On the Complexity of Tree Edit Distance with Variables

Tatsuya Akutsu
Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan

Tomoya Mori
Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan

Naotoshi Nakamura
The Thomas N. Sato BioMEC-X Laboratories, Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan
Karydo TherapeutiX, Inc., Tokyo, Japan
Interdisciplinary Biology Laboratory (iBLab), Division of Natural Science, Graduate School of Science, Nagoya University, Japan

Satoshi Kozawa
The Thomas N. Sato BioMEC-X Laboratories, Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan
Karydo TherapeutiX, Inc., Tokyo, Japan

Yuhei Ueno
The Thomas N. Sato BioMEC-X Laboratories, Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan
Karydo TherapeutiX, Inc., Tokyo, Japan
V-iCliniX Laboratory, Nara Medical University, Japan

Thomas N. Sato
The Thomas N. Sato BioMEC-X Laboratories, Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan
Karydo TherapeutiX, Inc., Tokyo, Japan
V-iCliniX Laboratory, Nara Medical University, Japan

Abstract

In this paper, we propose tree edit distance with variables, which is an extension of the tree edit distance to handle trees with variables and has a potential application to measuring the similarity between mathematical formulas. We analyze the computational complexity of several variants of this model. In particular, we show that the problem is NP-complete for ordered trees. We also show for unordered trees that the problem of deciding whether or not the distance is 0 is graph isomorphism complete but can be solved in polynomial time if the maximum outdegree of input trees is bounded by a constant. We also present parameterized and exponential-time algorithms for ordered and unordered cases, respectively.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases Tree edit distance, unification, parameterized algorithms

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2022.44


Funding Tatsuya Akutsu: Partially supported by JSPS KAKENHI #JP22H00532 and #JP22K19830.
Naotoshi Nakamura: Partially supported by JSPS KAKENHI #JP17H06003 and #JP19H05422.

© Tatsuya Akutsu, Tomoya Mori, Naotoshi Nakamura, Satoshi Kozawa, Yuhei Ueno, and Thomas N. Sato; licensed under Creative Commons License CC-BY 4.0
Introduction

Measuring the similarity of tree structured data is a fundamental problem in computer science because various kinds of data are represented as trees. Tree edit distance is one of the most extensively studied measures for dissimilarity between two rooted trees. It is known that tree edit distance can be computed in polynomial time for ordered trees [7, 14, 16, 18], whereas its computation is NP-hard for unordered trees [19].

Mathematical formulas are one of the widely used tree structured data. Indeed, many methods have been developed for retrieving similar mathematical formulas [1, 10, 15, 20]. Comparison of mathematical formula is also important for analysis of biological systems [17]. However, most of the developed search methods are heuristic ones and are not studied from a viewpoint of the computational complexity. Unification is a basic technique to evaluate the identity of logic formulas, and the computational complexity of various variants (e.g., allowing associative and/or commutative laws) has been studied [3, 11, 12]. However, unification does not give a similarity or distance measure. Although a combination of unification and tree edit distance was proposed under the name of “tree edit distance with variables”, only a quite restricted case (each variable can occur only once) was studied [3].

In order to compare mathematical formulas, it is important to consider trees with variables. For example, consider two functions \( f(x, y, z) \) and \( g(x, y, z) \) defined by:

\[
\begin{align*}
f(x, y, z) & = (x + y) \times z, \\
g(x, y, z) & = (x + z) \times y.
\end{align*}
\]

These two functions are essentially the same: the former one is identical to the latter one by replacing \( y \) and \( z \) with \( z \) and \( y \), respectively. In addition, consider a function \( h(x, y, z) \) defined by:

\[
h(x, y, z) = z \times (x + y).
\]

This function is also essentially the same as \( f \) and \( g \) because multiplication satisfies the commutative law. Functions \( f, g, \) and \( h \) can be respectively represented as \( T_1, T_2, \) and \( T_3 \) shown in Figure 1. If we ignore variable names assigned to leaves, these trees are identical as unordered rooted trees. However, considering variable names is important. For example, consider a function \( k \) defined by:

\[
k(x, y) = (x + y) \times x.
\]

This function can be represented as a rooted tree \( T_4 \) in Figure 1. Although (unordered) tree structures of \( T_1, \ldots, T_4 \) are identical, \( k \) is clearly different from \( f, g, \) and \( h \). Therefore, variable names assigned to leaves should be taken into account.

Based on the above discussion, we introduce tree edit distance with variables in this paper. Before giving this new distance measure, we briefly review the standard tree edit distance [6]. Let \( T_1 \) and \( T_2 \) be two rooted trees in which each node has a label from some alphabet. We consider two cases: both \( T_1 \) and \( T_2 \) are ordered trees, and both \( T_1 \) and \( T_2 \) are unordered trees. This distinction can be taken into account only when we consider whether or not two trees...
are identical (i.e., isomorphic) after tree editing operations. The tree edit distance $d_0(T_1, T_2)$ between $T_1$ and $T_2$ is defined as the cost of the minimum cost sequence of edit operations that transforms $T_1$ to $T_2$, where an operation is one of deletion of a node, insertion of a node, and change of the label of a node. Then, we define the tree edit distance between two trees in which leaves can have variables as labels $T_1$ and $T_2$ by $\text{dist}(T_1, T_2) = \min_{\theta} \text{dist}_0(T_1\theta, T_2\theta)$, where $\theta$ is a substitution (i.e., a set of assignments of constants to variables). See Section 2 for the precise definitions.

In this paper, we analyze the computational complexity of several variants/subcases of the tree edit distance problem with variables, with focusing on the unit cost model (i.e., the cost of each edit operation is 1). When discussing the complexity classes, we consider a decision version of the problem: whether or not $\text{dist}(T_1, T_2) \leq d$ for given $T_1$, $T_2$, and a given non-negative real number $d$. The main results are summarized in Table 1, where “iso” asks whether $d(T_1, T_2) = 0$, “BD” means that the maximum outdegree (i.e., the maximum number of children) of both $T_1$ and $T_2$ is bounded by a constant, $M$ denotes the number of occurrences of variables in $T_1$ and $T_2$, $n_i$ denotes the number of nodes in $T_i$, and $\delta$ is any positive constant. In this table, P, NPC, and GIC mean that the target problem is polynomial-time solvable, NP-complete, and Graph Isomorphism complete (i.e., as hard as the graph isomorphism problem under polynomial-time reduction), respectively. It is interesting to see that the complexity substantially changes according to introduction of variables.

### 2 Preliminaries

In this section, we review the precise definition of the tree edit distance and then formally define the tree edit distance with variables.

Let $T_1$ and $T_2$ be two rooted trees in which each node has a label from an alphabet $\Sigma$. We use $\ell(v)$ to denote the label of a node $v$. As mentioned in Section 1, we consider two cases: both $T_1$ and $T_2$ are ordered trees, and both $T_1$ and $T_2$ are unordered trees, and this distinction can be taken into account only when we consider whether or not two trees are identical after tree edit operations. We consider three kinds of edit operations (see also Figure 2):
We assign a cost for each editing operation: \( \gamma(a, b) \) denotes the cost of changing a node with label \( a \) to label \( b \), \( \gamma(a, e) \) denotes the cost of deleting a node labeled with \( a \), \( \gamma(e, a) \) denotes the cost of inserting a node labeled with \( a \). We assume that \( \gamma(x, y) \) satisfies the conditions of distance metric: \( \gamma(x, x) = 0 \), \( \gamma(x, y) = \gamma(y, x) \), \( \gamma(x, y) \geq 0 \), and \( \gamma(x, z) \leq \gamma(x, y) + \gamma(y, z) \). Then, the edit distance between \( T_1 \) and \( T_2 \) is defined as the cost of the minimum cost sequence of edit operations that transforms \( T_1 \) to \( T_2 \) (precisely, transforms \( T_1 \) to a tree identical to \( T_2 \)). It is well-known that this distance satisfies the conditions of distance measure, in both ordered and unordered cases.

In this paper, we focus on the unit cost model in which the cost of each edit operation is 1 (i.e., \( \gamma(x, y) = 1 \) for any \( x \neq y \)). Note that all hardness results hold for a general cost model because the unit cost model is a special case. For positive results, we need to consider mapping costs between variables in \( T_1 \) and \( T_2 \). Since it is difficult to define appropriate general costs for such cases, we only consider the unit cost model in this paper.

Here we define the tree edit distance with variables. Let \( \Sigma \) be a set of constant symbols, where each constant is denoted by a lower-case letter (e.g., \( a, b, c, x, y, z, a_1, a_2 \)). Let \( \Lambda \) be a set of variables, where each variable is denoted by an upper-case letter (e.g., \( X, Y, Z, X_1, X_2 \)). A substitution is a set of variable-constant pairs, \( \theta = \{(X_1, x_1), (X_2, x_2), \ldots, (X_k, x_k)\} \), where \( X_i \neq X_j \) holds for all \( i \neq j \) but \( x_i = x_j \) can hold for some \( (i, j) \). For a rooted tree \( T \) and a substitution \( \theta \), \( T\theta \) denotes the tree obtained by changing variables appeared in \( T \) to constants according to \( \theta \) (each \( X_i \) is replaced with \( x_i \)). Let \( dist_0(T_1, T_2) \) be the standard tree edit distance between \( T_1 \) and \( T_2 \) (i.e., distance between trees without variables). We reasonably assume the following:

- Variable symbols appear only in leaves.
- The sets of variables appearing \( T_1 \) and \( T_2 \) are disjoint.
- Distinct variables in the same tree must be substituted to distinct constants by \( \theta \).
- Every variable appearing in \( T_1 \) (resp., \( T_2 \)) is substituted to a constant symbol not appearing in \( T_1 \) or \( T_2 \) (because otherwise the cost of substituting a variable to a constant would be 0, which is not appropriate for measuring the distance between two mathematical expressions).

Then, we define the tree edit distance with variables as follows.

**Definition 1.** The tree edit distance with variables between \( T_1 \) and \( T_2 \) is

\[
dist(T_1, T_2) = \min_\theta dist_0(T_1\theta, T_2\theta).
\]
For example, consider trees $T_1$ and $T_2$ shown in Figure 3 and the unit cost model (i.e., $\gamma(x, y) = 1$ for any $x \neq y$). Then, $\text{dist}(T_1, T_2) = 5$ (in both ordered and unordered cases) by $\theta = \{(X, x), (Y, y), (Z, z), (W, w), (U, x), (V, y)\}$ and the following sequence of editing operations: change the label of node $w$ to $x$, insert node $h$, change the label of node $b$ to $f$, delete node $c$, and change the label of node $z$ to $g$, where we identify nodes by their labels.

As the basic property, the following holds.

**Proposition 2.** For both ordered and unordered cases, tree edit distance with variables satisfies the conditions of a distance measure.

**Proof.** Two trees that are isomorphic by one-to-one renaming of variables are regarded as identical. Clearly, $\text{dist}(T_1, T_2) = 0$ if and only if $T_1$ and $T_2$ are identical. Since $\text{dist}_0(T_1, T_2) = \text{dist}_0(T_2, T_1)$ holds for variable-free trees $T_1$ and $T_2$, $\text{dist}(T'_1, T'_2) = \text{dist}(T'_2, T'_1)$ holds for trees with variables $T'_1$ and $T'_2$. Let $\theta_{1,2} = \arg\min_\theta \text{dist}_0(T_1, T_2; \theta)$ and $\theta_{2,3} = \arg\min_\theta \text{dist}_0(T_2; \theta, T_3; \theta)$. We assume without loss of generality (w.l.o.g.) that $\theta_{1,2}$ and $\theta_{2,3}$ give the same substitutions for variables appearing in $T_2$. Let $\theta_{1,3}$ be the union of $\theta_{1,2}$ and $\theta_{2,3}$. Since $T'_1\theta_{1,2} = T_1\theta_{1,3}$, $T_2\theta_{1,2} = T_2\theta_{2,3}$, and $T_3\theta_{2,3} = T_3\theta_{1,3}$ hold, we have

\[
\text{dist}(T_1, T_3) \leq \text{dist}_0(T_1\theta_{1,3}, T_3\theta_{1,3}) \\
\leq \text{dist}_0(T_1\theta_{1,2}, T_2\theta_{1,2}) + \text{dist}_0(T_2\theta_{2,3}, T_3\theta_{2,3}) \\
= \text{dist}(T_1, T_2) + \text{dist}(T_2, T_3).
\]

There is a close relationship between the tree edit distance and the tree mapping [6]. For an unordered tree, a bijective mapping $\mathcal{M} \subseteq V(T_1) \times V(T_2)$ is called a tree mapping if for every $(u_1, v_1), (u_2, v_2) \in \mathcal{M}$, it holds that: (i) $u_1 = u_2$ if and only if $v_1 = v_2$; and (ii) $u_1$ is an ancestor of $u_2$ if and only if $v_1$ is an ancestor of $v_2$. Condition (i) states that each node appears at most once in $\mathcal{M}$, and condition (ii) states that ancestor-descendant relations must be preserved in $\mathcal{M}$. For ordered trees, the following condition is needed in addition to (i) and (ii): (iii) $u_1$ is left to $u_2$ if and only if $v_1$ is left to $v_2$. See [6] for the precise definition of “left”. It is known that any edit sequence can be modified without changing the total cost such that change-label operations follow deletion operations and insertion operations follow change-label operations. Then, an edit sequence gives a tree mapping: the nodes not deleted or inserted correspond to each other. Conversely, a tree mapping gives an edit sequence: nodes in $T_1$ (resp., $T_2$) that do not appear in $\mathcal{M}$ are regarded as deleted (resp., inserted), and any $(u, v) \in \mathcal{M}$ is regarded as change-labeled if their labels are different. Therefore, we use such words as “$u$ is mapped to $v$” when discussing about tree edit distance.
3 Ordered Trees

In this section, all trees are ordered trees, which means that the children of each node are ordered from left to right and that this ordering must be preserved among isomorphic trees. For each tree $T$, $V(T)$ and $E(T)$ denote the sets of nodes and edges, respectively. We let $n_1 = |V(T_1)|$ and $n_2 = |V(T_2)|$. For each node (resp., vertex) $v$ in a tree (resp., in a graph), $\ell(v)$ denotes the label of $v$.\footnote{We may use the label of a node to denote the node itself when there is no confusion.} We may use the label of a node to denote the node itself when there is no confusion.

\begin{proposition}
For ordered trees, whether or not $\text{dist}(T_1, T_2) = 0$ can be determined in polynomial time.
\end{proposition}

\textbf{Proof.} We construct an Euler string $\text{str}(T_i)$ [2] for each of the trees $T_i$ using depth first search (DFS). In constructing $\text{str}(T_i)$, we assign a unique integer number from 1, 2, \ldots as the label of a variable node every when we first encounter the variable. Then, it is straightforward to see $\text{str}(T_1) = \text{str}(T_2)$ if and only if $\text{dist}(T_1, T_2) = 0$.\hfill \blacktriangleleft

\begin{theorem}
For ordered trees, the tree edit distance problem with variables is NP-complete.
\end{theorem}

\textbf{Proof.} It is clear that the problem is in NP. Then, we show a polynomial-time reduction from the maximum clique problem (see also Figure 4). The maximum clique problem is, given an undirected graph $G(V, E)$ and an integer $k$, to decide whether or not there exists a complete subgraph (clique) of size ($\#$vertices) $k$ in $G(V, E)$, where all vertices have the same label. It is well-known that the problem is NP-complete.

From a given $k$, we construct $T_1$ as follows:

\begin{align*}
V(T_1) &= \{r_1\} \cup \{v_1, \ldots, v_k\} \cup \left( \bigcup_{i \in \{1, \ldots, k\}} \{v_{i,1}, \ldots, v_{i,k}\} \right), \\
E(T_1) &= \left( \bigcup_{i \in \{1, \ldots, k\}} \{(r_1, v_i)\} \right) \cup \left( \bigcup_{i \in \{1, \ldots, k\}} \{(v_i, v_{i,1}), \ldots, (v_i, v_{i,k})\} \right),
\end{align*}

\footnote{We mainly use “nodes” for trees and “vertices” for graphs.}
\[ \ell(r_1) = a, \]
\[ \ell(v_1) = \ell(v_2) = \ldots = \ell(v_k) = b, \]
\[ \ell(v_{i,j}) = X_{i,j} \text{ for all } i, \]
\[ \ell(v_{i,j}) = \ell(v_{j,i}) = X_{i,j} \text{ for all } i < j, \]

where \( X_{i,j} \neq X_{i',j'} \) for any \( i \neq i' \) or \( j \neq j' \).

From a given \( G(V,E) \) with \( V = \{w_1, \ldots, w_n\} \), we construct \( T_2 \) as follows:

\[ V(T_2) = \{r_2\} \cup \{u_1, \ldots, u_n\} \cup \left( \bigcup_{i \in \{1, \ldots, n\}} \{u_{i,1}, \ldots, u_{i,n}\} \right), \]
\[ E(T_2) = \left( \bigcup_{i \in \{1, \ldots, n\}} \{(r_2, u_i)\} \right) \cup \left( \bigcup_{i \in \{1, \ldots, n\}} \{(u_i, u_{i,1}), \ldots, (u_i, u_{i,n})\} \right), \]
\[ \ell(r_2) = a, \]
\[ \ell(u_1) = \ldots = \ell(u_n) = b, \]
\[ \ell(u_{i,j}) = \ell(u_{j,i}) = Y_{i,j} \text{ for all } \{w_i, w_j\} \in E \text{ with } i < j, \]
\[ \ell(u_{i,j}) = Y_{i,j} \text{ for all other } u_{i,j}. \]

where \( Y_{i,j} \neq Y_{i',j'} \) holds for any \( i \neq i' \) or \( j \neq j' \).

Here, we note that \( n_1 = 1 + k + k^2 \) and \( n_2 = 1 + n + n^2 \). We show that \( G(V,E) \) has a clique of size \( k \) if and only if \( \text{dist}(T_1, T_2) = n_2 - n_1 \) (i.e., \( T_1 \) is obtained by deletion operations from \( T_2 \) and renaming of variables). We say that a tree mapping \( \mathcal{M} \) is an inclusion mapping if \( \mathcal{M} \) corresponds to the sequence of edit operations with cost \( n_2 - n_1 \) (i.e., \( \mathcal{M} \) contains all nodes in \( T_1 \)).

Suppose that \( G(V,E) \) has a clique of size \( k \). We assume w.l.o.g. that \( \{w_1, w_2, \ldots, w_k\} \) be the set of nodes in that clique. Then, the following mapping gives an inclusion mapping from \( T_1 \) to \( T_2 \):

\[ \mathcal{M} = \{(r_1, r_2)\} \cup \{(v_i, u_i) \mid i = 1, \ldots, k\} \cup \{(X_{i,j}, Y_{i,j}) \mid 1 \leq i \leq j \leq k\}. \]

Conversely, suppose that there exists an inclusion mapping \( \mathcal{M} \) from \( T_1 \) to \( T_2 \). We assume w.l.o.g. that \( \mathcal{M} \) includes the following mappings:

\[ \{(r_1, r_2)\} \cup \{(v_i, u_i) \mid i = 1, \ldots, k\} \]

Then, for any \( (i,j) \) such that \( 1 \leq i < j \leq k \), \( X_{i,j} \) must be mapped to \( Y_{i,j} \) because \( v_i \) is mapped to \( u_i \) and \( v_j \) is mapped to \( u_j \), and \( X_{i,j} \) (resp., \( Y_{i,j} \)) is only one variable appearing in children of both \( v_i \) and \( v_j \) (resp., \( u_i \) and \( u_j \)). It means that for all \( (i,j) \) such that \( 1 \leq i < j \leq k \), there exists an edge between \( w_i \) and \( w_j \). Therefore, there exists a clique of size \( k \) in \( G(V,E) \). Note that although \( X_{i,j} \) may not be necessarily mapped to \( Y_{j,i} \), it does not cause a problem.

Finally, we consider the bounded degree case. In this case, it is enough to encode each non-leaf node as in Figure 5. Let \( \hat{T} \) be the tree obtained from \( T \) by this encoding. Then, it is straightforward to see that there exists an inclusion mapping from \( T_1 \) to \( T_2 \) if and only if there exists an inclusion mapping from \( \hat{T}_1 \) to \( \hat{T}_2 \).

**Proposition 5.** The tree edit distance problem with variables can be solved in polynomial time for ordered trees if each variable occurs once in input trees.
Proof. Let $F_i$ denote an ordered forest (i.e., an ordered set of rooted trees). For each $F_i$, $V(F_i)$ denotes the set of nodes in $F_i$. For the root $v$ of the rightmost tree in $F_i$, $F_i - T_i(v)$ denotes the forest obtained by removing the rightmost tree of $F_i$, and $F_i - v$ denotes the forest obtained by removing $v$ (i.e., each child $u$ of $v$ becomes the root of the subtree induced by $u$ and its descendants).

Recall that the edit distance for ordered trees (without variables) $\text{dist}_0(T_1, T_2)$ can be computed in $O(n_1^2 n_2^2)$ time by using the following dynamic programming algorithm [6, 18]:

$$D_0(F_1, \epsilon) = \sum_{u \in V(F_1)} \gamma(\ell(u), \epsilon),$$

$$D_0(\epsilon, F_2) = \sum_{v \in V(F_2)} \gamma(\epsilon, \ell(v)),$$

$$D_0(F_1, F_2) = \min \left\{ D_0(F_1 - u, F_2) + \gamma(\ell(u), \epsilon),
\quad D_0(F_1, F_2 - v) + \gamma(\epsilon, \ell(v)),
\quad D_0(F_1 - T_1(u), F_2 - T_2(v)) + D_0(T_1(u) - u, T_2(v) - v) + \gamma(\ell(u), \ell(v)) \right\},$$

where $\epsilon$ in $D_0(F_1, \epsilon)$ and $D_0(\epsilon, F_2)$ denotes the empty forest, $u$ and $v$ in the third recursion are the roots of the rightmost trees in $F_1$ and $F_2$, respectively, and $D_0(T_1, T_2)$ gives $\text{dist}_0(T_1, T_2)$. Then, it is enough to redefine $\gamma(x, y)$ function as follows:

$$\gamma(X_i, X_j) = 0, \quad \text{for any variable pair } (X_i, X_j) \text{ such that } X_i \neq X_j,$$

$$\gamma(a, a) = 0, \quad \text{for any constant symbol } a,$$

$$\gamma(x, y) = 1, \quad \text{for any other pair } (x, y).$$

Note that $\gamma(X_i, X_j) = 0$ always holds in this dynamic programming algorithm because $X_i$ and $X_j$ always appear in $F_1$ and $F_2$, respectively. Then, it is straightforward to see that this algorithm correctly computes the edit distance with variables and works in polynomial time.

Let $DP_{SO}(T_1, T_2)$ denote the algorithm for two input trees $T_1$ and $T_2$ given in the proof. The above result and proof are very similar to those in Theorem 11 of [3]. However, each variable can match a subtree in [3], whereas each variable can match a variable or constant here. Hereafter, $M$ denotes the total number of occurrences of variables, and $O^{**}(f(\cdots))$ denotes $O(f(\cdots) \cdot \text{poly}(n_1, n_2))$ time, where $\text{poly}(n_1, n_2)$ denotes some polynomial function of $n_1$ and $n_2$.  

![Figure 5](image-url) Encoding of the root node, where other non-leaf nodes are encoded in the same way except that label $a$ is replaced with label $b.$
Proposition 6. The tree edit distance problem with variables for ordered trees can be solved in $O^*(2^M)$ time.

Proof. Let $\mathcal{X} = (X^1, X^2, \ldots, X^{m_1})$ and $\mathcal{Y} = (Y^1, Y^2, \ldots, Y^{m_2})$ be the lists of occurrences of variables in the DFS ordering on $T_1$ and $T_2$, respectively, where $M = m_1 + m_2$. We examine all 0-1 assignments $\sigma$ on $\mathcal{X}$ and $\mathcal{Y}$, where 1 (resp., 0) means that the corresponding variable is mapped (resp., is not mapped) to a variable in the other tree. Let $\phi(X^i) = \{j \mid j \leq i, \sigma(X^j) = 1\}$ and $\phi(Y^i) = \{j \mid j \leq i, \sigma(Y^j) = 1\}$. Since any edit operation does not change the ordering, $X^i$ is mapped to $Y^j$ such that $\phi(X^i) = \phi(Y^j)$. In some cases, $X_i$ is mapped to multiple variables (e.g., $Y_j$ and $Y_k$). We ignore such an assignment $\sigma$ because of the constraint on $\theta$. Then, each $\sigma$ gives a matching (i.e., partial one-to-one mapping) between $\mathcal{X}$ and $\mathcal{Y}$, where $\mathcal{X}$ (resp., $\mathcal{Y}$) denotes the set of variables appearing in $\mathcal{X}$ (resp., $\mathcal{Y}$).

Then, we assign a unique constant symbol to each variable in $\mathcal{X}$. We assign the same symbol (e.g., $b_k$) as $X_i$ to $Y_j$ if $X_i$ is mapped to $Y_j$. If a variable $X_i$ (resp., $Y_j$) is not mapped to a variable, we assign a unique constant symbol to the variable (e.g., $c_k$ for $X_i$, and $d_h$ for $Y_j$).

Let $\theta_\sigma$ denote the resulting substitution.

For example, let $\mathcal{X} = (X_1, X_2, X_3, X_2, X_4, X_5)$, $\mathcal{Y} = (Y_1, Y_2, Y_3, Y_4, Y_5)$. For $\sigma(\mathcal{X}) = (1, 0, 1, 1, 0)$ and $\sigma(\mathcal{Y}) = (1, 1, 0, 1, 0)$, we have a mapping of $\sigma(X_1), \sigma(X_2), \sigma(X_3), \sigma(X_4), \sigma(X_5)$, respectively, where $\mathcal{X}$, $\mathcal{Y}$, $\theta_\sigma$ give a matching of $\mathcal{X}$ and $\mathcal{Y}$.

Then, we have a substitution $\theta_\sigma$ such that

\[
\mathcal{X}\theta_\sigma = \{ (b_1, b_2, b_3, b_2, b_4, c_1) \}, \quad \mathcal{Y}\theta_\sigma = \{ (b_1, b_3, b_2, b_4, d_1) \}.
\]

If $\sigma(\mathcal{X}) = (1, 0, 1, 0, 1)$ and $\sigma(\mathcal{Y}) = (1, 1, 1, 0, 1)$, we ignore this assignment because $X_2$ should be mapped to both $Y_2$ and $Y_3$.

By applying substitution $\theta_\sigma$ to $T_1$ and $T_2$, we obtain variable-free trees $T_1\theta_\sigma$ and $T_2\theta_\sigma$. For each assignment $\sigma$, we compute $\text{dist}_0(T_1\theta_\sigma, T_2\theta_\sigma)$. Since all possible substitutions are examined by testing all $\sigma$, $\min_{\sigma} \text{dist}_0(T_1\theta_\sigma, T_2\theta_\sigma)$ gives $\text{dist}(T_1, T_2)$. Since $2^M$ assignments are examined and $\text{dist}_0(T_1\theta_\sigma, T_2\theta_\sigma)$ can be computed in polynomial time, the proposition holds.

In the above, we consider all 0-1 assignments to all occurrences of variables. However, it is enough to find a mapping between $X_i$s and $Y_j$s and thus we need not consider all 0-1 assignments. Based on this idea, we have the following theorem.

Theorem 7. The tree edit distance problem with variables for ordered trees can be solved in $O^*(\sqrt{3}^M)$ time.

Proof. As in the proof of Proposition 6, let $\mathcal{X}$ and $\mathcal{Y}$ be the lists of occurrences of variables in the DFS ordering for $T_1$ and $T_2$, respectively. For each variable $X_i$ occurring $h$ times in $\mathcal{X}$, we consider the following $2h - 1$ assignments: $(1, 0, 0, \cdots, 0, 0, 0)$, $(0, 1, 0, \cdots, 0, 0, 0)$, $(0, 0, 1, \cdots, 0, 0, 0)$, $\cdots$, $(0, 0, \cdots, 0, 0, 1)$, and $(1, 1, 1, \cdots, 1, 1, 1)$, $(0, 1, 1, \cdots, 1, 1, 1)$, $(0, 0, 1, \cdots, 1, 1, 1)$, $\cdots$, $(0, 0, 0, \cdots, 0, 1, 1)$. The first $h$ cases mean that at most one occurrence of $X_i$ is mapped to some variable $Y_j$. In this case, $X_i$ is called a single occurrence variable, and the occurrences of $X_i$ corresponding to “0” are replaced by a unique constant (e.g., $c_k$) not appearing in the other parts whereas the occurrence of $X_i$ corresponding to “1” is kept as it is. The remaining $h - 1$ cases mean that the first “1” corresponds to the first occurrence of $X_i$ that is mapped to some $Y_j$ and that at least two occurrences of $X_i$ are mapped to the same number of occurrences of $Y_j$. In this case, $X_i$ is called a multi occurrence variable, and a unique constant (e.g., $b_h$) is shared by $X_i$ and $Y_j$. For each multi occurrence variable, only the position of the first “1” is relevant. For example, suppose that $X_i$ is the
Tree Edit Distance with Variables

multi occurrence variable to which “1” is assigned first in \( X \). Then, all occurrences of \( X_i \) are substituted by \( b_1 \). Suppose also that \( X_{i'} \) is the next multi occurrence variable to which “1” is assigned. Then, all occurrences of \( X_{i'} \) are substituted by \( b_2 \). \( Y_j \)'s are handled in an analogous way to \( X_i \)'s except that \( d_k \) is used in place of \( c_k \).

From 0-1 assignments on variables given as above, we obtain substituted sequences of \( X \) and \( Y \), which are denoted by \( \lambda(\chi) \) and \( \lambda(\gamma) \), respectively. For example, let

\[
\chi = (X_1, X_2, X_3, X_2, X_3, X_4, X_5, X_5, X_3, X_2, X_5),
\gamma = (Y_2, Y_1, Y_3, Y_4, Y_3, Y_4, Y_2, Y_5, Y_5, Y_1).
\]

Suppose that \((1), (0, 1, 1), (1, 1, 1), (1, 1), \) and \((0, 1)\) are assigned to \( X_1, X_2, X_3, X_4, \) and \( X_5 \), respectively. Furthermore, suppose that \((1, 0), (1, 1), (0, 1, 1), (1, 1, 1)\), and \((1)\) are assigned to \( Y_1, Y_2, Y_3, \) and \( Y_5 \), respectively. Then, we have

\[
\lambda(\chi) = (X_1, b_2, b_1, b_2, b_1, b_1, b_2, b_2, X_5),
\lambda(\gamma) = (b_1, Y_1, b_3, b_2, b_3, b_2, b_3, b_1, Y_5, b_2, d_1).
\]

Note that \( X_1 \) (and any other variable) can match another variable in \( \lambda(\gamma) \) with cost 0 and can match a constant symbol with cost 1. Note also that \( X_3 \)'s and \( Y_2 \)'s are substituted by \( b_1 \) because these are the multi occurrence variables to which “1” is assigned first in \( X \) and \( Y \), respectively. Note also that \( X_2 \)'s are substituted by \( b_2 \) because it is the multi occurrence variable that receives “1” after \( X_3 \) receives it.

Then, we consider the following procedure.

(i) \( d_{\text{min}} \leftarrow 0 \).
(ii) For all \( \lambda(\chi) \) and \( \lambda(\gamma) \), do step (iii).
(iii) \( d_{\text{min}} \leftarrow \min(d_{\text{min}}, DP_{SO}(\lambda(\chi), \lambda(\gamma))) \).
(iv) Output \( d_{\text{min}} \).

The correctness of this procedure follows from the following observation. Let \( M \) be the tree mapping corresponding to the minimum cost edit sequence. If \( X_i \) and \( Y_j \) match to each other at two or more position pairs (i.e., both are multi occurrence variables) in \( M \), then there must exist a \( \lambda \) such that the same constant \( b_k \) is assigned to \( X_i \) and \( Y_j \) because \( b_k \)'s are used only for multi occurrence variables. If \( X_i \) and \( Y_j \) match to each other at exactly one position pair in \( M \), both \( X_i \) and \( Y_j \) are treated as single occurrence variables and 1’s in the 0-1 assignments correspond to the matching position pair. The other occurrences of variables correspond to deletions, insertions, or change-labels because \( b_i, c_i, \) and \( d_i \) are constant symbols not appearing in the original input trees.

Here, we analyze the number of combinations of 0-1 assignments, which gives the exponential factor of the algorithm. Let \( \alpha_i M \) be the total number of occurrences of variables \( X_i \) and \( Y_j \) each of which occur \( l \) times. For example, \( \alpha_1 = \frac{3}{2} \), \( \alpha_2 = \frac{8}{2} \), \( \alpha_3 = \frac{12}{2} \), and \( \alpha_t = 0 \) for \( t \geq 4 \), for the above mentioned \( X \) and \( Y \). Note that \( \sum_{i=1}^{M} \alpha_i = 1 \) holds. For each variable occurring \( h \) times \((h = 1, \ldots, M)\), the number of examined 0-1 assignments is \( 2h - 1 \). Since \( 2h - 1 = 1 \) for \( h = 1 \), the total number of combinations of 0-1 assignments for \( X \) and \( Y \) is

\[
L_2(\alpha_2, \ldots, \alpha_M) = \prod_{h=2}^{M} (2h - 1)^{\alpha_h M}.
\]

Claim 8. \( f(h) = (2h - 1)^{\frac{h}{2}} \) is decreasing with respect to \( h = 2, 3, \ldots \).
Proof. It is seen by a simple numerical calculation that \((2 \cdot 2 - 1)^{\frac{1}{2}} > (2 \cdot 3 - 1)^{\frac{1}{2}}\). For \(h \geq 3\), by taking the derivative of \(\ln(f(h))\), we have
\[
\frac{d \ln(f(h))}{dh} = \frac{df}{dh} \frac{1}{f(h)} = -\frac{\ln(2h - 1)}{h^2} + \frac{2}{(2h - 1)h} < 0.
\]
Therefore, we have
\[
L_2(\alpha_2, \ldots, \alpha_M) = \prod_{h=2}^{M} (2h - 1)^{\alpha_h M} \leq \prod_{h=2}^{M} (2 \cdot 2 - 1)^{\alpha_h M} = (3)^{\left(\sum_{h=2}^{M} \alpha_h\right) M} \leq 3^M < \sqrt{3}^M.
\]
Since the other parts can be clearly done in polynomial time, the theorem holds.

\section{Unordered Trees}

In this section, all trees are unordered rooted trees. The graph isomorphism problem is, given two undirected graphs \(G_1(V_1, E_1)\) and \(G_2(V_2, E_2)\), to decide whether or not there exists a bijection \(\phi\) from \(V_1\) to \(V_2\) such that \(\{u, v\} \in E_1\) if and only if \(\{\phi(u), \phi(v)\} \in E_2\). It is unclear whether graph isomorphism is in \(P\) or \(NP\)-complete [5]. However, it is known that graph isomorphism can be solved in polynomial time if the maximum degree of input graphs is bounded by a constant [9, 13].

\begin{theorem}
For unordered trees, the problem of deciding \(dist(T_1, T_2) = 0\) is graph isomorphism complete. Furthermore, the problem can be solved in polynomial time if the maximum outdegree of \(T_1\) and \(T_2\) is bounded by a constant.
\end{theorem}

Proof. First, we show that graph isomorphism can be reduced to the problem in polynomial time. For each of \(G_1\) and \(G_2\), we construct trees as for \(T_2\) in the proof of Theorem 4. Then, it is straightforward to see that \(G_1\) and \(G_2\) are isomorphic if and only if \(dist(T_1, T_2) = 0\).

Next, we show that the problem can be reduced to graph isomorphism in polynomial time (see also Figure 6). Here, we consider w.l.o.g. graph isomorphism over labeled graphs (because it is obvious that labeled cases can be reduced to unlabeled cases in polynomial time). We show how to construct \(G_1(V_1, E_1)\) from \(T_1\), where an identical construction can be used for \(T_2\). We construct \(G_1(V_1, E_1)\) by adding vertices and edges to \(T_1\) as follows. For each variable \(X_i\), we create a new vertex \(v_X\) with constant label \(a\), connect \(v_X\) to all leaves in \(T_1\) having label \(X_i\), and change the labels of these leaves to \(b\), where \(a\) and \(b\) are constant symbols not appearing in \(T_1\) or \(T_2\) (we use the same \(a\) and \(b\) for all variables in \(T_1\) and \(T_2\)). Then, it is straightforward to see that \(G_1\) and \(G_2\) are isomorphic if and only if \(dist(T_1, T_2) = 0\).

Finally, we prove the last claim. We modify the reduction shown above (see also \(G'_1\) in Figure 6). For each variable \(X_i\), we make a copy \(\tilde{T}_1\) of \(T_1\) and then delete all of the following nodes in \(\tilde{T}_1\):
- a node which is not a node with label \(X_i\) or its ancestor,
- a node which is an ancestor of the lowest common ancestor of all nodes with label \(X_i\).

Then, we apply the deletion operation to the nodes in \(\tilde{T}_1\) each of which has a single child and change labels of all internal nodes to \(a\). Denote the resulting tree by \(T_{X_i}\). Finally, we identify leaves of \(T_{X_i}\) with the corresponding leaves in \(T_1\). Let \(G'_1\) be the graph obtained by applying this procedure to all variables. We construct \(G'_2\) in the same way. Clearly, this construction can be done in polynomial time. Furthermore, the maximum degree of the resulting graphs is
bounded by the maximum degree of the input trees (if the maximum outdegree of the input trees is no less than 2). Since the structure of each $T_X$ does not depend on the ordering of nodes, $G_1$ and $G_2'$ are isomorphic if and only if $\text{dist}(T_1, T_2) = 0$. Since isomorphism of graphs of bounded degree can be tested in polynomial time [9, 13], the last claim holds.

As in Section 3, let $M$ be the number of occurrences of variables in $T_1$ and $T_2$.

**Proposition 10.** $\text{dist}(T_1, T_2)$ can be computed in $O\left(\left(M \beta \alpha \alpha \right)^{\frac{1}{2} + \delta} M \cdot 1.26n_1 + n_2\right)$ time for unordered trees, where $\delta$ is any small positive constant.

**Proof.** Recall $\text{dist}(T_1, T_2) = \min_{\theta} \text{dist}_\theta(T_1, T_2\theta)$. Therefore, the problem can be solved by computing $\text{dist}_\theta(T_1, T_2\theta)$ for all essentially different $\theta$, where “essentially different” $\theta_1$ and $\theta_2$ mean that $\theta_1$ and $\theta_2$ give distinct correspondences between variables in $T_1$ and those in $T_2$. Let $h_1$ and $h_2$ be the numbers of variables in $T_1$ and $T_2$, respectively. Since we consider an upper bound, we assume w.l.o.g. that $h_1 = \alpha M$ and $h_2 = (1 - \alpha)M$, where $0 < \alpha < \frac{1}{2}$. The number of one-to-one mappings from the variables in $T_1$ to the variables in $T_2$ is bounded by

$$\frac{h_2!}{(h_2 - h_1)!} = \frac{(1 - \alpha)M!}{(1 - 2\alpha)M!}. \tag{1}$$

Note that some variable in $T_1$ may not be mapped to a variable in $T_2$ in some substitution $\theta$. However, the distance would not be decreased and thus such a substitution can be ignored. By using upper and lower bounds of Stirling’s approximation $\sqrt{2\pi n(\frac{n}{e})^n} \leq n! \leq e\sqrt{2\pi n(\frac{n}{e})^n}$, we have

$$\frac{(1 - \alpha)M!}{(1 - 2\alpha)M!} \leq \frac{e\sqrt{(1 - \alpha)M} \left(\frac{(1 - \alpha)M}{e}\right)^{(1 - \alpha)M}}{\sqrt{2\pi (1 - 2\alpha)M} \left(\frac{(1 - 2\alpha)M}{e}\right)^{(1 - 2\alpha)M}} = e^{\sqrt{(1 - \alpha)\left(\frac{(1 - \alpha)(1 - \alpha)}{2\pi(1 - 2\alpha)}\right) \left(\frac{(1 - \alpha)}{1 - 2\alpha}\right)^M \cdot \left(\frac{M}{e}\right)^{\alpha M}}$$

Since $\frac{(1 - \alpha)(1 - \alpha)}{(1 - 2\alpha)(1 - 2\alpha)} < 1.15$ holds for $0 < \alpha < \frac{1}{2}$ (using numerical calculations. Note that $\lim_{\alpha \to \frac{1}{2}} (1 - 2\alpha)^{(1 - 2\alpha)} = 1$), the above term is $O\left(1.15^M \cdot \left(\frac{M}{e}\right)^{\frac{\alpha M}{1 - 2\alpha}}\right)$ for a constant $\alpha$. Note that if $\alpha$ is very close to $\frac{1}{2}$, we need to consider a factor of $\sqrt{\frac{(1 - \alpha)}{2\pi(1 - 2\alpha)}}$ because $\alpha$ is not constant. In such a case, we use $((\frac{1}{2} + \epsilon)M)!$ to bound Eq.(1), where $\epsilon = \frac{1}{2} - \alpha$ and we can use

**Figure 6** Transformation from tree $T_1$ to graph $G_1$ of unbounded degree and graph $G_1'$ of bounded degree.
arbitrary small constant $\epsilon > 0$. This term is smaller than $O \left( \left( \frac{M}{2} \right)^{\left( \frac{1}{2} + \delta \right) M} \right)$ for $\delta > \epsilon$. Since $O \left( 1.15^M \cdot \left( \frac{M}{2} \right)^{\frac{3}{4}} \right) \leq O \left( \left( \frac{M}{2} \right)^{\left( \frac{1}{2} + \delta \right) M} \right)$ holds too, Eq. (1) is bounded by $O \left( \left( \frac{M}{2} \right)^{\left( \frac{1}{2} + \delta \right) M} \right)$ for any constant $\delta > 0$.

Since the tree edit distance between two unordered trees can be computed in $O(1.26^{n_1 + n_2})$ time [4], the proposition holds.

In the above theorem, $M$ is defined as the number of occurrences of variables (in order to use the same parameter as in Theorem 7). However, $M$ can be defined as the total number of variables in $T_1$ and $T_2$ in this theorem because we only consider the number of variables in the proof.

### 5 Concluding Remarks

In this paper, we have introduced and studied the tree edit distance problem with variables. We showed that the problem (decision problem version) is NP-complete even for ordered trees, whereas it is well-known that edit distance for ordered tree can be computed in polynomial time. We presented parameterized and exponential-time algorithms for the ordered and unordered cases, respectively. Since these algorithms are not necessarily optimal, improvements of these algorithms are left as open problems. As for the formalization, the unit cost model is assumed mainly because defining an appropriate cost model via substitutions on variables is difficult. Giving such costs and developing the corresponding algorithms would benefit the future practical applications.

In this paper, we assumed that mathematical formulas are given as rooted trees. However, such formulas may be represented more efficiently by directed acyclic graphs (DAGs) with reusing identical sub-trees. Since it is not straightforward to extend the algorithms for the tree edit distance to those for the graph edit distance for DAGs [8], it would be interesting to study such extensions with variables.

### References


