Heavy Caterpillar Distances for Rooted Labeled Unordered Trees

Nozomi Abe, Takuya Yoshino and Kouich Hirata
Kyushu Institute of Technology, Kawazu 680-4, Iizuka 820-8502, Japan

Abstract: In this paper, we introduce two heavy caterpillar distances between rooted labeled unordered trees (trees, for short) based on the edit distance between the heavy caterpillars obtained from the heavy paths in trees. Then, we show that the heavy caterpillar distances provide the upper bound of the edit distance for trees, can be computed in quadratic time under the unit cost function and are incomparable with other variations of the edit distance.

1 INTRODUCTION

Comparing tree-structured data such as HTML and XML data for web mining or RNA and glycan data for bioinformatics is one of the important tasks for data mining. The most famous distance measure (Deza and Deza, 2016) between rooted labeled unordered trees (trees, for short) is the edit distance \( \tau_{\text{TAI}} \) (Tai, 1979). The edit distance is formulated as the minimum cost of edit operations, consisting of a substitution, a deletion and an insertion, applied to transform a tree to another tree. It is known that the edit distance is always a metric and coincides with the minimum cost of Tai mappings (Tai, 1979). Unfortunately, the problem of computing the edit distance between trees is MAX SNP-hard (Zhang and Jiang, 1994). This statement also holds even if trees are binary or the maximum height of trees is at most 3 (Akutsu et al., 2013; Hirata et al., 2011).

Many variations of the edit distance have developed as more structurally sensitive distances as the minimum cost of the variations of the Tai mapping (Jiang et al., 1995; Kan et al., 2014; Kuboyama, 2007; Lu et al., 2001; Wang and Zhang, 2001; Yamamoto et al., 2014; Yoshino and Hirata, 2017; Zhang, 1996). In particular, the alignment distance \( \tau_{\text{ALN}} \) (Jiang et al., 1995) and the segmental distance \( \tau_{\text{SEG}} \) (Kan et al., 2014) are the most general variations of \( \tau_{\text{TAI}} \), where \( \tau_{\text{ALN}} \) is incomparable with \( \tau_{\text{SEG}} \), and the isolated-subtree distance \( \tau_{\text{ILST}} \) (Wang and Zhang, 2001) (or constrained distance) (Zhang, 1996) is the most general tractable variation of \( \tau_{\text{TAI}} \) (Yoshino and Hirata, 2017).

A caterpillar (cf. (Gallian, 2007)) is a tree transformed to a path after removing all the leaves in it. Recently, Muraka et al. (Muraka et al., 2018) have shown that the problem of computing the edit distance between caterpillars is tractable and the structural restriction of caterpillars provides the limitation of the tractability for computing the edit distance. Also Muraka et al. (Muraka et al., 2019) have developed the method to fast approximate the edit distance between caterpillars.

Hence, in this paper, we introduce new distances for trees by using the edit distance between the embedded caterpillars. Then, we focus on the heavy path (Sleator and Tarjan, 1983), which is a famous embedded path in a tree obtained by selecting vertices whose number of descendants is largest from the root. In particular, Demaine et al. (Demaine et al., 2009) have adopted the heavy path to analyze the time complexity of computing the edit distance for rooted labeled ordered trees.

In this paper, first we formulate a heavy caterpillar in a tree as the caterpillar whose backbone is the heavy path in the tree and whose set of leaves consists of all the adjacent vertices to the heavy path in the tree. Then, we introduce the following two heavy caterpillar distances \( \tau_{\text{HC}} \) and \( \tau_{\text{HR}} \) between trees.

The heavy caterpillar distance \( \tau_{\text{HC}} \) is formulated as the sum of the edit distance between heavy caterpillars and the cost of deleting and inserting the remained vertices not contained in the heavy caterpillars. On the other hand, the heavy caterpillar distance \( \tau_{\text{HR}} \) is formulated as the sum of the edit distance between heavy caterpillars and the cost of the Tai map-
ping obtained by repeating recursively, after selecting vertices (as leaves in heavy caterpillars) to bridge the Tai mapping between the heavy caterpillars, to compute the edit distance (the Tai mapping) between the heavy caterpillars of the complete subtree rooted by the selected vertices.

Then, in this paper, we show that the heavy caterpillar distances \( \tau_{HC} \) and \( \tau_{RC} \) provide the upper bound of \( \tau_{TAI} \), that is, \( \tau_{TAI} \leq \tau_{HC} \leq \tau_{RC} \). For the maximum height \( h \) and the maximum number \( \lambda \) of leaves in given two trees, we can compute \( \tau_{HC} \) in \( O(h^2\lambda^2) \) time under the general cost function and in \( O(h^2\lambda) \) time under the unit cost function, and \( \tau_{RC}(T_1, T_2) \) in \( O(h^2\lambda^2) \) time under the general cost function and in \( O(h^2\lambda) \) time under the unit cost function. Furthermore, we show that \( \tau_{HC} \) and \( \tau_{RC} \) are incomparable with \( \tau_{LST} \), \( \tau_{ALN} \) and \( \tau_{SG} \). Hence, the heavy caterpillar distances \( \tau_{HC} \) and \( \tau_{RC} \) provide another tractable variations of the edit distance \( \tau_{TAI} \) incomparable with the isolated-subtree distance \( \tau_{LST} \).

2 PRELIMINARIES

A tree \( T \) is a connected graph \((V,E)\) without cycles, where \( V \) is the set of vertices and \( E \) is the set of edges. We denote \( V \) and \( E \) by \( V(T) \) and \( E(T) \). The size of \( T \) is \(|V|\) and denoted by \(|T|\). We sometime denote \( v \in V(T) \) by \( v \in T \). We denote an empty tree \((\emptyset, \emptyset)\) by \( \emptyset \). A rooted tree is a tree with one node \( r \) chosen as its root. We denote the root of a rooted tree \( T \) by \( r(T) \).

Let \( T \) be a rooted tree such that \( r = r(T) \) and \( u, v, w \in T \). We denote the unique path from \( r \) to \( v \), that is, the tree \((V', E')\) such that \( V' = \{v_1, \ldots, v_k\}, v_1 = r, v_k = v \) and \((v_i, v_{i+1}) \in E'\) for every \( i \) \((1 \leq i \leq k - 1)\), by \( U_P(v) \).

The parent of \( v \neq r \), which we denote by \( \text{par}(v) \), is its adjacent node on \( U_P(v) \) and the ancestors of \( v \neq r \) are the nodes on \( U_P(v) - \{v\} \). We denote the set of all ancestors of \( v \) by \( \text{ance}(v) \). We say that \( u \) is a child of \( v \) if \( v \) is the parent of \( u \) and \( u \) is a descendant of \( v \) if \( v \) is an ancestor of \( u \). We denote the set of children of \( v \) by \( ch(v) \) and that \( v \) is an ancestor of \( u \) by \( u \leq v \). We call a node with no children a leaf and denote the set of all the leaves in \( T \) by \( l(T) \).

A rooted path \( P \) is a rooted tree \( \{(v_1, v_2, \ldots, v_n) | 1 \leq i \leq n - 1 \} \) such that \( r(P) = v_1 \). We call the node \( v_n \) (the leaf of \( P \) ) an endpoint of \( P \) and denote it by \( e(P) \).

The degree of \( v \), denoted by \( d(v) \), is the number of children of \( v \), and the degree of \( T \), denoted by \( d(T) \), is \( \max \{d(v) | v \in T \} \). The height of \( v \), denoted by \( h(v) \), is \( \max \{|U_P(w)| | w \in l(T[v])\} \), and the height of \( T \), denoted by \( h(T) \), is \( \max \{h(v) | v \in T \} \).

We use the ancestor orders \( \leq \) and \( \preceq \), that is, \( u < v \) if \( v \) is an ancestor of \( u \) and \( u \leq v \) if \( u < v \) or \( u = v \). We say that \( w \) is the least common ancestor of \( u \) and \( v \), denoted by \( u \lor v \), if \( u \leq w \) and \( w \leq v \) and there exists no node \( w' \in T \) such that \( w' < w \) and \( w < w' \). Let \( T \) be a rooted tree \((V,E)\) and \( v \) a node in \( T \). A complete subtree of \( T \) at \( v \), denoted by \( T[v] \), is a rooted tree \((V', E')\) such that \( r(T') = v \), \( V' = \{u \in V | u \leq v \} \) and \( E' = \{(u, w) \in E | u, w \in V'\} \).

We say that \( u \) is to the left of \( v \) in \( T \) if \( \text{pre}(u) \leq \text{pre}(v) \) for the preorder number \( \text{pre} \) in \( T \) and \( \text{post}(u) \leq \text{post}(v) \) for the postorder number \( \text{post} \) in \( T \). We say that a rooted tree is ordered if a left-to-right order among siblings is given; unordered otherwise. We say that a rooted tree is labeled if each node is assigned a symbol from a fixed finite alphabet \( \Sigma \). For a node \( v \), we denote the label of \( v \) by \( l(v) \), and sometimes identify \( v \) with \( l(v) \). In this paper, we call a rooted labeled unordered tree a tree simply.

Furthermore, we call a set of trees a forest. In particular, we denote the forest obtained by deleting \( v \) in \( T[v] \) by \( T(v) \).

Definition 1 (Caterpillar (cf., Gallian, 2007)). We say that a tree is a caterpillar if it is transformed to a rooted path after removing all the leaves in it. For a caterpillar \( C \), we call the remained rooted path a backbone of \( C \) and denote it by \( bb(C) \).

It is obvious that \( r(C) = r(bb(C)) \) and \( V(C) = bb(C) \cup lv(C) \) for a caterpillar \( C \), that is, every node in a caterpillar is either a leaf or an element of the backbone.

Next, we introduce a tree edit distance and a Tai mapping.

Definition 2 (Edit operations (Tai, 1979)). The edit operations of a tree \( T \) are defined as follows, see Figure 1:

1. Substitution: Change the label of the node \( v \) in \( T \).
2. Deletion: Delete a node \( v \) in \( T \) with parent \( v' \), making the children of \( v \) become the children of \( v' \). The children are inserted in the place of \( v \) as a subset of the children of \( v' \). In particular, if \( v \) is the root in \( T \), then the result applying the deletion is a forest consisting of the children of the root.
3. Insertion: The complement of deletion. Insert a node \( v \) as a child of \( v' \) in \( T \) making \( v \) the parent of a subset of the children of \( v' \).

Let \( \varepsilon \not\in \Sigma \) denote a special blank symbol and define \( \Sigma_e = \Sigma \cup \{\varepsilon\} \). Then, we represent each edit operation by \((l_1 \mapsto l_2)\), where \((l_1, l_2) \in (\Sigma_e \times \Sigma_e - \{(\varepsilon, \varepsilon)\}) \). The operation is a substitution if \( l_1 \neq \varepsilon \) and \( l_2 \neq \varepsilon \), a deletion if \( l_2 = \varepsilon \), and an insertion if \( l_1 = \varepsilon \). For nodes \( v \) and \( w \), we also denote \((l(v) \mapsto l(w))\) by \( (v \mapsto w) \). We define a cost function \( \gamma : (\Sigma_e \times \Sigma_e \setminus \{(\varepsilon, \varepsilon)\}) \mapsto \mathbb{R}^+ \) on
pairs of labels. We often constrain a cost function $\gamma$ to be a metric; that is, $\gamma(l_1, l_2) \geq 0$, $\gamma(l_1, l_2) = 0$ iff $l_1 = l_2$, $\gamma(l_1, l_2) = \gamma(l_2, l_1)$ and $\gamma(l_1, l_2) \leq \gamma(l_1, l_3) + \gamma(l_2, l_3)$. In particular, we call the cost function that $\gamma(l_1, l_2) = 1$ if $l_1 \neq l_2$ a unit cost function.

**Definition 3** (Edit distance (Tai, 1979)). For a cost function $\gamma$, the cost of an edit operation $e = l_1 \mapsto l_2$ is given by $\gamma(e) = \gamma(l_1, l_2)$. The cost of a sequence $E = e_1, \ldots, e_k$ of edit operations is given by $\gamma(E) = \sum_{i=1}^{k} \gamma(e_i)$. Then, an *edit distance* $\tau_{TAI}(T_1, T_2)$ between trees $T_1$ and $T_2$ is defined as follows:

$$
\tau_{TAI}(T_1, T_2) = \min \left\{ \gamma(E) \mid \begin{array}{l}
E \text{ is a sequence of edit operations} \\
\text{transforming } T_1 \text{ to } T_2
\end{array} \right\}
$$

**Definition 4** (Tai mapping (Tai, 1979)). Let $T_1$ and $T_2$ be trees. We say that a triple $(M, T_1, T_2)$ is a *Tai mapping* (a mapping, for short) from $T_1$ to $T_2$ if $M \subseteq V(T_1) \times V(T_2)$ and every pair $(v_1, w_1)$ and $(v_2, w_2)$ in $M$ satisfies the following conditions.

1. $v_1 = v_2$ iff $w_1 = w_2$ (one-to-one condition).
2. $v_1 \leq v_2$ iff $w_1 \leq w_2$ (ancestor condition).

We will use $M$ instead of $(M, T_1, T_2)$ when there is no confusion denote it by $M \in \mathcal{M}_{TAI}(T_1, T_2)$.

Let $M$ be a mapping from $T_1$ to $T_2$. Let $I_M$ and $J_M$ be the sets of nodes in $T_1$ and $T_2$ but not in $M$, that is, $I_M = \{ v \in T_1 \mid (v, w) \notin M \}$ and $J_M = \{ w \in T_2 \mid (v, w) \notin M \}$. Then, the cost $\gamma(M)$ of $M$ is given as follows.

$$
\gamma(M) = \sum_{(v, w) \in M} \gamma(v, w) + \sum_{v \in I_M} \gamma(v, e) + \sum_{w \in J_M} \gamma(e, w).
$$

**Theorem 1** (Tai, 1979)). $\tau_{TAI}(T_1, T_2) = \min \{ \gamma(M) \mid M \in \mathcal{M}_{TAI}(T_1, T_2) \}$.

Furthermore, we introduce the variations of Tai mappings. Whereas the alignment distance (Jiang et al., 1995) has first defined by using an alignment tree between two trees as the common supertree, it is known that the alignment distance coincides with the minimum cost of less-constrained mappings (Kuboyama, 2007). Hence, in this paper, we regard the less-constrained mapping as an alignable mapping and formulate the alignment distance as the minimum cost of alignable mappings.

**Definition 5** (Variations of Tai mapping). Let $T_1$ and $T_2$ be trees and $M \in \mathcal{M}_{TAI}(T_1, T_2)$.

1. We say that $M$ is an *alignable mapping* (Kuboyama, 2007) (or an *less-constrained mapping* (Lu et al., 2001)), denoted by $M \in \mathcal{M}_{ALN}(T_1, T_2)$, if $M$ satisfies the following condition:

$$
\forall (v_1, w_1)(v_2, w_2)(v_3, w_3) \in M \\
\left( (v_1 \cup v_2 < v_1 \cup v_3) \implies (w_1 \cup w_2 < w_1 \cup w_3) \right)
$$

Also we define an *alignment distance* $\tau_{ALN}(T_1, T_2)$ (Jiang et al., 1995) as the minimum cost of all the alignable mappings, that is:

$$
\tau_{ALN}(T_1, T_2) = \min \{ \gamma(M) \mid M \in \mathcal{M}_{ALN}(T_1, T_2) \}.
$$

2. We say that $M$ is an *isolated-subtree mapping* (Wang and Zhang, 2001) (or a *constrained mapping* (Zhang, 1996)), denoted by $M \in \mathcal{M}_{ILST}(T_1, T_2)$, if $M$ satisfies the following condition:

$$
\forall (v_1, w_1)(v_2, w_2)(v_3, w_3) \in M \\
\left( (v_3 < v_1 \cup v_2) \iff (w_3 < w_1 \cup w_2) \right)
$$

Also we define an *isolated-subtree distance* $\tau_{ILST}(T_1, T_2)$ as the minimum cost of all the isolated-subtree mappings, that is:

$$
\tau_{ILST}(T_1, T_2) = \min \{ \gamma(M) \mid M \in \mathcal{M}_{ILST}(T_1, T_2) \}.
$$

3. We say that $M$ is a *segmental mapping* (Kan et al., 2014), denoted by $M \in \mathcal{M}_{SG}(T_1, T_2)$, if $M$ satisfies the following condition.

$$
\exists (v', w') \in M \\
\left( (v' \in \text{anc}(v)) \land (w' \in \text{anc}(w)) \right) \\
\implies \left( \text{par}(v), \text{par}(w) \right) \in M.
$$

Also we define a *segmental distance* $\tau_{SG}(T_1, T_2)$ as the minimum cost of all the segmental mappings, that is:

$$
\tau_{SG}(T_1, T_2) = \min \{ \gamma(M) \mid M \in \mathcal{M}_{SG}(T_1, T_2) \}.
$$

Furthermore, for distances $\tau_A$ and $\tau_B$, we say that $\tau_A$ is *incomparable* with $\tau_B$ if there exist trees $T_1$, $T_2$, $T_3$ and $T_4$ such that $\tau_A(T_1, T_2) < \tau_B(T_1, T_2)$ and $\tau_B(T_3, T_4) < \tau_A(T_3, T_4)$. 


Theorem 2 ((Kuboyama, 2007; Yoshino and Hirata, 2017)). Let $T_1$ and $T_2$ be trees. Then, it holds that $M\text{ILST}(T_1, T_2) \subseteq M\text{ALST}(T_1, T_2)$ and $M\text{SO}(T_1, T_2) \subseteq M\text{TAl}(T_1, T_2)$. On the other hand, $M\text{ILST}(T_1, T_2)$ or $M\text{ALST}(T_1, T_2)$ is incomparable with $M\text{SO}(T_1, T_2)$ with respect to set inclusion.

Theorem 2 implies $\tau_{\text{TAl}}(T_1, T_2) \leq \tau_{\text{ILST}}(T_1, T_2)$ and $\tau_{\text{TAl}}(T_1, T_2) \leq \tau_{\text{SO}}(T_1, T_2)$ for every tree $T_1$ and $T_2$. On the other hand, $\tau_{\text{ILST}}$ or $\tau_{\text{ALST}}$ is incomparable with $\tau_{\text{SO}}$. Furthermore, the following theorem is known for the problem of computing $\tau_{\text{TAl}}$ and its variations.

Theorem 3. Let $T_1$ and $T_2$ be trees such that $n = \max\{|T_1|, |T_2|\}$ and $d = \min\{d(T_1), d(T_2)\}$.

1. The problem of computing $\tau_{\text{TAl}}(T_1, T_2)$ is MAX SNP-hard (Zhang and Jiang, 1994). This statement holds even if both $T_1$ and $T_2$ are binary, the maximum height of $T_1$ and $T_2$ is at most 3 or the cost function is the unit cost function (Akutsu et al., 2013; Hirata et al., 2011).

2. The problem of computing $\tau_{\text{ALST}}(T_1, T_2)$ is MAX SNP-hard. On the other hand, if the degrees of $T_1$ and $T_2$ are bounded by some constants, then we can compute $\tau_{\text{ALST}}(T_1, T_2)$ in polynomial time with respect to $n$ (Jiang et al., 1995).

3. We can compute $\tau_{\text{ILST}}(T_1, T_2)$ in $O(n^2d)$ time (cf., (Yamamoto et al., 2014)).

4. The problem of computing $\tau_{\text{SO}}(T_1, T_2)$ is MAX SNP-hard. This statement holds even if both $T_1$ and $T_2$ are binary or the cost function is the unit cost function (Yamamoto et al., 2014).

In contrast to Theorem 3, Muraka et al. (Muraka et al., 2018) have recently shown the following theorem of the edit distance for caterpillars.

Theorem 4 ((Muraka et al., 2018)). Let $C_1$ and $C_2$ be caterpillars, $h = \max\{h(C_1), h(C_2)\}$ and $\lambda = \max\{h(C_1), h(C_2)\}$. Then, we can compute $\tau_{\text{TAl}}(C_1, C_2)$ in $O(h^2\lambda^3)$ time under the general cost function and $O(h^3\lambda^2)$ time under the unit cost function.

3 HEAVY CATERPILLAR DISTANCES

In this section, we introduce the heavy caterpillar in a tree, based on the heavy path (Sleator and Tarjan, 1983). Then, we formulate another variation of the edit distance as heavy caterpillar distances based on the edit distance for heavy caterpillars.

Definition 6 (Heavy path (Sleator and Tarjan, 1983)). Let $T$ be a tree. For $v \in T$ and $w \in ch(v)$, $w$ is a heavy child of $v$ if $|T[w]|$ is maximum and denote it by $hv(v)$. A heavy path of $T$ is the rooted path $\{v_1, \ldots, v_n\} \subseteq \{(v_i, v_{i+1}) \mid 1 \leq i \leq n - 1\}$ such that $v_1 = r(T)$, $v_{i+1} = hv(v_i)$ $(1 \leq i \leq n - 1)$ and $v_n \in hv(T)$.

If there exist more than two heavy children of $v$, then we may name one of them arbitrary a heavy child of $v$. Then, based on the heavy path in a tree, we introduce the heavy caterpillar in a tree as follows.

Definition 7 (Heavy caterpillar). Let $T$ be a tree and $P$ the heavy path of $T$. Then, we define the heavy caterpillar $hc(T) = (V, E)$ of $T$ as follows.

$$V = V(P) \cup \{w \in ch(v) \mid v \in V(P)\},$$

$$E = E(P) \cup \{(v, w) \mid v \in V(P), w \in ch(v)\}.$$

We denote the minimum cost Tai mapping between $C_1 = hc(T_1)$ and $C_2 = hc(T_2)$ by $M_{hc}(C_1, C_2)$. Then, the algorithm HVY\text{CAT}MAP in Algorithm 1 returns a Tai mapping based on the heavy caterpillars $C_1$ and $C_2$. We define the heavy caterpillar mapping between $T_1$ and $T_2$ as the mapping obtained from the algorithm HVY\text{CAT}MAP($T_1, T_2$) and denote it by $M_{hc}(T_1, T_2)$.

Algorithm 1: HVY\text{CAT}MAP.

1 procedure HVY\text{CAT}MAP($T_1, T_2$)

1.1 /* $T_1, T_2$: trees */

2 $C_1 \leftarrow hc(T_1); C_2 \leftarrow hc(T_2); L_1 \leftarrow lv(C_1); L_2 \leftarrow lv(C_2); M \leftarrow M_{hc}(C_1, C_2);$

3 $L \leftarrow \{(v, w) \in M \mid v \in L_1, w \in L_2, T_1(v) \neq 0, T_2(w) \neq 0\};$

4 foreach $(v, w) \in L$ do

5 $M_1 \leftarrow HVY\text{CAT}MAP(T_1[v], T_2[w]);$

6 $M \leftarrow M \cup M_1;$

7 return $M$;

Definition 8 (Heavy caterpillar distances). Let $T_i$ be a tree, $C_i = hc(T_i)$ and $D_i = T_i \setminus C_i$ $(i = 1, 2)$. Then, we define the heavy caterpillar distances $\tau_{hc}(T_1, T_2)$ and $\tau_{\text{hc}}(T_1, T_2)$ as follows.

$$\tau_{hc}(T_1, T_2) = \tau_{\text{TAl}}(C_1, C_2) + \sum_{v \in D_1} \gamma(v, e) + \sum_{w \in D_2} \gamma(e, w),$$

$$\tau_{\text{hc}}(T_1, T_2) = \gamma(M_{hc}(T_1, T_2)).$$

Theorem 5. For trees $T_1$ and $T_2$, it holds that $\tau_{\text{TAl}}(T_1, T_2) \leq \tau_{\text{hc}}(T_1, T_2) \leq \tau_{\text{hc}}(T_1, T_2)$.

Proof. For $C_i = hc(T_i)$ and $M' = M_{hc}(T_1, T_2) \setminus M_{hc}(C_1, C_2)$, since $\tau_{\text{TAl}}(C_1, C_2) = \gamma(M_{hc}(C_1, C_2))$, it holds that $\tau_{\text{hc}}(T_1, T_2) = \tau_{\text{TAl}}(C_1, C_2) + \gamma(M')$. If $M' = \emptyset$, then it holds that $\gamma(M') = \sum_{v \in D_1} \gamma(v, e) + \sum_{w \in D_2} \gamma(e, w)$, which implies that $\tau_{\text{hc}}(T_1, T_2) \leq \tau_{hc}(T_1, T_2)$.
In order to show that $\tau_{\text{TAI}}(T_1, T_2) \leq \hat{\tau}_{\text{HIC}}(T_1, T_2)$, it is sufficient to show that the heavy caterpillar map\-ping $M_{\text{hc}}(T_1, T_2)$ is a Tai mapping. If it is true, then it holds that $\tau_{\text{TAI}}(T_1, T_2) \leq \gamma(M_{\text{hc}}(T_1, T_2))$.

Let $L' = \{(v_1, w_1), \ldots, (v_k, w_k)\}$ be the union of all the $L$ selected at line 2 in HVYCATMAP in Algorithm 1 recursively, where $v_0 = r(T_1)$ and $w_0 = r(T_2)$. Also let $M_i$ be the output of HVYCATMAP($T_i[v_i], T_2[w_i]$) $(0 \leq i \leq k)$ and $M = M_0 \cup M_1 \cup \cdots \cup M_k$, where $M_0 = M_{\text{hc}}(C_1, C_2) \in M_{\text{TAI}}(T_1, T_2)$. Note that $M_i \in M_{\text{TAI}}(T_i[v_i], T_2[w_i])$, so $M_i \in M_{\text{TAI}}(T_1, T_2)$.

Since $M_i$ is mutually distinct for every $i$ and $M_i \in M_{\text{TAI}}(T_i[v_i], T_2[w_i])$, $M$ satisfies the one-to-one condition. By the construction of $L$, $(M \setminus M_i) \cup \{(v_i, w_i)\}$ satisfies the ancestor condition for every $(v_i, w_i) \in L'$, which implies that $M$ satisfies the ancestor condition. Hence, it holds that $M \in M_{\text{TAI}}(T_1, T_2)$. Since $M = M_{\text{hc}}(T_1, T_2)$, it holds that $M_{\text{hc}}(T_1, T_2) \in M_{\text{TAI}}(T_1, T_2)$.

**Theorem 6.** Let $T_1$ and $T_2$ be trees, where $h = \max\{h(T_1), h(T_2)\}$ and $\lambda = \max\{|l(T_1)|, |l(T_2)|\}$. Then, we can compute $\tau_{\text{HIC}}(T_1, T_2)$ in $O(h^2\lambda^2)$ time under the general cost function and in $O(h^2\lambda)$ time under the unit cost function. Also we can compute $\tau_{\text{HIC}}(T_1, T_2)$ in $O(h^2\lambda^2)$ time under the general cost function and in $O(h^2\lambda^2)$ time under the unit cost function.

**Proof.** Let $G = h(T_i)$ ($i = 1, 2$). First, we can obtain $C_i$ in $O(|T_i|) = O(h\lambda)$ time (Sleator and Tarjan, 1983). Since it is essential for computing $\tau_{\text{HIC}}(T_1, T_2)$ to compute $\tau_{\text{HIC}}(C_1, C_2)$, the time complexity of computing $\tau_{\text{HIC}}$ follows from Theorem 4.

Next, consider the number of recursive calls in HVYCATMAP in Algorithm 1. For $L'$ in the proof of Theorem 5, we denote $L_1 = \{v \in V(T_1) \mid (v, w) \in L'\}$ and $L_2 = \{w \in V(T_2) \mid (v, w) \in L'\}$. Then, for every leaf $u \in l(T_1) \setminus l(C_1)$ (resp., $u \in l(T_2) \setminus l(C_2)$), there exists exactly one $v \in L_1$ (resp., $w \in L_2$) such that $T_1[v]$ (resp., $T_2[w]$) called as HVYCATMAP($T_1[v], T_2[w]$) at line 4 in Algorithm 1 contains $u$. This statement implies that $|L'| \leq \lambda$. Hence, the number of recursive calls is at most $\lambda$, so the statement of computing $\tau_{\text{HIC}}$ holds.

In the remainder of this section, we assume that the cost function is the unit cost function. Then, we compare $\tau_{\text{HIC}}$ with the edit distance $\tau_{\text{TAI}}$ and its other variations $\tau_{\text{ALN}}, \tau_{\text{LST}}$ and $\tau_{\text{ALN}}$.

**Lemma 1.** There exist trees $T_1$ and $T_2$ such that $|T_1| = |T_2| = O(n)$, $\tau_{\text{TAI}}(T_1, T_2) = O(1)$ but $\tau_{\text{HIC}}(T_1, T_2) = \Omega(n)$.

**Proof.** Consider $T_1$ and $T_2$ illustrated in Figure 2. It is obvious that $|T_1| = |T_2| = 2n + 1$. Also it holds that $\tau_{\text{TAI}}(T_1, T_2) = 2$ because $M_1$ in Figure 2 is the minimum cost mapping for $\tau_{\text{TAI}}$. Note that $\tau_{\text{LST}}(T_1, T_2) = \tau_{\text{ALN}}(T_1, T_2) = \tau_{\text{SO}}(T_1, T_2) = 2$.

On the other hand, by the definition of $\tau_{\text{HIC}}$, we construct the mapping with cost 0 between $hc(T_1)$ and $hc(T_2)$, that is, the second child of the root in $T_1$ (labeled by $a$) is corresponding to the third child of the root in $T_2$ (labeled by $a$) and the third child of the root in $T_1$ (labeled by $b$) is to the second child of the root in $T_2$ (labeled by $b$). Then, $M_2$ in Figure 2 is the minimum cost mapping for $\tau_{\text{HIC}}$. Hence, it holds that $\tau_{\text{HIC}}(T_1, T_2) = 2n - 4$. □

![Figure 2: Trees $T_1$ and $T_2$ in Lemma 1 and the minimum cost mappings $M_1$ for $\tau_{\text{TAI}}$ and $M_2$ for $\tau_{\text{HIC}}$.](image)

**Lemma 2.** There exist trees $T_1$ and $T_2$ such that $|T_1| = |T_2| = O(n)$, $\tau_{\text{TAI}}(T_1, T_2) = \tau_{\text{HIC}}(T_1, T_2) = O(1)$ but $\tau_{\text{LST}}(T_1, T_2) = \tau_{\text{ALN}}(T_1, T_2) = \tau_{\text{SO}}(T_1, T_2) = \Omega(n)$.

**Proof.** Consider $T_1$ and $T_2$ illustrated in Figure 3. It is obvious that $|T_1| = |T_2| = 2n + 1$. Since $T_1$ and $T_2$ are caterpillars, it holds that $\tau_{\text{TAI}}(T_1, T_2) = \tau_{\text{HIC}}(T_1, T_2) = \tau_{\text{LST}}(T_1, T_2) = \tau_{\text{ALN}}(T_1, T_2) = \tau_{\text{SO}}(T_1, T_2) = 2$.

On the other hand, the minimum cost isolated-subtree mapping maps $r_1 = r(T_1)$ to $r_2 = r(T_2)$, $n + 1$ children of $r_1$ to $n + 1$ children of $r_2$, so the number of the remained (non-mapped) vertices is $n - 1 + n = 2n - 1$. Hence, it holds that $\tau_{\text{LST}}(T_1, T_2) = 2n - 1$. □
Lemma 3. There exist trees $T_1$ and $T_2$ such that $|T_1| = |T_2| = O(n)$, $\tau_{TA}(T_1, T_2) = \tau_{HC}(T_1, T_2) = O(1)$ but $\tau_{ALN}(T_1, T_2) = \Omega(n)$.

Proof. Consider trees $T_1$ and $T_2$ in Figure 4. It is obvious that $|T_1| = |T_2| = 2n + 2$. Since $T_1$ is transformed to $T_2$ by inserting a vertex labeled $d$ in $T_1$ after deleting a vertex labeled by $d$ in $T_1$, it holds that $\tau_{TA}(T_1, T_2) = 2$. Since $T_1$ and $T_2$ are caterpillars, it also holds that $\tau_{HC}(T_1, T_2) = \tau_{HC}(T_1, T_2) = 2$. Note that $\tau_{SG}(T_1, T_2) = 4$.

On the other hand, the minimum cost alignable mapping maps a vertex labeled by $b$ (resp., $c$) in $T_1$ to a vertex labeled by $c$ (resp., $b$) in $T_2$ injectively. Then, it holds that $\tau_{ALN}(T_1, T_2) = 2n$. Also it holds that $\tau_{LST}(T_1, T_2) = 2n$. \hfill \Box

Lemma 4. There exist trees $T_1$ and $T_2$ such that $|T_1| = |T_2| = O(n)$, $\tau_{TA}(T_1, T_2) = \tau_{HC}(T_1, T_2) = O(1)$ but $\tau_{SG}(T_1, T_2) = \Omega(n)$.

Proof. Consider $T_1$ and $T_2$ illustrated in Figure 5 (cf., (Kan et al., 2014)) and let $C_i = hc(T_i)$ ($i = 1, 2$). It is obvious that $|T_1| = 4n$ and $|T_2| = 4n - 2$. Also it holds that $\tau_{TA}(T_1, T_2) = 2$.

For $C_1$ and $C_2$ in Figure 5, it holds that $\tau_{HC}(T_1, T_2) = \tau_{HC}(C_1, C_2) + 2n = 2n + 2$. Since the minimum cost mapping for $\tau_{HC}(C_1, C_2)$ maps the rightmost vertex $v$ in $C_1$ to the rightmost vertex $w$ in $C_2$, $hc(T_1, T_2)$ maps the children of $v$ in $T_1$ to the children of $w$ in $T_2$ injectively. Hence, it holds that $\tau_{HC}(T_1, T_2) = \tau_{HC}(C_1, C_2) = 2$. Note that $\tau_{LST}(T_1, T_2) = \tau_{ALN}(T_1, T_2) = 2$.

On the other hand, since the minimum cost segmental mapping maps to the path with $n - 1$ vertices and its $n$ children and the vertex and its $n$ children in $T_2$, the number of remained (i.e., non-mapped) vertices is $n + 1$ in $T_1$ and $n - 1$ in $T_2$, so it holds that $\tau_{SG}(T_1, T_2) = 2n$. \hfill \Box

4 CONCLUSION

In this paper, we have introduced heavy the caterpillar distances $\tau_{HC}$ and $\tau_{HC}$ and shown that they provide the upper bound of the edit distance $\tau_{TA}$, they are tractable, in particular, quadratic-time computable under the unit cost function, and incomparable with other variations of $\tau_{TA}$ presented by (Yoshino and Hirata, 2017). Since $\tau_{LST}$ is the most general tractable variation of $\tau_{TA}$ (Yoshino and Hirata, 2017), $\tau_{HC}$ and $\tau_{HC}$ are another tractable variations of $\tau_{TA}$ incomparable with $\tau_{LST}$.

Concerned with Lemma 1, it is possible to avoid this problem to compute the edit distance (the Tai mapping) between heavy caterpillars by considering the occurrences of labels in the descendants. It is a future work whether or not we can design a new method to avoid to this problem.

The heavy caterpillar distances $\tau_{HC}$ and $\tau_{HC}$ are defined by $M_{hc}(C_1, C_2)$ and $M_{hc}(T_1, T_2)$ as opera-
tional, whereas other variations of $\tau_{T_AI}$ are based on the declarative definition of the Tai mapping. Then, it is a future work whether or not to give the declarative definition of $\tau_{HC}$ and $\hat{\tau}_{HC}$.

In general, we cannot determine the heavy path and then the heavy caterpillar uniquely. Then, it is a future work to design the method to select the heavy path and the heavy caterpillar uniquely appropriate to $\tau_{HC}$ and $\hat{\tau}_{HC}$.

Finally, after improving that the heavy caterpillar distances $\tau_{HC}$ and $\hat{\tau}_{HC}$ are determined uniquely, it is an important future work to give experimental results to compare $\tau_{HC}$ and $\hat{\tau}_{HC}$ with the isolated-subtree distance $\tau_{ILST}$ for real data.

ACKNOWLEDGMENTS

This work is partially supported by Grant-in-Aid for Scientific Research 17H00762, 16H02870 and 16H01743 from the Ministry of Education, Culture, Sports, Science and Technology, Japan. The authors would like to thank anonymous referees of ICPRAM’20 for valuable comments to revise the submitted version of this paper.

REFERENCES


