the left-to-right order among siblings is not significant). Such trees arise naturally in genealogical studies, for example, the genetic study of the tracking of diseases. For many such applications, it would be useful to compare unordered labeled trees by some meaningful distance metric. The editing distance metric, used with some success for ordered labeled trees [4], is a natural such metric.

In section 2 we review the previous NP-completeness results on editing distance between unordered trees. In section 3 we introduce the new distance metric between trees, which we call constrained editing distance. In section 4 we will investigate the properties of the new distance metric. In section 5 we present an algorithm to compute the new distance metric and analyse the complexity. In section 6 we consider the approximate unordered tree matching problem.

2 Preliminaries

In this section we will first introduce some basic definitions and then review the previous results on unordered trees. Unless otherwise stated, all trees we consider in the paper are rooted, labeled, and unordered.

* Research supported by the Natural Sciences and Engineering Research Council of Canada under Grant No. OGP0046373.

2.1 Editing Operations

We consider three kinds of operations. Changing a node n means changing the label on n. Deleting a node n means making the children of n become the children of the parent of n and then removing n. Inserting is the complement of deleting. This means that inserting n as the child of m will make n the parent of a *subset* (as opposed to a consecutive subsequence [11]) of the current children of m.

Following [5, 11, 12], we represent an edit operation as $a \to b$, where a is either λ or a label of a node in tree T_1 and b is either λ or a label of a node in tree T_2. We call $a \to b$ a change operation if $a \neq \lambda$ and $b \neq \lambda$; a delete operation if $b = \lambda$; and an insert operation if $a = \lambda$.

Let S be a sequence $s_1, ..., s_k$ of edit operations. An S-derivation from tree A to tree B is a sequence of trees $A_0, ..., A_k$ such that $A = A_0$, $B = A_k$, and $A_{i-1} \to A_i$ via s_i for $1 \leq i \leq k$. Let γ be a cost function which assigns to each edit operation $a \to b$ a nonnegative real number $\gamma(a \to b)$.

We constrain γ to be a distance metric. That is, i) $\gamma(a \to b) \geq 0$, $\gamma(a \to a) = 0$; ii) $\gamma(a \to b) = \gamma(b \to a)$; and iii) $\gamma(a \to c) \leq \gamma(a \to b) + \gamma(b \to c)$.

We extend γ to the sequence of editing operations S by letting $\gamma(S) = \sum_{i=1}^{|S|} \gamma(s_i)$.

2.2 Editing Distance and Editing Distance Mapping

The results in this subsection are from [12]. We will omit the proofs.

Editing Distance. [12] defined the *editing distance* between two trees by considering the minimum cost editing operations sequence that transforms one tree to the other. Formally the distance between T_1 and T_2 is defined as:

$D_e(T_1, T_2) = \min_S \{\gamma(S) \mid S \text{ is an edit operation sequence taking } T_1 \text{ to } T_2\}$.

Editing Distance Mappings. The edit operations give rise to a mapping which is a graphical specification of what edit operations apply to each node in the two unordered labeled trees.

Suppose that we have a numbering for each tree. Let $t[i]$ be the ith node of tree T in the given numbering. Formally we define a triple (M_e, T_1, T_2) to be an editing distance mapping from T_1 to T_2, where M_e is any set of pair of integers (i, j) satisfying:

(1) $1 \leq i \leq |\ T_1\ |$, $1 \leq j \leq |\ T_2\ |$;
(2) For any pair of (i_1, j_1) and (i_2, j_2) in M_e,
   (a) $i_1 = i_2$ iff $j_1 = j_2$ (one-to-one)
   (b) $t_1[i_1]$ is an ancestor of $t_1[i_2]$ iff $t_2[j_1]$ is an ancestor of $t_2[j_2]$ (ancestor order preserved)

We will use M_e instead of (M_e, T_1, T_2) if there is no confusion. Let M_e be an editing distance mapping from T_1 to T_2. Then we can define the cost of M_e:

$$\gamma(M_e) = \sum_{(i,j) \in M_e} \gamma(t_1[i] \to t_2[j]) + \sum_{i \notin M_e} \gamma(t_1[i] \to \lambda) + \sum_{j \notin M_e} \gamma(\lambda \to t_2[j])$$

The relation between an editing distance mapping and a sequence of editing operations is as follows:

Lemma 1. *Given S, a sequence $s_1, \ldots, s_k$ of edit operations from T_1 to T_2, there exists a editing distance mapping M_e from T_1 to T_2 such that $\gamma(M_e) \leq \gamma(S)$. Conversely, for any mapping M_e, there exists a sequence of editing operations such that $\gamma(S) = \gamma(M_e)$.*

Based on the lemma, the following theorem states the relation between the editing distance and the editing distance mappings. This is why we call this kind mapping editing distance mapping.

Theorem 2. $D_e(T_1, T_2) = \min_{M_e}\{\gamma(M_e) \mid M_e$ *is a mapping from* T_1 *to* $T_2\}$

One of the results in [12] is that finding $D_e(T_1, T_2)$ is NP-complete even if the trees are binary trees with a label alphabet of size two. Kilpelainen and Mannila [2] showed that even the inclusion problem for unordered trees is NP-complete. In fact we recently proved a stronger result that the problem of finding the minimum cost mapping (edit distance) between two unordered trees and the problem of finding the largest common subtree of two unordered trees are both MAX SNP-hard which means that there is no polynomial time approximation scheme (PTAS) for these problems unless P=NP. Since unordered trees are important in some applications, one would like to find distances that can be efficiently computed.

3 A New Editing based Distance Metric between Unordered Trees

Our new distance metric is based on a restriction of the mappings allowed between two trees. The intuitive idea is that two separate subtrees of T_1 should be mapped to two separate subtrees in T_2. This idea was proposed by Tanaka and Tanaka [7] in their definition for a structure preserving mapping between two ordered labeled trees although the definition in [7] dose not capture the idea precisely. Tanaka and Tanaka [7] also showed that in some applications (e.g., classification tree comparison) the structural preserving mapping is more meaningful than editing distance mapping. We refined the definition for ordered trees [10] and in this paper extend the definition from ordered trees to unordered trees

3.1 New Mappings

Suppose again that we have a numbering for each tree. Let $t[i]$ be the ith node of tree T in the given numbering. Let $T[i]$ be the subtree rooted at $t[i]$ and $F[i]$ be the unordered forest obtained by deleting $t[i]$ from $T[i]$.

Formally we define a triple (M, T_1, T_2) to be a mapping from T_1 to T_2, where M is any set of pairs of integers (i, j) satisfying:

(1) $1 \leq i \leq |\, T_1 \,|$, $1 \leq j \leq |\, T_2 \,|$;
(2) For any pair of (i_1, j_1) and (i_2, j_2) in M_e,
   (a) $i_1 = i_2$ iff $j_1 = j_2$ (one-to-one)
   (b) $t_1[i_1]$ is an ancestor of $t_1[i_2]$ iff $t_2[j_1]$ is an ancestor of $t_2[j_2]$ (ancestor order preserved);

(3) For any triple (i_1, j_1), (i_2, j_2) and (i_3, j_3) in M, let $t_1[I]$ be the $lca(t_1[i_1], t_1[i_2])$ and $t_2[J]$ be the $lca(t_2[j_1], t_2[j_2])$, where lca represents least common ancestor. $t_1[I]$ is not an ancestor or a descendant of $t_1[i_3]$ iff $t_2[J]$ is not an ancestor or a descendant of $t_2[j_3]$.

We will use M instead of (M, T_1, T_2) if there is no confusion. Let M be a mapping from T_1 to T_2. Then we can similarly define the cost of M:

$$\gamma(M) = \sum_{(i,j)\in M} \gamma(t_1[i] \rightarrow t_2[j]) + \sum_{i \notin M} \gamma(t_1[i] \rightarrow \lambda) + \sum_{j \notin M} \gamma(\lambda \rightarrow t_2[j])$$

Mappings can be composed. Let M_1 be a mapping from T_1 to T_2 and M_2 be a mapping from T_2 to T_3. Define

$$M_1 \circ M_2 = \{(i,j) \mid \exists k \; s.t. \; (i,k) \in M_1 \; and \; (k,j) \in M_2\}.$$

Lemma 3. *1)* $M_1 \circ M_2$ *is a mapping between* T_1 *and* T_3. *2)* $\gamma(M_1 \circ M_2) \leq \gamma(M_1) + \gamma(M_2)$.

Proof: (1) follows from the definition of mapping. Let us check condition (3) only. Let (i_1, j_1), (i_2, j_2) and (i_3, j_3) be in $M_1 \circ M_2$. By the definition of $M_1 \circ M_2$, there are k_1, k_2 and k_3 such that (i_1, k_1), (i_2, k_2) and (i_3, k_3) are in M_1 and (k_1, j_1), (k_2, j_2) and (k_3, j_3) are in M_2. Let I be the $lca(i_1, i_2)$, K be the $lca(k_1, k_2)$ and J be the $lca(j_1, j_2)$. By the definition of M_1 and M_2, I is not an ancestor or descendant of i_3 iff K is not an ancestor or descendant of k_3. Moreover K is not an ancestor or descendant of k_3 iff J is not an ancestor or descendant of j_3. Therefore I is not an ancestor or descendant of i_3 iff J is not an ancestor or descendant of j_3.

(2) Let M_1 be the mapping from T_1 to T_2. Let M_2 be the mapping from T_2 to T_3. Let $M_1 \circ M_2$ be the composed mapping from T_1 to T_3 and let I and J be the corresponding deletion and insertion sets. Three general situations occur. $(i,j) \in M_1 \circ M_2$, $i \notin M_1$, or $j \notin M_2$. In each case this corresponds to an editing operation $\gamma(x \rightarrow y)$ where x and y may be nodes or may be λ. In all such cases, the triangle inequality on the distance metric γ ensures that $\gamma(x \rightarrow y) \leq \gamma(x \rightarrow z) + \gamma(z \rightarrow y)$. □

3.2 A New Editing Based Distance between Trees

We can now define a dissimilarity measure between T_1 and T_2 as:

$$D(T_1, T_2) = \min_M \{\gamma(M) \mid M \; is \; a \; mapping \; from \; T_1 \; to \; T_2\}$$

In fact this dissimilarity measure is a distance metric.

Theorem 4. *1)* $D(T, T) = 0$;
2) $D(T_1, T_2) = D(T_2, T_1)$;
3) $D(T_1, T_3) \leq D(T_1, T_2) + D(T_2, T_3)$.

Proof: 1) and 2) follow directly from the definition of the mapping. For 3), consider the minimum cost mappings M_1 between T_1 and T_2 and M_2 between T_2 and T_3. It is easy to see the following:

$$D(T_1, T_3) \leq \gamma(M_1 \circ M_2) \leq \gamma(M_1) + \gamma(M_2) = D(T_1, T_2) + D(T_2, T_3).$$

□

The relation between our new distance metric D and the editing distance metric D_e is: $D_e(T_1, T_2) \leq D(T_1, T_2)$. The reason is that any new mapping we defined is always a editing distance mapping.

4 Properties of the New Distance

In this section we will present several lemmas which will be the basis for the algorithm in the next section.

Lemma 5. *Let $t_1[i_1], t_1[i_2], ...t_1[i_{n_i}]$ be the children of $t_1[i]$ and $t_2[j_1], t_2[j_2], ...t_2[j_{n_j}]$ be the children of $t_2[j]$, then $D(\theta, \theta) = 0$;*
$D(F_1[i], \theta) = \sum_{k=1}^{n_i} D(T_1[i_k], \theta)$; $D(T_1[i], \theta) = D(F_1[i], \theta) + \gamma(t_1[i] \rightarrow \lambda)$;
$D(\theta, F_2[j]) = \sum_{k=1}^{n_j} D(\theta, T_2[j_k])$; $D(\theta, T_2[j]) = D(\theta, F_2[j]) + \gamma(\lambda \rightarrow t_2[j])$.

Lemma 6. *Let $t_1[i_1], t_1[i_2], ...t_1[i_{n_i}]$ be the children of $t_1[i]$ and $t_2[j_1], t_2[j_2], ...t_2[j_{n_j}]$ be the children of $t_2[j]$, then*

$$D(T_1[i], T_2[j]) = \min \begin{cases} D(T_1[i], \theta) + \min_{1 \leq s \leq n_i} \{D(T_1[i_s], T_2[j]) - D(T_1[i_s], \theta)\} \\ D(\theta, T_2[j]) + \min_{1 \leq t \leq n_j} \{D(T_1[i], T_2[j_t]) - D(\theta, T_2[j_t])\} \\ D(F_1[i], F_2[j]) + \gamma(t_1[i] \rightarrow t_2[j]) \end{cases}$$

Proof: Consider the minimum-cost mapping M between $T_1[i]$ and $T_2[j]$. There are four cases: (1) $i \in M$ and $j \notin M$, (2) $i \notin M$ and $j \in M$, (3) $i \in M$ and $j \in M$, (4) $i \notin M$ and $j \notin M$.

Case 1: let (i, t) in M. Since $j \notin M$, t must be a node in $F_2[j]$. Let $t_2[j_t]$ be the child of $t[j]$ on the path from $t_2[t]$ to $t_2[j]$. Thus $D(T_1[i], T_2[j]) = D(T_1[i], T_2[j_t])$ $+D(\theta, T_2[j_1]) + ... + D(\theta, T_2[j_{t-1}]) + D(\theta, T_2[j_{t+1}]) + ... + D(\theta, T_2[j_{n_j}]) + \gamma(\lambda, t_2[j])$. Since $D(\theta, T_2[j]) = \gamma(\lambda, t_2[j]) + \sum_{k=1}^{n_j} D(\theta, T_2[j_k])$, we can rewrite the right hand side of the formula as $D(\theta, T_2[j]) + D(T_1[i], T_2[j_t]) - D(\theta, T_2[j_t])$. Since the range of k is from 1 to n_j, we take the minimum of these corresponding costs.

Case 2 is similar to case 1.

Case 3: since $i \in M$ and $j \in M$, by the condition of mapping, (i, j) must be in M. Since $M - (i, j)$ is a mapping between $F_1[i]$ and $F_2[j]$, and for any mapping M' between $F_1[i]$ and $F_2[j]$, $(i, j) \cup M'$ is a mapping between $T_1[i]$ and $T_2[j]$, we know $D(T_1[i], T_2[j]) = D(F_1[i], F_2[j]) + \gamma(t_1[i], t_2[j])$.

Case 4 is similar to Case 3. The formula would be $D(T_1[i], T_2[j]) = D(F_1[i], F_2[j])$ $+\gamma(t_1[i], \lambda) + \gamma(\lambda, t_2[j])$. Since $\gamma(t_1[i], t_2[j]) \leq \gamma(t_1[i], \lambda) + \gamma(\lambda, t_2[j])$, we do not have to include this case in our final formula. □

Before we proceed to the next lemma we need the following definition.

Define a restricted mapping $RM(i,j)$ between $F_1[i]$ and $F_2[j]$ as follows:
1) $RM(i,j)$ is a mapping between $F_1[i]$ and $F_2[j]$,
2) if (l,k) is in $RM(i,j)$ and $t_1[l]$ is in $T_1[i_s]$ and $t_2[k]$ is in $T_2[j_t]$, then for any (l_1,k_1) in $RM(i,j)$ $t_1[l_1]$ is in $T_1[i_s]$ if and only if $t_2[k_1]$ is in $T_2[j_t]$.

Since a restricted mapping is a mapping, the cost of a restricted mapping is well defined.

Lemma 7. *Let $t_1[i_1], t_1[i_2], ...t_1[i_{n_i}]$ be the children of $t_1[i]$ and $t_2[j_1], t_2[j_2], ...t_2[j_{n_j}]$ be the children of $t_2[j]$, then*

$$D(F_1[i],F_2[j]) = \min \begin{cases} D(F_1[i],\theta) + \min_{1\le s\le n_i}\{D(F_1[i_s],F_2[j]) - D(F_1[i_s],\theta)\} \\ D(\theta,F_2[j]) + \min_{1\le t\le n_j}\{D(F_1[i],F_2[j_t]) - D(\theta,F_2[j_t])\} \\ \min_{RM(i,j)}\gamma(RM(i,j)) \end{cases}$$

Proof: We consider the minimum-cost mapping M between $F_1[i]$ and $F_2[j]$. There are four cases.

Case 1: there is a $1 \le s \le n_i$ such that if $(k,l) \in M$, then $t_1[k]$ is a node in subtree $T_1[i_s]$; and there are (k_1,l_1) and (k_2,l_2) in M such that $t_2[l_1]$ is a node in $T_2[i_{t_1}]$ and $t_2[l_2]$ is a node in $T_2[i_{t_2}]$, where $1 \le t_1 \ne t_2 \le n_j$. Note that in this case i_s cannot be in M. This is similar to case 1 in Lemma 6, and hence we have following formula: $D(F_1[i],F_2[j]) = D(F_1[i],\theta) + \min_{1\le s\le n_i} D(F_1[i_s],F_2[j]) - D(F_1[i_s],\theta)$.

Case 2: there is a $1 \le t \le n_j$ such that if $(k,l) \in M$, then $t_2[l]$ is a node in subtree $T_2[i_t]$; and there are (k_1,l_1) and (k_2,l_2) in M such that $t_1[k_1]$ is a node in $T_2[i_{s_1}]$ and $t_1[k_2]$ is a node in $T_2[i_{s_2}]$, where $1 \le s_1 \ne s_2 \le n_i$. This is similar to case 1.

Case 3: there are s and t such that if $(k,l) \in M$ then $t_1[k]$ is node in $T_1[i_s]$ and $t_2[l]$ is node in $T_2[i_t]$. In this case M is a restricted mapping and therefore $D(F_1[i],F_2[j]) = \min_{RM(i,j)}\gamma(RM(i,j))$.

Case 4: there are $(k_1,l_1),(k_2,l_2),(x_1,y_1),(x_2,y_2)$ in M such that $t_1[k_1]$ is a node in $T_1[i_{s_1}]$, $t_1[k_2]$ is a node in $T_1[i_{s_2}]$, $t_2[y_1]$ is a node in $T_2[i_{t_1}]$, and $t_2[y_2]$ is a node in $T_2[i_{t_2}]$, where $1 \le s_1 \ne s_2 \le n_i$ and $1 \le t_1 \ne t_2 \le n_j$. We will show that in this case the mapping M is a restricted mapping between $F_1[i]$ and $F_2[j]$.

Suppose this is not true. Then, w.l.o.g., we assume that there are (a_1,b_1) and (a_2,b_2) in M such that $t_1[a_1]$ and $t_1[a_2]$ belong to the same subtree and $t_2[b_1]$ and $t_2[b_2]$ belong to different subtrees. Let $(a_3,b_3) \in M$ such that $t_1[a_3]$ and $t_1[a_1]$ belong to different subtrees of $F_1[i]$. Now consider $lca(t_1[a_1],t_1[a_2])$ and $t_1[a_3]$. Since $t_1[a_1]$ and $t_1[a_2]$ belong to the same subtree which is different from the subtree $t_1[a_3]$ belongs to, we know that $lca(t_1[a_1],t_1[a_2])$ and $t_1[a_3]$ are not in ancestor or descendant relationship. However if we consider $lca(t_2[b_1],t_2[b_2])$ and $t_2[b_3]$, it is easy to see that $lca(t_2[b_1],t_2[b_2]) = t_2[j]$ which is an ancestor of $t_2[b_3]$. This means that M is not a valid mapping. Contradiction. □

From lemma 7, in order to compute $D(F_1[i],F_2[j])$, $\min_{RM(i,j)}\gamma(RM(i,j))$ have to be computed first. The next two lemmas will establish the relationship between $\min_{RM(i,j)}\gamma(RM(i,j))$ and $D(T_1[i_s],T_2[j_t])$, where $1 \le s \le n_i$ and $1 \le n_j$. We need the following definition in the next two lemmas.

Given $I = \{i_1,i_2,...i_{n_i}\}$ and $J = \{j_1,j_2,...j_{n_j}\}$, We define a partial function between I and J $PF(i,j)$ as follows:

1) $PF(i,j)$ is a set of pairs (s,t) such that $1 \leq s \leq n_i$ and $1 \leq t \leq n_j$,
2) let (s,t) and (x,y) be in $PF(i,j)$, $s = x$ if and only if $t = y$.
The cost of a partial function, $gamma(FP(i,j))$, is defined as follows:
$\sum_{(s,t)\in PF(i,j)} D(T_1[i_s], T_2[j_t]) + \sum_{s\notin PF(i,j)} D(T_1[i_s], \theta) + \sum_{t\notin PF(i,j)} D(\theta, T_2[j_t])$.

Lemma 8. $\min_{RM(i,j)} \gamma(RM(i,j)) = \min_{PF(i,j)} \gamma(PF(i,j))$

Proof: Given a restricted mapping $RM(i,j)$, we define a partial function $PF(i,j)$ as follows.

$$PF(i,j) = \{(s,t) | \textit{ there is } (k,l) \textit{ in } RM \textit{ s.t. } t_1[k]\ (t_2[l]) \textit{ is a node in } T_1[i_s]\ (T_2[j_t])\}$$

By the definition of restricted mapping this is indeed a partial function. Furthermore it is easy to see that $\gamma(PF(i,j)) \leq \gamma(RM(i,j))$.

On the other hand, given a partial function $PF(i,j)$, we can construct a restricted mapping $RM(i,j)$ as follows.

$$RM(i,j) = \left\{(k,l) \middle| \begin{array}{l} \textit{there exists } (s,t) \in PF(i,j) \textit{ s.t. } (k,l) \textit{ is in} \\ \textit{the minimum cost mapping between } T_1[i_s] \textit{ and } T_2[j_t] \end{array}\right\}$$

It is clear that this is a restricted mapping. Therefore $\gamma(RM(i,j)) \leq \gamma(PF(i,j))$.

Hence $\min_{RM(i,j)} \gamma(RM(i,j)) = \min_{PF(i,j)} \gamma(PF(i,j))$. □

In fact we can add one more condition in the definition of partial function, namely $|PF(i,j)| = \min\{n_i, n_j\}$. The reason is that if $|PF(i,j)| < \min\{n_i, n_j\}$ then there are s and t such that $s \notin PF(i,j)$ and $t \notin PF(i,j)$. Because D is a distance metric, $D(T_1[i_s], T_2[j_t]) \leq D(T_1[i_s], \theta) + D(\theta, T_2[j_t])$. Therefore $\gamma(PF'(i,j) = \{(s,t) \cup PF(i,j)\}) \leq \gamma(PF(i,j))$.

Combining lemma 7, 8 and the above observation, we have proved the following lemma.

Lemma 9. *Let $t_1[i_1], t_1[i_2], \ldots t_1[i_{n_i}]$ be the children of $t_1[i]$ and $t_2[j_1], t_2[j_2], \ldots t_2[j_{n_j}]$ be the children of $t_2[j]$, then*

$$D(F_1[i], F_2[j]) = \min \begin{cases} D(F_1[i], \theta) + \min\limits_{1\leq s\leq n_i} D(F_1[i_s], F_2[j]) - D(F_1[i_s], \theta) \\ D(\theta, F_2[j]) + \min\limits_{1\leq t\leq n_j} D(F_1[i], F_2[j_t]) - D(\theta, F_2[j_t]) \\ \min\limits_{|PF(i,j)|=\min\{n_i,n_j\}} \gamma(PF(i,j)) \end{cases}$$

5 Algorithm and Complexity

We first consider how to compute $\min\limits_{|PF(i,j)|=\min\{n_i,n_j\}} \gamma(PF(i,j))$ and then present our simple algorithm.

5.1 Algorithm

From the definition of $PF(i,j)$ and $\gamma(PF(i,j))$, it is clear that this problem is related to the minimum cost bipartite matching problem. If $n_i = n_j$, then this is exactly the minimum cost bipartite matching problem. If $n_i \neq n_j$, then we have to consider those extra trees in one of the forests. Suppose that $n_i > n_j$. One way to solve this problem is to add $n_i - n_j$ empty trees to $F[j]$ and then use bipartite matching. However this will result in redundant computation. We will reduce this problem directly to the minimum cost maximum flow problem by adding only one empty tree to $F[j]$.

Given $F_1[i]$ and $F_2[j]$, w.l.o.g., we assume that $n_i > n_j$. Let $I = \{i_1, i_2, ...i_{n_i}\}$ and $J = \{j_1, j_2, ...j_{n_j}\}$, where i_k, $1 \leq k \leq n_i$, represents tree $T_1[i_k]$, and j_l, $1 \leq l \leq n_j$, represents tree $T_2[j_l]$.

We construct a graph $G = (V, E)$ as follows:

vertex set: $V = \{s, t, e\} \cup I \cup J$, where s is the source, t is the sink and e represents an empty tree;
edge set: $[s, i_k], [j_l, t], [e, t]$ with cost zero, $[i_k, j_l]$ with cost $D(T_1[i_k], T_2[j_l])$, and $[i_k, e]$ with cost $D(T_1[i_k], \theta)$. All the edges have capacity one except $[e, t]$ whose capacity is $n_i - n_j$.

G is a network with integer capacities, nonnegative costs, and the maximum flow $f^* = n_i = \max\{n_i, n_j\}$, see figure 1.

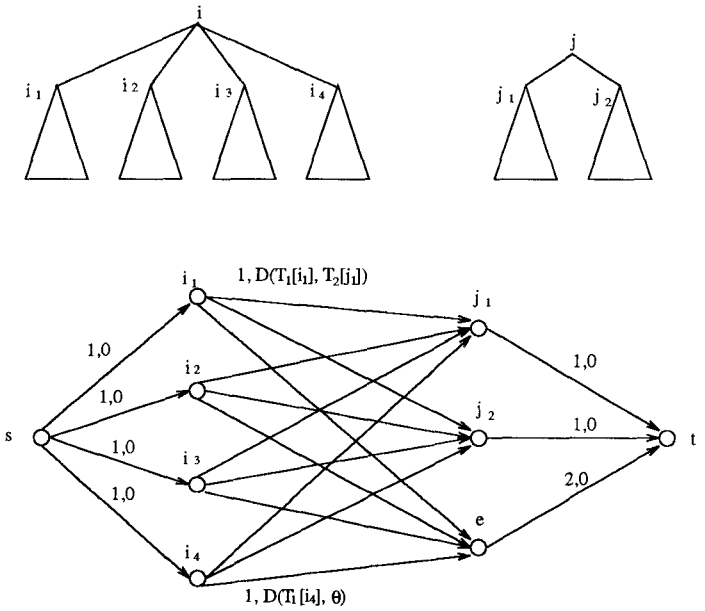

Fig. 1. Reduction to the minimum cost flow problem.

Now let us examine the meaning of $\gamma(PF(i,j))$. It is easy to see that given a $PF(i,j)$, $\gamma(PF(i,j))$ represents the cost of the following maximum flow on G: for

any $(k,l) \in PF(i,j)$ the flow on edge $[i_k, j_l]$ is one; for any $k \notin PF(i,j)$ the flow on edge $[i_k, e]$ is one; the flow on edge $[s, i_k]$ and edge $[j_l, t]$ are one; the flow on edge $[e,t]$ is $n_i - n_j$; and all the flows on the other edges are zero.

Therefore $\min_{|PF(i,j)|=\min\{n_i,n_j\}} \gamma(PF(i,j))$ is exactly the cost of the minimum cost maximum flow of G. Hence we can use minimum cost maximum flow algorithm to compute $\min_{|PF(i,j)|=\min\{n_i,n_j\}} \gamma(PF(i,j))$.

We are now ready to give our algorithm.

Input: T_1 and T_2
Output: $D(T_1[i], T_2[j])$, where $1 \le i \le |T_1|$ and $1 \le j \le |T_2|$

$D(\theta, \theta) = 0$;
for $i = 1$ to $|T_1|$
  $D(F_1[i], \theta) = \sum_{k=1}^{n_i} D(T_1[i_k], \theta)$
  $D(T_1[i], \theta) = D(F_1[i], \theta) + \gamma(t_1[i] \rightarrow \lambda)$

for $j = 1$ to $|T_2|$
  $D(\theta, F_2[j]) = \sum_{k=1}^{n_j} D(\theta, T_2[j_k])$
  $D(\theta, T_2[j]) = D(\theta, F_2[j]) + \gamma(\lambda \rightarrow t_2[j])$

for $i = 1$ to $|T_1|$
  for $j = 1$ to $|T_2|$

$$D(F_1[i], F_2[j]) = \min \begin{cases} D(F_1[i], \theta) + \min\limits_{1 \le s \le n_i} \{D(F_1[i_s], F_2[j]) - D(F_1[i_s], \theta)\} \\ D(\theta, F_2[j]) + \min\limits_{1 \le t \le n_j} \{D(F_1[i], F_2[j_t]) - D(\theta, F_2[j_t])\} \\ \min\limits_{|PF(i,j)|=\min\{n_i,n_j\}} \gamma(PF(i,j)) \end{cases}$$

$$D(T_1[i], T_2[j]) = \min \begin{cases} D(T_1[i], \theta) + \min_{1 \le s \le n_i} \{D(T_1[i_s], T_2[j]) - D(T_1[i_s], \theta)\} \\ D(\theta, T_2[j]) + \min_{1 \le t \le n_j} \{D(T_1[i], T_2[j_t]) - D(\theta, T_2[j_t])\} \\ D(F_1[i], F_2[j]) + \gamma(t_1[i] \rightarrow t_2[j]) \end{cases}$$

5.2 Complexity

The complexity of computing $D(T_1[i], T_2[j])$ is, by lemma 6, bounded by $O(n_i + n_j)$. The complexity of computing $D(F_1[i], F_2[j])$ is bounded by the $O(n_i + n_j)$ plus the complexity of the minimum cost maximum flow computation.

Our graph is a graph with integer capacities, nonnegative edge costs, and maximum flow $f^* = max\{n_i, n_j\}$. The complexity of finding minimum cost maximum flow for such a graph with n vertices and m edges is $O(m|f^*| \log_{(2+m/n)} n) \le O(m|f^*| \log_2 n)$ [6]. For our graph, $n = n_i + n_j + 3$ and $m = n_i * n_j + 2n_i + n_j$; therefore the complexity is bounded by $O(n_i * n_j * \max\{n_i, n_j\} * \log_2(max\{n_i, n_j\}))$.

Hence for any pair i and j, the complexity of computing $D(T_1[i], T_2[j])$ and $D(F_1[i], F_2[j])$ is bounded by $O(n_i * n_j * \max\{n_i, n_j\} * \log_2(max\{n_i, n_j\}))$. Therefore the complexity of our algorithm is

$$\sum_{i=1}^{|T_1|} \sum_{j=1}^{|T_2|} O(n_i \times n_j \times \max\{n_i, n_j\} \times \log_2(max\{n_i, n_j\}))$$

$$\leq \sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|} O(n_i \times n_j \times \max\{deg(T_1), deg(T_2)\} \times \log_2(max\{deg(T_1), deg(T_2)\}))$$

$$\leq O(\max\{deg(T_1), deg(T_2)\} \times \log_2(max\{deg(T_1), deg(T_2)\}) \times \sum_{i=1}^{|T_1|} n_i \times \sum_{j=1}^{|T_2|} n_j)$$

$$\leq O(|T_1| \times |T_2| \times \max\{deg(T_1), deg(T_2)\} \times \log_2(max\{deg(T_1), deg(T_2)\}))$$

6 Approximate Unordered Tree Matching

Approximate unordered tree matching is a natural extension of approximate string matching [3, 8, 1] and approximate ordered tree matching [11, 13]. We omit details in this section.

In this section, we use $D[i]$ to represent subtree rooted at $d[i]$ and use $D_f[i]$ to represent the forest obtained by removing $d[i]$ from $D[i]$.

We first define the operation of removing at a node.

Removing at node $d[i]$ means removing the subtree rooted at $d[i]$.

Define a subtree set $S(D)$ as follows: $S(D)$ is a set of numbers satisfying:

(1) $i \in S(D)$ implies that $1 \leq i \leq |D|$

(2) $i, j \in S(D)$ implies that neither is an ancestor of the other.

Define $R(D, S(D))$ to be the tree D with removing at all nodes in $S(D)$.

Now we can give the definition of approximate unordered tree matching. Given tree D and P, for each i, we want to compute

$$D_r(D[i], P) = \min_S\{D(R(D[i], S(D[i])), P).$$

The minimum here is over all possible subtree sets $S(D[i])$.

In the following we give an algorithm for approximate unordered tree matching with time complexity $O(|P| \times |D| \times deg(P) \times \log_2(max\{deg(P), deg(D)\}))$. The algorithm needs a subroutine to compute $\min_{PF(i,j)} \gamma_r(PF(i,j))$. We can similarly reduce this problem to the minimum cost flow problem. However we have to modify the definition of the cost of a partial function:

$$\gamma_r(PF(i,j)) = \sum_{(s,t)\in PF(i,j)} D_r(D[i_s], P[j_t]) + \sum_{s\notin PF(i,j)} D_r(\theta, P[i_s]).$$

Input: D and P
Output: $D_r(D[i], P)$, where $1 \leq i \leq |D|$

$D(\theta, \theta) = 0$;
for $i = 1$ to $|D|$
    $D_r(D_f[i], \theta) = 0$
    $D_r(D[i], \theta) = 0$

for $j = 1$ to $|P|$
    $D_r(\theta, P_f[j]) = \sum_{k=1}^{n_j} D_r(\theta, P[j_k])$
    $D_r(\theta, P[j]) = D_r(\theta, P_f[j]) + \gamma(\lambda \rightarrow p[j])$

for $i = 1$ to $|D|$
    for $j = 1$ to $|P|$

$$D_r(D_f[i], P_f[j]) = \min \begin{cases} \min\limits_{1 \leq s \leq n_i} D_r(D_f[i_s], P_f[j]) + \gamma(d[i_s] \rightarrow \lambda) \\ D_r(\theta, P_f[j]) + \min\limits_{1 \leq t \leq n_j} D_r(D_f[i], P_f[j_t]) - D_r(\theta, P_f[j_t]) \\ \min\limits_{PF(i,j)} \gamma_r(PF(i,j)) \end{cases}$$

$$D_r(D[i], P[j]) = \min \begin{cases} D_r(\theta, P[j]) \\ \gamma(d[i] \rightarrow \lambda) + \min_{1 \leq s \leq n_i} D_r(D[i_s], P[j]) \\ D_r(\theta, P[j]) + \min_{1 \leq t \leq n_j} D_r(D[i], P[j_t]) - D_r(\theta, P[j_t]) \\ D_r(D_f[i], P_f[j]) + \gamma(d[i] \rightarrow p[j]) \end{cases}$$

7 Conclusion

Motivated by the NP-complete results in [2, 12], we have defined a constrained editing distance metric between unordered labeled trees. We present an algorithm for computing this distance metric based on a reduction to the minimum cost maximum flow problem. Our algorithm is generalizable with the same complexity to the approximate unordered tree matching problem.

The work presented is part of a project to develop a comprehensive tool for approximate tree pattern matching [9]. The proposed algorithms and their implementation will be integrated into this tool.

References

1. G. M. Landau and U. Vishkin, 'Fast parallel and serial approximate string matching', *J. Algorithms*, vol. 10, pp.157-169, 1989
2. Pekka Kilpelainen and Heikki Mannila, 'Ordered and unordered tree inclusion', To appear *SIAM J. on Computing*
3. P. H. Sellers, 'The theory and computation of evolutionary distances' *J. Algorithms* vol. 1, pp.359-373, 1980
4. Bruce Shapiro and Kaizhong Zhang, 'Comparing multiple RNA secondary structures using tree comparisons' *Comput. Appl. Biosci.* vol. 6, no. 4, pp.309-318, 1990
5. K. C. Tai, 'The tree-to-tree correction problem', *J. ACM*, vol. 26, pp.422-433, 1979
6. Robert E. Tarjan, 'Data structures and network algorithms', CBMS-NSF Regional Conference Series in Applied Mathematics, 1983

7. E. Tanaka and K. Tanaka, 'The tree-to-tree editing problem', *International Journal of Pattern Recognition and Artificial Intelligence*', vol. 2, no. 2, pp.221-240 1988
8. E. Ukkonen, 'Finding approximate patterns in strings', *J. Algorithm*, vol. 6, pp.132-137, 1985
9. Jason T.L. Wang, Kaizhong Zhang, Karpjoo Jeong and Dennis Shasha 'ATBE: A system for approximate tree matching', To appear *IEEE Trans. on Knowledge and Data Engineering*
10. Kaizhong Zhang, 'An editing based distance between ordered labeled trees', In preparation.
11. Kaizhong Zhang and Dennis Shasha, 'Simple fast algorithms for the editing distance between trees and related problems', *SIAM J. Computing* vol. 18, no. 6, pp.1245-1262, 1989
12. Kaizhong Zhang, Rick Statman and Dennis Shasha, 'On the editing distance between unordered labeled trees' *Information Processing Letters* no. 42, pp.133-139, 1992
13. Kaizhong Zhang, Dennis Shasha, and Jason Wang, 'Approximate tree matching in the presence of variable length don't cares', To appear *J. Algorithms*