数理解析研究所講究録 1426 巻 2005 年 20-25

A Hierarchy of Tree Edit Distance Measures

久保山 哲二*1 申 吉浩*2 宮原 哲浩*3
Tetsuji Kuboyama Kilho Shin Tetsuhiro Miyahara

*1東京大学 国際・産学共同研究センター
Center for Collaborative Research, University of Tokyo

*2東京大学 先端科学技術センター
Research Center for Advanced Science and Technology, University of Tokyo

*3広島市立大学 情報科学部
Faculty of Information Sciences, Hiroshima City University

The notion of tree edit distance provides a unifying framework for measuring distance and finding approximate common patterns between two trees. A diversity of tree edit distance measures have been proposed to deal with tree related problems, such as minor containment, maximum common subtree isomorphism, maximum common embedded subtree, and alignment of trees. These classes of problems are characterized by the conditions of the edit mappings, which specify how to accociate nodes in one tree with nodes in the other. In this paper, we study the relationship between classes of edit distance measures. In prior work, some of the edit mappings have often been misstated, and not well-formalized. So, we rectify these misstatements, and establish a new hierarchy among the classes of edit distance measures with a few new classes; for examles, we establish the relationship between tree edit distance and alignment of trees by showing that the mapping condition for alignment of trees is identical to that for a variant of edit distance, called less-constrained edit distance.

1. Introduction

The *tree edit distance* was introduced in [1, 2] as a natural generalization of string edit distance [3, 4]. The methods of comparing and matching tree structures using tree edit distance enjoy a wide range of applications in computational biology [5, 6, 7], image analysis [8], pattern recognition [9], natural language processing [10], information extraction from Web pages [11], and many others.

The tree edit distance between two trees is defined as the minimum cost of edit operations to transform one tree into the other. The standard set of operations includes: (1) *relabeling* a node v; (2) *deleting* a node v (and contracting the edge between v and its parent); (3) *inserting* a new node v under a node w (and moving a consecutive w's children and all their descendants under v).

Edit distance measures for trees have, in general, two aspects in giving the definitions: a sequence of operations, and an *edit mapping*. An edit mapping is a collection of node-to-node correspondences between two trees. The conditions of edit mappings specify the matching semantics in finding the similarities between two trees, and give declarative definitions of edit distance measures. In prior work, a hierarchy among the classes of edit mappings is established [12, 13]. However, a few conditions of edit mappings were misstated, and not well-defined.

In this paper, we give a new mathematical formulation for tree edit distance to elucidate the relationships among tree edit distance measures. By the formulation, we focus on the definitions of edit mappings, and rectify existing misstatements and redundancies with respect to tree edit distance. Moreover, we prove the equivalence between alignment of trees[14] and less-constrained edit distance[15].

連絡先: *1kuboyama@ccr.u-tokyo.ac.jp,
*2kilho_shin@mpeg2.rcast.u-tokyo.ac.jp,
*3miyahara@its.hiroshima-cu.ac.jp

The rest of this paper is organized as follows: the next section describes tree edit distance in an operational way, followed by our new formulation of tree edit distance to give a declarative semantic in Section 3. In Section 4, we formulate five types of tree edit distance measures based on our formulation. In Section 5, we establish a new hierarchical view of tree edit distance measures, which includes our main theorem, the equivalence between alignment of trees and less-constrained edit distance.

2. Tree Edit Distance

Unless otherwise stated, all trees we consider in this paper are rooted, labeled, and unordered trees.

2.1 Operational Definition

The tree edit distance between two trees is defined as the minimum cost of elementary edit operations to transform one tree into the other. In transforming one tree to the other, some elementary edit operations are introduced [1, 2].

Let α be a labeling function which assigns a label from a set $\Sigma = \{a, b, c, \ldots\}$ to each node. Let λ denote the unique null symbol not in Σ.

Definition 1. An *edit operation* on a tree T is any of the following three operations:

- *deletion* of a non-root node $v \in V$ from T, moving all children of v right under the parent of v; denoted by $\alpha(v) \to \lambda$,
- *insertion* of a new node $v \notin V$ as a child of a node $w \in V$, moving a consecutive subsequence of w's children (and their descendants) right under the new node v; note that this operation is the reverse of deletion; denoted by $\lambda \to \alpha(v)$,
- *relabeling* of the label of a node $v \in V$ with the label of a new node $w \notin V$; denoted by $\alpha(v) \to \alpha(w)$.

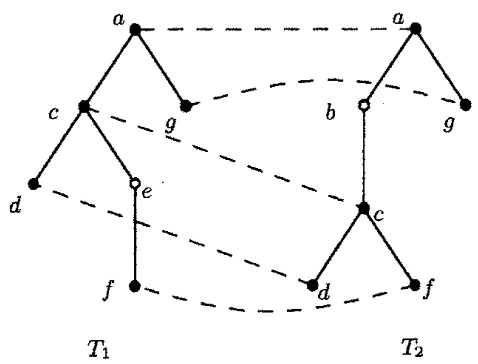

Figure 1: An example: the dashed lines between nodes denote an edit mapping.

These operations are used to transform a tree T_1 to a tree T_2. Note that all the operations are applied to only T_1. Let S be a sequence of edit operations to transform T_1 to T_2. Let γ be a cost function of edit operations. γ is defined to be a distance metric as follows: for $a, b, c \in \Sigma \cup \{\lambda\}$, (i) $\gamma(a \to b) \geq 0$; (ii) $\gamma(a \to b) = \gamma(b \to c)$; and (iii) $\gamma(a \to c) \leq \gamma(a \to b) + \gamma(b \to c)$. The cost function γ for edit operations is generalized for sequences S of edit operations by letting $\gamma(S) = \Sigma_{s \in S}\gamma(s)$.

The edit distance between T_1 and T_2 is defined [1] as

$$D(T_1, T_2) = \min_S\{\gamma(S)\}.$$

2.2 Edit Mappings

The effect of a sequence of edit operations is reduced to a structure called *edit mapping* [1], which is comparable to *trace* [3] in string edit distance. An *edit mapping* depicts node-to-node correspondences between two trees according to the structural similarity, or shows how nodes in one tree are preserved after transformed to the other.

Definition 2. An *edit mapping* from a tree T_1 to a tree T_2 is a set $M \subseteq V(T_1) \times V(T_2)$ such that, for all (x_1, x_2), $(y_1, y_2) \in M$, $x_1 = y_1 \Leftrightarrow x_2 = y_2$.

Note that this definition does not require M to preserve ancestor-descendant relation. For simplicity, We refer to the edit mapping as the mapping. The edit mapping provides a qualitative view of edit distance. Let M be a base mapping. The mapping cost of M is defined as

$$\begin{aligned}\gamma(M) = \quad & \textstyle\sum_{(v_1,v_2)\in M}\gamma(\alpha(v_1) \to \alpha(v_2)) + \\ & \textstyle\sum_{v_1 \in V_{\overline{M}}(T_1)}\gamma(\alpha(v) \to \lambda) + \\ & \textstyle\sum_{v_2 \in V_{\overline{M}}(T_2)}\gamma(\lambda \to \alpha(v_2)).\end{aligned}$$

The following theorem is due to Taï [1].

Theorem 1 ([1]). Let S be a sequence of edit operations to transform T_1 to T_2, and M a mapping from T_1 to T_2.

$$D(T_1, T_2) = \min_S\{\gamma(S)\} = \min_M\{\gamma(M)\}.$$

This theorem plays the role of a bridge between an operational definition and a declarative definition for the edit distance. For example, Fig. 1 shows an edit mapping.

The rest of this subsection we show a number of existing tree edit distance measures by their mapping conditions.

2.2.1 Standard Mapping: $\mathcal{S}$

This mapping characterizes the standard edit distance by Zhang *et al.* [16].

Definition 3. A mapping M is *standard* if the following condition holds:
(S) $\forall(x_1, x_2), (y_1, y_2) \in M[x_1 < x_2 \Leftrightarrow y_1 < y_2]$.

Computing the edit distance based on the genealogical mapping is known to be NP-complete [16], even for binary trees having a label alphabet of size two.

2.2.2 Top-down Mapping: $\mathcal{TD}$

This mapping characterizes the edit distance in which insertion and deletion operations are applied only to leaves. The top-down mapping originated in Selkow [17], and Yang [18] gave an algorithm of computing an edit distance based on the top-down mapping for ordered trees. Our definition is slightly different from the definition in [12] since it is not well-defined.

Definition 4. A mapping $M = M(T_1, T_2)$ is *top-down* if the following condition holds:
(TD) $M \neq \emptyset \Rightarrow [(r(T_1), r(T_2)) \in M \wedge [(x_1, x_2) \in M$
$\wedge x_1 \neq r(T_1) \wedge x_2 \neq r(T_2) \Rightarrow (p(x_1), p(x_2)) \in M]]$.

2.2.3 Constrained Mapping: $\mathcal{C}$

The constrained mapping was introduced by Zhang *et al.* to circumvent the negative results that computing the edit distance for unordered labeled trees is NP-complete [16] (in fact MAX SNP-hard [19]).

Definition 5 (Zhang [20]). A mapping M is *constrained* if the following condition holds:
(C) $\forall(x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[z_1 < x_1 \smile y_1 \Leftrightarrow z_2 < x_2 \smile y_2]$.

2.2.4 Structure-Respecting Mapping: $\mathcal{SR}$

This mapping was introduced by Richter [21] to deal with syntactic trees.

Definition 6 (Richter [21]). A mapping M is *structure-respecting* if the following condition holds:
(SR) $\forall(x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$,
any of x_1, y_1, z_1 is not
an ancestor of any of the others,
$[x_1 \smile y_1 = x_1 \smile z_1 \Leftrightarrow x_2 \smile y_2 = x_2 \smile z_2]$.

The following proposition asserts that M being constrained is equinalent with M being structure-respecting, which was stated in Lu et. al [15] without proof.

Proposition 2. For a mapping M, the following are equivalent:

1. M is standard and satisfies the following:
(SR') $\forall(x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
[any of x_1, y_1, z_1 is not an ancestor of any of the others,

$x_1 \smile y_1 < x_1 \smile z_1 \Leftrightarrow$
any of x_2, y_2, z_2 is not an ancestor of any of the others,
$x_2 \smile y_2 < x_2 \smile z_2$],

2. M is structure-respecting, and
3. M is constrained.

Proof. **(1)⇒(2):** We prove the contraposition of (SR). If $x_2 \smile y_2 \neq x_2 \smile z_2$, we may assume $x_2 \smile y_2 < x_2 \smile z_2$ since $x_2 \smile y_2$ and $x_2 \smile z_2$ are comparable. $x_1 \smile y_1 < x_1 \smile z_1$ immediately follows by (SR'). **(2)⇒(3):** Assume that $z_1 < x_1 \smile y_1$. If z_2 and $x_2 \smile y_2$ are comparable, $z_2 < x_2 \smile y_2$ holds by (S) (if $z_2 \geq x_2 \smile y_2$, then $z_1 \geq x_1$ and $z_1 \geq y_1$ hold by (S), which contradicts to the assumption $z_1 < x_1 \smile y_1$). (i) If any two of x_1, y_1, z_1 are comparable, i.e. z_1 is comparable with x_1 or y_1 (if $x_1 \leq y_1$, then $z_1 < x_1 \smile y_1 = y_1$), z_2 and $x_2 \smile y_2$ are comparable by (S). (ii) Suppose that any of x_1, y_1, z_1 is not an ancestor of any of the others. Since we may assume that $x_1 \smile y_1 = x_1 \smile z_1$ without loss of generality, $x_2 \smile z_2 = x_2 \smile y_2$ holds by (SR). Therefore, z_2 and $x_2 \smile y_2$ are comparable, too. **(3)⇒(1):** Assume that $x_1 \smile y_1 < x_1 \smile z_1$ and any of x_1, y_1, z_1 is not an ancestor of any of the others. By (S), we have any of x_2, y_2, z_2 is not an ancestor of any of the others. We have $z_2 \not\leq x_2 \smile y_2$, since $x_1 \leq z_1$ follows $z_2 = x_2 \smile y_2$ by (S) and $z_1 < x_1 \smile y_1$ does $z_2 < x_2 \smile y_2$ by (C). Therefore, $x_2 \smile y_2 < x_2 \smile z_2$ holds. □

3. Theoretical Foundation for Tree Edit Distance

In this section, we give a new formulation of tree edit distance.

3.1 Rooted Trees

Definition 7. A *rooted tree* $T = (V, \leq)$ is a nonempty, finite, and partially ordered set with the maximum element $r(T) \in V$ called the *root*, and such that $\{w \in V | v \leq w\}$ is a totally ordered subset of V for every $v \in V$.

We call the elements of V the *nodes* of T, and denote the set of all nodes in T by $V(T)$. We let $E(T) = \{(x, y) \in V(T) \times V(T) | (x < y) \wedge \nexists z \in V(T)[x < z < y]\}$. The element of $E(T)$ is called an *edge* of T. A node y such that $x \leq y$ is an *ancestor* of x. If $x \leq y$ and $x \neq y$, then y is a *proper ancestor* of x, denoted by $x < y$. The *parent* of node x is the minimum nodes of proper ancestors of x, denoted by $p(()x)$. A *leaf* of a tree T is a minimal node in T. The size of a tree T is the number of nodes in T, denoted by $|T|$.

Definition 8. For an arbitrary rooted tree $T = (V, \leq)$, a *common ancestor* of $U \subseteq V$ is an element $x \in V$, if exists, such that for all $y \in U$, $y \leq x$. A common ancestor x of U is the *least common ancestor* of U if, for any common ancestor y of U, $x \leq y$ holds. We denote the least common ancestor of U by lca U, and lca $\{x, y\}$ by $x \smile y$.

Lemma 3. The following properties hold in terms of the least common ancestor:

1. $x \smile x = x$,
2. $x \smile y = y \smile z$,
3. $(x \smile y) \smile z = x \smile (y \smile z)$,
4. $x \leq y \Leftrightarrow x \smile y = y$,
5. $x \smile y < x \smile z \Rightarrow y \smile z = x \smile z$, and
6. $x \smile y = x \smile z \Rightarrow y \smile z \leq x \smile y$.

Corollary 4. For any three nodes x, y, z, either of the following properties holds:

1. $x \smile y < x \smile z$, and $x \smile z = y \smile z$,
2. $x \smile y = x \smile z$, and $y \smile z \leq x \smile z$,
3. $x \smile y > x \smile z$, and $x \smile y = y \smile z$

Proof. It follows straightforwardly from Lemma 3-(5), and (6). □

3.2 Tree Homomorphism and Isomorphism

Definition 9. Let T_1 and T_2 be two trees. A *homomorphism* from T_1 to T_2 is a mapping ϕ from $V(T_1)$ to $V(T_2)$ such that

1. $\phi(r(T_1)) = r(T_2)$, and
2. $x < y \Rightarrow \phi(x) \leq \phi(y)$.

We refer to $\phi : V(T_1) \to V(T_2)$ as $\phi : T_1 \to T_2$ if there is no confusing.

Proposition 5. The composition of homomorphisms is a homomorphism.

Definition 10. Let T_1 and T_2 be two trees. An *isomorphism* from T_1 to T_2 is a bijection ϕ from $V(T_1)$ to $V(T_2)$ such that

$$(x, y) \in E(T_1) \Leftrightarrow (\phi(x), \phi(y)) \in E(T_2).$$

Proposition 6. Every isomorphism is also a homomorphism.

Proposition 7. Let T_1 and T_2 be two trees. Suppose that ϕ is a bijection from T_1 to T_2, then the following conditions are equivalent:

1. ϕ is an isomorphism, and
2. $\phi(x) < \phi(y) \Rightarrow x < y$

Proposition 8. A mapping ϕ from a tree T to T is an isomorphism if and only if ϕ is an identity mapping on $V(T_1)$.

3.3 Embedding and Insertion

We first define an embedding, which is regarded as consecutive insertions of nodes into a tree.

3.3.1 Embedding

Definition 11. Let T_1 and T_2 be two trees. An *embedding* ϕ from T_1 to T_2 is an injection from $V(T_1)$ to $V(T_2)$ such that

1. ϕ is a homomorphism, and
2. $\phi(x) < \phi(y) \Rightarrow x < y$.

We define $\mathrm{red}(\phi) = |V(T_2) \setminus \phi(V(T_1))|$ as the *redundancy* of the embedding ϕ from T_1 to T_2.

Proposition 9. Suppose that ϕ be a mapping from a tree T_1 to a tree T_2, and ψ be an embedding from T_2 to a tree T_3, then the following conditions hold:

1. if ϕ is an embedding, $\psi \circ \phi$ is also an embedding, and
2. if $\psi \circ \phi$ is an embedding, ϕ is also an embedding.

In both cases, $\mathrm{red}(\psi \circ \phi) = \mathrm{red}(\phi) + \mathrm{red}(\psi)$.

3.3.2 Insertion

Now, we are ready to give a declarative definition of the insertion operation.

Definition 12. Let T_1 and T_2 be two trees, and v a node in T_2. An embedding ϕ from T_1 to T_2 is an *insertion* of v into T_1 if $\phi(V(T_1)) = V(T_2) \setminus \{v\}$.

Proposition 10. Let ϕ be an embedding from a tree T_1 to a tree T_2, and ϕ also an insertion of a node v into T_1. If v be a node in T_2 such that $v \neq r(T_2)$, then there exists an insertion of v to T_1.

Any insertion of v is uniquely determined except that the insertion is an isomorphism. Hence, by i_v, we denote the insertion of v.

The following theorem proves that Definition 12 of the insertion is equivalent to the operational definition of the insertion.

Theorem 11. Let ϕ be an embedding from T_1 to T_2 with $V(T_1) \setminus \phi(V(T_1)) = \{v_1, \ldots, v_n\}$. There exist a sequence of trees $S_0, S_1, \ldots, S_n$, and insertions $\phi_i : S_i \rightarrow S_i - 1$ ($i \in \{1, \ldots, n\}$) such that

1. $S_0 = T_2$,
2. $S_n = T_1$,
3. $\phi_1 \circ \cdots \circ \phi_i(V(S_i)) = V(T_2) \setminus \{v_1, \ldots, v_i\}$, and
4. $\phi = \phi_n \circ \cdots \circ \phi_1$

$$\begin{tikzcd}
S_n \arrow[r, "\phi_n", "\mathrm{Ins}_{x_n}"'] \arrow[d, equal] & S_{n-1} \arrow[r, "\phi_{n-1}", "\mathrm{Ins}_{x_{n-1}}"'] & \cdots \arrow[r, "\phi_2", "\mathrm{Ins}_{x_2}"'] & S_1 \arrow[r, "\phi_1", "\mathrm{Ins}_{x_1}"'] & S_0 \arrow[d, equal] \\
T_1 \arrow[rrrr, "\phi"] & & & & T_2 .
\end{tikzcd}$$

3.4 Degeneration and Deletion

We define a degeneration, which is regarded as consecutive deletions of nodes from a tree.

3.4.1 Degeneration

Definition 13. Let T_1 and T_2 be two trees. A *degeneration* ϕ from T_1 to T_2 is a surjection from $V(T_1)$ to $V(T_2)$ such that

1. $\phi(x) = \phi(y) \Rightarrow \phi(x \smile y) = \phi(x) = \phi(y)$, and
2. $\phi(x) < \phi(y) \Rightarrow \exists y'[\phi(x) = \phi(y') \wedge x < y']$.

We define $\mathrm{Dup}(\phi) = \{x \in V(T_1) | \phi(x) = \phi(p(x))\}$ as the *duplication* of the degeneration ϕ from T_1 to T_2.

Proposition 12. Let T_1 and T_2 be two trees, and ϕ be a degeneration from T_1 to T_2. There exists a unique embedding ψ from T_2 to T_1 such that $\phi \circ \psi$ is the identity mapping on $V(T_1)$, and $\psi \circ \phi$ is the identity mapping on $V(T_2) \setminus \mathrm{Dup}(\phi)$.

We denote the degeneration corresponding to an embedding ϕ denoted by $\bar{\phi}$.

3.4.2 Deletion

Definition 14. Let T_1 and T_2 be two trees, and v a node in T_2. A degeneration ϕ from T_1 to T_2 is *deletion* of v from T_1 if $\mathrm{Dup}(\phi) = \{v\}$.

Theorem 13. Let ϕ be a degeneration from T_1 to T_2 with $\mathrm{Dup}(\phi) = \{v_1, \ldots, v_n\}$. There exist a sequence of trees $S_0, S_1, \ldots, S_n$, and deletions $\phi_i : S_i \rightarrow S_i - 1$ ($i \in \{1, \ldots, n\}$) such that

1. $S_0 = T_1$,
2. $S_n = T_2$,
3. $\mathrm{Dup}(\phi_n \circ \cdots \circ \phi_1) = \{v_1, \ldots, v_i\}$, and
4. $\phi = \phi_{n-1} \circ \phi_0$;

$$\begin{tikzcd}
S_0 \arrow[r, "\phi_0", "\mathrm{Del}_{x_0}"'] \arrow[d, equal] & S_1 \arrow[r, "\phi_1", "\mathrm{Del}_{x_1}"'] & \cdots \arrow[r, "\phi_{n-2}", "\mathrm{Del}_{x_{n-2}}"'] & S_{n-1} \arrow[r, "\phi_{n-1}", "\mathrm{Del}_{x_{n-1}}"'] & S_n \arrow[d, equal] \\
T_1 \arrow[rrrr, "\phi"] & & & & T_2 .
\end{tikzcd}$$

4. Characterization of Edit Distance Measures

In this section, we consider the edit mapping conditions for unordered trees, and introduce a few of new edit mapping conditions to investigate the relationship among known classes of edit mappings. Due to space limitation, most of the proofs are omitted.

For an edit mapping M from T_1 to T_2, we define:

$$\begin{aligned}
V_M(T_1) &= \{x \in V(T_1) | \exists x \in V(T_2) \text{ s.t. } (x, y) \in M\}, \\
V_M(T_2) &= \{y \in V(T_2) | \exists y \in V(T_1) \text{ s.t. } (x, y) \in M\}, \\
V_{\overline{M}}(T_1) &= V(T_1) \setminus V_M(T_1), \\
V_{\overline{M}}(T_2) &= V(T_2) \setminus V_M(T_2).
\end{aligned}$$

4.1 Alignable Mapping: $\mathcal{A}$

The alignment of trees was introduced by Jiang *et al.* [14], and efficient algorithm for similar trees were proposed for ordered trees [22] and unordered trees [23]. The definition of the alignment has been given in an operational way [14, 12, 13].

We give a new definition of alignment of trees.

Definition 15. A mapping M from T_1 to T_2 is *alignable* if and only if there exists a triplet (U, ϕ, ψ) such as

1. $\phi : T_1 \rightarrow U$ is an embedding,
2. $\psi : T_2 \rightarrow U$ is an embedding, and
3. $\forall (x, y) \in M[\phi(x) = \psi(x)]$;

$$\begin{tikzcd}
& U & \\
T_1 \arrow[ur, "\varphi"] \arrow[rr, "M"] & & T_2 . \arrow[ul, "\psi"']
\end{tikzcd}$$

Figure 2 illustrates an example of an alignable mapping.

Lemma 14. Suppose that T_1 and T_2 are two trees, and $M \subseteq V(T_1) \times V(T_2)$ is an alignable mapping (U, ϕ, ψ), then the following condition holds:

$$M = \{(x, \bar{\psi}(\phi(x))) | x \in V_M(T_1)\}.$$

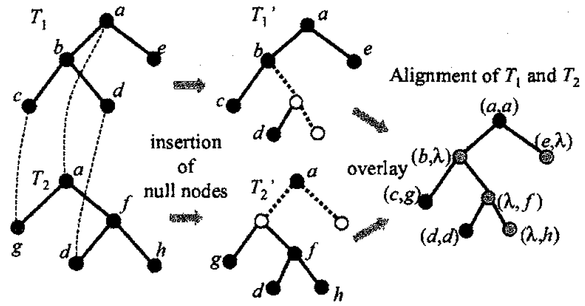

Figure 2: An alignable mapping from T_1 to T_2: the lines between two trees indicate an alignable mapping.

We give a few properties of alignable mappings.

Lemma 15. Let T_1 and T_2 be two trees. Any singleton mapping $M = \{(x, y)\}$ from T_1 to T_2 is alignable.

Lemma 16. Let T_1, T_1', T_2 and T_2' be four trees, and M an alignable mapping from T_1 to T_2. For two insertions $\phi : T_1 \to T_1'$ and $\psi : T_2 \to T_2'$ which both do not necessarily preserve their roots, the mapping $M' = \{(\phi(x), \psi(y)) | (x, y) \in M\}$ is an alignable mapping from T_1' to T_2'.

Lemma 17. Let $T_1 = \{r(T_1)\{T_{1,1}, T_{1,2}\}\}$ and $T_2 = \{r(T_2)\{T_{2,1}, T_{2,2}\}\}$ be two trees, M a mapping from T_1 to T_2. The mapping M is alignable from T_1 to T_2 if the following conditions hold:

1. $\forall (x, y) \in M[x \in V(T_{1,i}) \Leftrightarrow y \in V(T_{2,i})]$, for $i \in \{1, 2\}$,
2. $M_i = M \cap (V(T_{1,i}) \times V(T_{2,i}))$ is an alignable mapping from $T_{1,i}$ to $T_{2,i}$.

4.2 Less Constrained Mapping: $\mathcal{L}$

The less-constrained mapping was introduced in [15] to relax the condition of the constrained mapping. The definition of the mapping in [15] is not correct. We rectify it and give a new mapping definition as follows.

Definition 16. A mapping M is *less-constrained* if the following conditions hold:

(L0) $\forall (x_1, x_2), (y_1, y_2) \in M\ [x_1 < x_2 \Leftrightarrow y_1 < y_2]$,

(L1) $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_1 \smile y_1 < x_1 \smile z_1 \Rightarrow x_2 \smile y_2 = x_2 \smile z_2]$,

(L2) $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_2 \smile y_2 < x_2 \smile z_2 \Rightarrow x_1 \smile y_1 = x_1 \smile z_1]$.

4.3 Confucianistic Mapping: $\mathcal{CF}$

We introduce a new mapping, the confucianistic mapping, which lives up to its name since this mapping respects ancestor-descendant relation between two trees.

Definition 17. A mapping M is *confucianistic* if the following conditions hold:

(CF1) $\forall (w_1, w_2), (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[w_1 \smile x_1 < y_1 \smile z_1 \Rightarrow w_2 \smile x_2 \leq y_2 \smile z_2]$

(CF2) $\forall (w_1, w_2), (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[w_2 \smile x_2 \leq y_2 \smile z_2 \Rightarrow w_1 \smile x_1 < y_1 \smile z_1]$

4.4 Triangular Mapping: $\mathcal{T}$

We introduce the triangular mapping as follows.

Definition 18. A mapping M is *triangular* if the following condition holds:

(T) $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_1 \smile y_1 < x_1 \smile z_1 \Leftrightarrow x_2 \smile y_2 < x_2 \smile z_2]$.

4.5 Quasi-Triangular Mapping: $\mathcal{QT}$

This mapping is obtained by relaxing the condition of the triangular mapping.

Definition 19. A mapping M is *quasi-triangular* if the following condition holds:

(QT1) $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_1 \smile y_1 < x_1 \smile z_1 \Rightarrow x_2 \smile y_2 = x_2 \smile z_2]$, and

(QT2) $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_2 \smile y_2 < x_2 \smile z_2 \Rightarrow x_1 \smile y_1 = x_1 \smile z_1]$.

5. Hierarchy of the Mapping Classes

Proposition 18. If the condition of the triangular mapping holds, then that of the constrained mapping also holds, and not vice versa.

Proof. From the premise $z_1 < x_1 \smile y_1$, we may assume, without loss of generality, $x_1 \smile z_1 = x_1 \smile y_1$. Hence, we have $z_1 < x_1 \smile z_1$. By $z_1 = z_1 \smile z_1$ and the condition (T), we have $z_2 \smile z_2 < x_2 \smile z_2$. It follows that $z_2 < x_2 \smile z_2$. Moreover, by the condition (S), which is equivalent to the condition (T), we have $x_2 \smile z_2 = x_2 \smile y_2$. Therefore, $z_2 < x_2 \smile y_2$. □

Lemma 19. The constrained mapping implies the ancestor-descendant relation.

Proof. According to the condition (C), for all $(x_1, x_2), (y_1, y_2) \in M$, $x_1 < y_1 \smile y_1 \Leftrightarrow x_2 < y_2 \smile y_2$. Hence, we immediately have $x_1 < y_1 \Leftrightarrow x_2 < y_2$. □

Proposition 20. A mapping M is confucianistic if and only if M is genealogical and quasi-triangular, and the following conditions hold:

1. $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_1 \smile y_1 = y_1 \smile z_1 = z_1 \smile x_1 \notin \{x_1, y_1, z_1\}$
$\Rightarrow x_2 \smile y_2 = y_2 \smile z_2 = z_2 \smile x_2]$
2. $\forall (x_1, x_2), (y_1, y_2), (z_1, z_2) \in M$
$[x_2 \smile y_2 = y_2 \smile z_2 = z_2 \smile x_2 \notin \{x_2, y_2, z_2\}$
$\Rightarrow x_1 \smile y_1 = y_1 \smile z_1 = z_1 \smile x_1]$

Theorem 21. The condition of the alignable mapping is equivalent to that of the less-constrained mapping.

The following hierarchy of the mapping classes is established.

Theorem 22.
1. $\mathcal{TD} \subset \mathcal{T} \subset \mathcal{SR} = \mathcal{C} \subset \mathcal{A} = \mathcal{L} = (\mathcal{QT} \cap \mathcal{S}) \subset \mathcal{S}$
2. $\mathcal{CF} \subset \mathcal{A}$

Figure 3 shows that the hierarchy of tree edit distance measures.

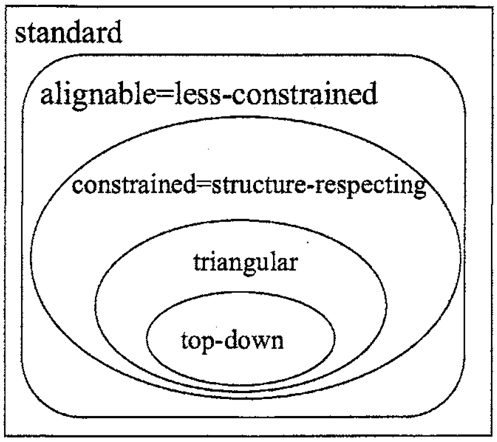

Figure 3: A hierarchy of tree edit distance measures

6. Conclusion

In this paper, we introduced a new theoretical formulation of tree edit distance, and investigated the relationship among the classes of tree edit distance. We then rectified some misstatements and redundancies in prior work, and established a new hierarchy among the edit mapping conditions. Moreover, we showed that the mapping condition for alignment of trees is identical to that for a variant of edit distance, called less-constrained edit distance.

References

[1] K.-C. Taï. The tree-to-tree correction problem. *Journal of the ACM*, 26(3):422–433, July 1979.

[2] K. Zhang and D. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262, 1989.

[3] R.A. Wagner and M.J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.

[4] D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

[5] Y. Sakakibara. Pair hidden markov models on tree structures. *Bioinformatics*, 19:232–240, 2003.

[6] M. Höchsmann, T. Töller, R. Giegerich, and S. Kurtz. Local similarity in rna secondary structures. In *Proceedings of the Computational Systems Bioinformatics (CSB ' 03)*. IEEE, 2003.

[7] L. Wang and J. Zhao. Parametric alignment of ordered trees. *Bioinformatics*, 19(17):2237–2245, 2003.

[8] A. Torsello and E. R. Hancock. Graph clustering with tree-unions. In *LNCS*, volume 2756, pages pp. 451 – 459. Springer-Verlag Heidelberg, November 2003. ISBN: 3-540-40730-8.

[9] P. Ferraro and C. Godin. A distance measure between plant architectures. *Annals of Forest Science*, 57:445–461, 2000.

[10] M. Vilares, F. J. Ribadas, and V. M. Darriba. Approximate vldc pattern matching in shared-forest. In *LNCS*, volume 2004, pages 483–494, 2001. Proceedings of the Second International Conference on Computational Linguistics and Intelligent Text Processing.

[11] D. C. Reis, P. B. Golgher, A. S. Silva, and A. H. F. Laender. Automatic web news extraction using tree edit distance. In *WWW2004*, pages 502–511, 2004.

[12] J.T.-L. Wang and K. Zhang. Finding similar consensus between trees: an algorithm and a distance hierarchy. *Pattern Recognition*, 34:127–137, 2001.

[13] G. Valiente. An efficient bottom-up distance between trees. In *Proc. 8th Int. Symposium on String Processing and Information Retrieval*, pages 212–219. IEEE Computer Science Press, 2001.

[14] T. Jiang, L. Wang, and K. Zhang. Alignment of trees — an alternative to tree edit. *Theoretical Computer Science*, 143:137–148, 1995.

[15] C. L. Lu, Z.-Y. Su, and G. Y. Tang. A new measure of edit distance between labeled trees. In *Lecture Notes in Computer Science*, volume 2108, pages pp. 338–348. Springer-Verlag Heidelberg, 2001.

[16] K. Zhang, R. Statman, and D. Shasha. On the editing distance between unordered labeled trees. *Information Processing Letters*, 42(3):133–139, 1992.

[17] S. M. Selkow. The tree-to-tree editing problem. *Information Processing Letters*, 6(6):184–186, December 1977.

[18] Wuu Yang. Identifying syntactic differences between two programs. *Software - Practice and Experience*, 21(7):739–755, 1991.

[19] K. Zhang and T. Jiang. Some max snp-hard results concerning unordered labeled trees. *Information Processing Letters*, 49:249–254, 1994.

[20] K. Zhang. A constrained edit distance between unordered labeled trees. *Algorithmica*, 15:205–222, 1996.

[21] T. Richter. A new measure of the distance between ordered trees and its applications. Technical Report 85166-CS, Dept. of Computer Science, Univ. of Bonn, 1997.

[22] J. Jansson and A. Lingas. A fast algorithm for optimal alignment between similar ordered trees. *Fundamenta Informaticae*, 56:105–120, 2003.

[23] D. Fukagawa and T. Akutsu. Fast algorithms for comparison of similar unordered trees. In *Proc. 15th Int. Symp. Algorithms and Computation (ISAAC 2004)*, 2004.