Local Similarity between Semi-Ordered Trees by Finding the Constrained Mapping

Tomoya Yamzaki[1], Keisuke Otaki[1], Madori Ikeda[1], Ryo Yoshinaka[1], Akihiro Yamamoto[1] and Tetsuji Kuboyama[2]

[1] Graduate School of Informatics, Kyoto University
[2] Computer Centre, Gakushuin University

Abstract. We propose an algorithm based on a dynamic programming algorithm for finding local structures with the highest similarity within two semi-ordered trees. Finding such local structures is useful, because trees treated in many fields often share only limited regions. For example in botany, trees representing plant architectures often share only local structures and such local structures have an important property. A semi-ordered tree is a tree with a total semi-order relation defined on the set of children of each vertex and is used as the topological structure of such plant architectures. The proposed method for evaluating the similarity between two semi-ordered trees is based on the algorithm for computing the tree edit distance between them. Our method is to evaluate the local similarity by selecting the constrained mapping, and additionally to extract the local structure with the highest similarity, that is, the maximum score.

1 Introduction

Recently, the comparison of trees has had important roles in many fields, such as biology and botany. The goal of our research is to give an efficient method for comparisons of two trees that is useful for such fields. The class of trees varies in accordance with the fields, and the similarity between trees, which should be the foundation of comparing trees, depends on the class. *A semi-ordered tree* is a tree with a total semi-order relation defined on the set of children of each vertex. A semi-ordered tree is used as a model of a topological structure of a plant architecture and a self-assembly of a plant because the semi-order relations are measured between the components of a plant [7].

Similarities between two trees are categorized into two types, *global* and *local*. An example of the global similarity is based on the *edit distance*, which is defined by Smith and Waterman [8] for comparing two strings. Following their works, edit distances between ordered, unordered, and semi-ordered trees are defined by Zhang and Jiang [12, 13] and Ouangraoua and Ferraro [7]. In each class of trees, the edit distance between two trees is the minimum sum of the costs of the edit operations (deleting, inserting, and relabeling of a labeled node) to convert a tree to another one. It is also defined in various manners based on a mapping which is a set of pairs of vertices with a one-to-one relation between two trees, because computing the edit distance is equivalent to searching for the mapping which comprises the maximum common substructure tree. Therefore, the similarity based on an edit distance is considered the global one.

In this paper we propose the *local similarity* between two semi-ordered trees based on [4] and [7]. Our idea is to use a *local mapping* to evaluate local similarity between semi-ordered trees. Local similarity aims at the best pair of subtrees, such that the optimal similarity based on the scoring system is the best possible. We give an algorithm to find the local mapping by calculating local similarity within two semi-ordered trees. The algorithm is based on a dynamic programming algorithm and extracts the local structure with the highest similarity. The local structure gives more relevant results than the global approach when we deal with biological objects [4].

This paper is organized as follows: In the following section we formally define semi-ordered trees and constrained mappings. In Section 3, we define the local similarity between two semi-ordered trees and the dynamic programming algorithm for computing the local similarity. In the last section, we give a conclusion showing the complexity of our algorithm, and explain our future work.

2 Notation and Definition

A *labeled rooted tree* is $T = (V, E, r, \alpha)$ where V is a set of *vertices*, E is a set of *edges*, r is the root vertex and α is a *label function*. Unless otherwise noted, we represent a labeled rooted tree as $T = (V, E)$ for simplification. Moreover, we often identify T with its vertex set V. A label function is defined as

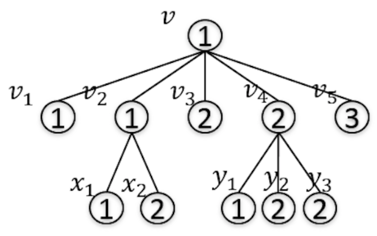

Fig. 1. An example of a semi-ordered tree

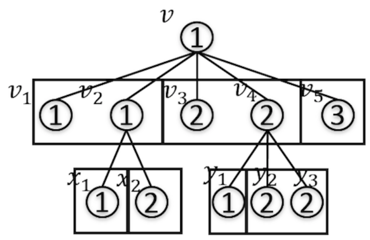

Fig. 2. An example of a partition of a semi-ordered tree

$\alpha : V \to \Sigma$, where Σ is an alphabet. A tree $\theta = (\emptyset, \emptyset)$ is especially called the *empty tree*. A subtree of T rooted at a vertex v which includes all descendants of v and v itself are denoted by $T[v]$. The forest which is induced by deleting v from $T[v]$ and deleting edges which connect v with children of v is denoted by $F[v]$. If X is the set of root vertices of trees in $F[v]$, $F[v]$ is denoted by $F[X]$. The set of children of a vertex v is denoted by $child(v)$. The nearest common ancestor of vertices v and w is denoted by $nca(v, w)$. The ancestor-descendant relation is denoted by $<$ or $\leq$, and $v < w$ means w is an ancestor of v and $v \leq w$ means $v < w$ or $v = w$. The *prefix of a tree* is either the empty tree or a patial subtree $T' = (V', E')$ of $T[v]$ such that, for $V' \subseteq V$ and every $u \in V' \setminus \{v\}$, there exists a parent vertex v' of u satisfying that $v' \in V'$. For a tree $T[v] = (V, E)$ and a vertex $r \in V$, a *local structure* of $T[v]$ is a prefix of $T[r]$. For a tree $T = (V, E)$ and $v \in V$, let the set of all prefixes of $T[v]$ be $Tpre[v]$. For a forest $F[v]$, we define the set of all *prefixes of a forest* $F[v]$ whose root vertices are $v_1, \ldots, v_n$ as $Fpre[v] = \{\{t_1, \ldots, t_n\} \mid t_1 \in Tpre[v_1], \ldots, t_n \in Tpre[v_n]\}$.

Semi-ordered trees

Well-known classes of trees are ordered trees and unordered trees. In this paper, we treat the class of semi-ordered trees, proposed in [7], which is a super class of both of the classes. A semi-ordered tree is a tree with a total *semi-order relation* defined on the set of children of each vertex. A semi-order relation $\sqsubseteq$ is a reflexive and transitive binary relation. If $\sqsubseteq$ satisfies the comparability, $\sqsubseteq$ is called a *total semi-order relation.*

Definition 1 (Semi-ordered tree). A semi-ordered tree is a pair $(T, S_{\sqsubseteq})$ with $S_{\sqsubseteq} = \{\sqsubseteq_v \mid v \in T\}$ where $\sqsubseteq_v$ is a total semi-order relation on $child(v)$.

An example of semi-ordered trees is shown in Fig. 1. The children of a vertex of a semi-ordered tree are divided into equivalence classes which are totally ordered (see Fig.2). For any $v \in T$ and s, $t \in child(v)$, we represent a binary relation which satisfies $s \sqsubseteq_v t$ and $t \not\sqsubseteq_v s$ as $s \sqsubset_v t$. For any $v, w \in V$, $v \sqsubseteq w$ is defined as follows:

$$\forall v, w \in V.\ v \sqsubseteq w \iff \begin{cases} v \leq w \text{ , or} \\ \exists s, t \in child(nca(v, w)) \text{ s.t. } v \leq s, w \leq t, \text{ and } s \sqsubset_{nca(v,w)} t. \end{cases}$$

For every $s, t \in child(v)$, we define

$$s \equiv t \iff s \sqsubseteq_v t \,\text{and}\, t \sqsubseteq_v s.$$

For any $s \in child(v)$, the equivalence class of the children s is denoted by $[s]$:

$$[s] = \{t \in child(v) \mid s \equiv t\}.$$

The division of $child(v)$, denoted by $C(v)_{\equiv}$, is defined as

$$C(v)_{\equiv} = \{[s] \mid s \in child(v)\}.$$

In Fig.2, we show an example of the division of the vertices of the tree in Fig.1, the division of $child(v)$ is $C(v)_{\equiv} = \{\{v_1, v_2\}, \{v_3, v_4\}, \{v_5\}\}$. A tree T is unordered iff $|C(v)_{\equiv}| = 1$ for every $v \in T$, and ordered iff $|C(v)| = |child(v)|$ for every $v \in T$.

Constrained mappings

A *mapping* is a set of pairs of vertices. Various types of mappings have been proposed. They are distinguished by their restrictions, for example, the restriction between the ancestor-descendent relation.

Our method adopts the *constrained mappings* proposed by Zhang [13]. Let $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ be rooted semi-ordered trees. A subset M of $V_1 \times V_2$ is called a constrained mapping on semi-ordered trees if any pairs $(u_1, u_2), (v_1, v_2)$ and $(w_1, w_2) \in M$ satisfy all of the four conditions.

$$\begin{aligned} u_1 = v_1 &\iff u_2 = v_2 \text{ (one-to-one relation)}, \\ u_1 \le v_1 &\iff u_2 \le v_2 \text{ (ancestor-descendant preservation)}, \\ u_1 \sqsubseteq v_1 &\iff u_2 \sqsubseteq v_2 \text{ (total semi-order preservation), and} \\ w_1 < nca(u_1, v_1) &\iff w_2 < nca(u_2, v_2) \text{ (structure preservation)}. \end{aligned}$$

Restricted mappings

For two trees $T[v]$ and $T[w]$, let X_1 and X_2 be subsets of $child(v)$ and $child(w)$, respectively. Let $V_{F[X_1]}$ and $V_{F[X_2]}$ be all the vertices of $F[X_1]$ and $F[X_2]$, respectively. A constrained mapping M between two forests $F[X_1]$ and $F[X_2]$ is called a *restricted mapping* if any (v_1, v_2) and $(w_1, w_2) \in M$ satisfy the following condition:

$$nca(v_1, w_1) \in V_{F[X_1]} \iff nca(v_2, w_2) \in V_{F[X_2]}.$$

3 Local similarity

Scoring systems

If we use a distance function between trees to find local structures, the smaller two local structures are, the higher the similarity between them is. Then, it is obvious that the similarity between two local structures is the maximum when each of the local structures is an empty tree. In order to avoid outputting such trivial results, we introduce a point addition scoring system, not a point deduction scoring system. We prepare a new symbol, called an empty symbol denoted by $\lambda \notin \Sigma$. The set of integers are denoted by $\mathbb{Z}$. A function $s : (\Sigma \cup \{\lambda\}) \times (\Sigma \cup \{\lambda\}) \to \mathbb{Z}$ is a called *scoring function* if it satisfies all of the four conditions:

$$\begin{aligned} &s(v, w) = s(w, v), \\ &s(v, \lambda) = s(\lambda, w) < 0, \\ &s(v, v) > s(v, w) \text{ (if } v \ne w\text{), and} \\ &s(v, w) \ge s(v, \lambda) + s(\lambda, w). \end{aligned}$$

For all vertices, hereafter, $s(\alpha(v), \alpha(w))$ is represented as $s(v, w)$ for simplification.

Let $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ be rooted semi-ordered trees and M be a mapping. Then $M|_1$ and $M|_2$ are defined as follows.

$$\begin{aligned} M|_1 &= \{v_1 \in V_1 \mid \exists v_2 \in V_2, (v_1, v_2) \in M\}, \text{ and} \\ M|_2 &= \{v_2 \in V_2 \mid \exists v_1 \in V_1, (v_1, v_2) \in M\}. \end{aligned}$$

The score of a mapping M, denoted by $S(M, V_1, V_2)$, is defined as

$$S(M, V_1, V_2) = \sum_{(v,w) \in M} s(v, w) + \sum_{v \in V_1 - M|_1} s(v, \lambda) + \sum_{w \in V_2 - M|_2} s(\lambda, w).$$

The set of constrained mappings between T_1 and T_2 is denoted by $Map[T_1, T_2]$. The maximum score between two prefixes of two trees is denoted by $PS(T[v], T[w])$. For $\tau_1 \in Tpre[v]$ and $\tau_2 \in Tpre[w]$, it is defined as

$$PS(T_1, T_2) = \max_{\substack{\tau_1 \in Tpre[v] \\ \tau_2 \in Tpre[w]}} \max_{M \in Map[\tau_1, \tau_2]} S(M, \tau_1, \tau_2).$$

The maximum score between the prefixes of forests is denoted by $PS(F[v], F[w])$. As is the case of $PS(T_1, T_2)$, $PS(F[v], F[w])$ is defined. We define the maximum similarity $LS(T_1, T_2)$ as

$$LS(T_1, T_2) = \max_{\substack{v \in V_1 \\ w \in V_2}} PS(T_1[v], T_2[w]).$$

Recursive formula of local mapping between semi-ordered trees

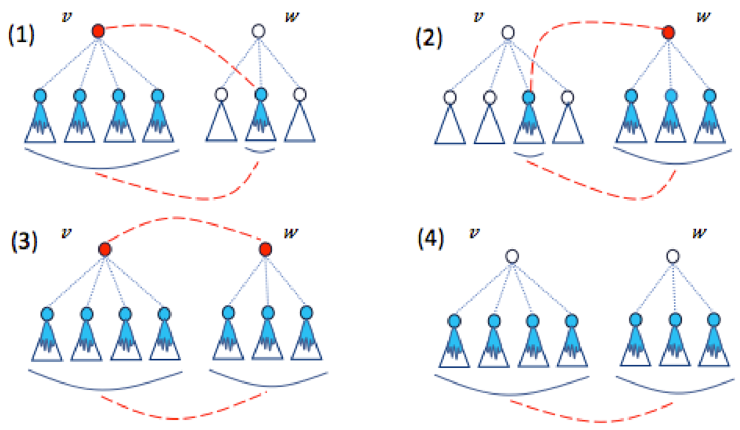

Fig. 3. Classifying a semi-ordered tree based on the root node

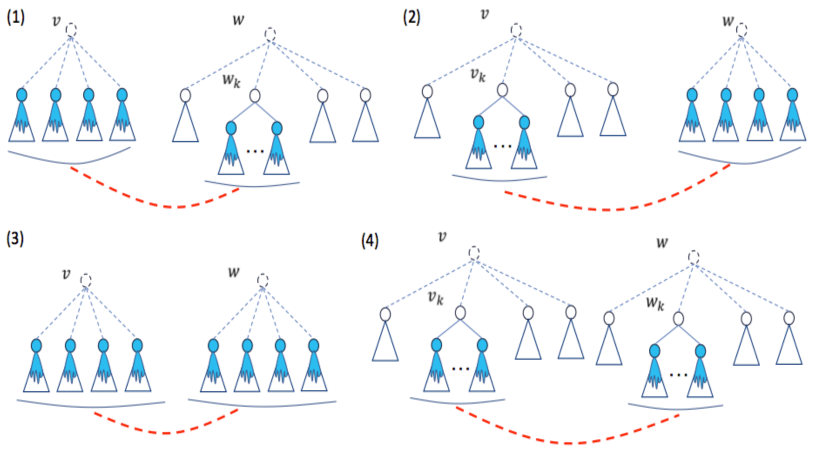

Fig. 4. Classifying a semi-ordered forest based on the constrained mapping

First, we classify a pair of semi-ordered trees by focusing on the root vertex. There are four cases of the classification in Fig. 3. However, the score of the case (4) in Fig. 3 is always smaller than the case (3) in Fig. 3 because of the definition of s, $s(v,w) \geq s(v,\lambda) + s(\lambda,w)$. Therefore, we do not have to take the case (4) into account.

Proposition 1. *The local score between two semi-ordered trees $T_1[v]$ and $T_2[w]$ is represented as*

$$PS(T_1[v], T_2[w]) = \max \begin{cases} 0, & \\ \max_{w_k \in child(w)} PS(T_1[v], T_2[w_k]) + s(\lambda, w), & \text{((1) in Fig.3)} \\ \max_{v_k \in child(v)} PS(T_1[v_k], T_2[w]) + s(v, \lambda), & \text{((2) in Fig.3)} \\ PS(F[v], F[w]) + s(v, w). & \text{((3) in Fig.3)} \end{cases}$$

Next, we classify a pair of semi-ordered forests based on the constrained mapping on the semi-ordered trees. We have to consider four cases of the classification as shown in Fig 4. As with Proposition 1, we should not take the case (4) in Fig. 4 into account.

Proposition 2. *Let the set of restricted mapping between two semi-ordered forests $F[v]$ and $F[w]$ be $R(F[v], F[w])$. Then, the local score $PS(F[v], F[w])$ is represented as*

$$PS(F[v], F[w]) = \max \begin{cases} 0, & \\ \max_{w_k \in child(w)} PS(F[v], F[w_k]) + s(\lambda, w_k), & \text{((1) in Fig.4)} \\ \max_{v_k \in child(v)} PS(F[v_k], F[w]) + s(v_k, \lambda), & \text{((2) in Fig.4)} \\ \max_{M \in R(F[v], F[w])} S(M, F[v], F[w]). & \text{((3) in Fig.4)} \end{cases}$$

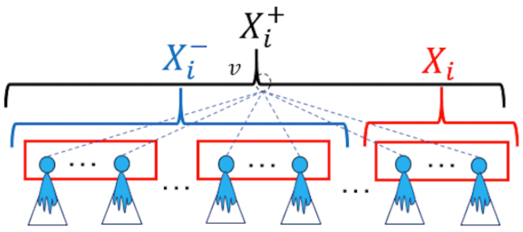

Fig. 5. An exmple of X_i, X_i^- and X_i^+.

Third, we classify a semi-ordered forest based on a semi-order relation. For simplification, $S_R(F[v], F[w])$ is defined as

$$S_R(F[v], F[w]) = \max_{M \in R(F[v], F[w])} S(M, F[v], F[w]).$$

For the set of all root vertices V_1 and V_2 of $F[v]$ and $F[w]$, respectively, we define sets of vertices X_1 and X_2 as

$$X_i = \{w \in V_i \,|\, \forall v \in V_i, v \sqsubseteq w\} \ (i = 1, 2)$$

Moreover, we define X_1^-, X_1^+, X_2^-, and X_2^+ as

$$X_i^- = \{v_i \in V_i \mid \exists v_j \in X_i \ \text{ s.t. } v_i \sqsubset v_j\}, \text{ and } X_i^+ = \{v_i \in V_i \mid \exists v_j \in X_i \text{ s.t. } v_i \sqsubseteq v_j\}. \ (i = 1, 2)$$

For X_i^-, X_i^+ and X_i, it holds that $X_i^+ = X_i^- \cup X_i$. We show an example of X_i, X_i^- and X_i^+ in Fig. 5.

Proposition 3. *The score between two forests, $F[X_1^+]$ and $F[X_2^+]$, which is based on a restricted mapping between the forests is represented recursively as*

$$S_R(F[X_1^+], F[X_2^+]) = \max \begin{cases} 0, \\ S_R(F[X_1], F[X_2]) + S_R(F[X_1^-], F[X_2^-]), \\ S_R(F[X_1^-], F[X_2^+]), \\ S_R(F[X_1^+], F[X_2^-]). \end{cases}$$

We show how to calculate $S_R(F[X_1], F[X_2])$ below. Let $I_1 = \{1, \dots, |X_1|\}$ ($I_2 = \{1, \dots, |X_2|\}$) be the set of indices of elements of $X_1 = \{v_1, \dots, v_{|X_1|}\}$ ($X_2 = \{w_1, \dots, w_{|X_2|}\}$), and $I = I_1 \times I_2$. For a restricted mapping $M \in R(F[v], F[w])$, $i_1 \in I_1$, and $i_2 \in I_2$, the mapping $M^{(i_1,i_2)}$ between $T_1[v_{i_1}] \in F[v]$ and $T_2[w_{i_2}] \in F[w]$ is defined as

$$M^{(i_1,i_2)} = M \cap (T_1[v_{i_1}] \times T_2[v_{i_2}]).$$

Then, because it holds that

$$M = \bigcup_{(i_1,i_2) \in I} M^{(i_1,i_2)},$$

and the property of the restricted mapping, we obtain that

$$S(M) = \sum_{(i_1,i_2) \in I} S(M^{(i_1,i_2)}).$$

Therefore, a mapping M which satisfies the following formula gives the local score.

$$S(M) = \max_I \{ \sum_{(i_1,i_2) \in I} PS(\tau_1[v_{i_1}], \tau_2[w_{i_2}]) \}.$$

Finding a mapping M which satisfies the above formula is reduced to the bipartite graph maximum score matching problem. In constructing a bipartite graph from two semi-ordered trees, it is important to set two dummy vertices which are e_v and e_w as seen in Fig. 6. An example of a graph which reduces from two forests $F[v]$ and $F[w]$ is shown Fig. 6.

We show a pseudocode which calculates the optimal local mapping and the local score based on it between two semi-ordered trees in Algorithm 1 and a pseudocode which calculates the local structures between two semi-ordered trees in Algorithm 2.

Algorithm 1 Local Score and Local mappings between Semi-ordered Trees

```
  INPUT : Semi-ordered trees $T_1, T_2$
  $LS(\theta, \theta) = 0, MappingList = \emptyset$
  for $v$ in $T_1$ and $w$ in $T_2$ do
    $LS(T_1[v], \theta) = 0, LS(F[v], \theta) = 0$
    $LS(\theta, T_2[w]) = 0, LS(\theta, F[w]) = 0$
  end for
  for $v$ in $T_1$ and $w$ $in$ $T_2$ do
    $M_{v,w} = \emptyset$
    for $X_1$ in $C(v)_{\equiv}$ and $X_2$ in $C(w)_{\equiv}$ do
      Calculate $S_R(F[X_1], F[X_2])$ by reducing to a graph maximum score matching problem.
      Calculate $S_R(F[X_1^+], F[X_2^+])$ by Proposition 3.
    end for
    Calculate $PS(F[v], F[w])$ by Proposition 2.
    Calculate $PS(T[v], T[w])$ and the local mapping $M$ by Proposition 1.
    Add the local mapping $M_{v,w}$ to $MappingList$.
  end for
  for $v$ in $T_1$ and $w$ in $T_2$ do
    $LS(T_1, T_2) = \max_{v,w} PS(T_1[v], T_2[w])$.
    $(v_{max}, w_{max}) = (v, w)$ satisfying $PS(T_1[v_{max}], T_2[w_{max}]) = LS(T_1, T_2)$.
  end for
  $LocalMapping = M_{v_{max}, w_{max}} \in MappingList$
  OUTPUT : $LS(T_1, T_2)$ and $LocalMapping$.
```

Algorithm 2 Local structures between semi-ordered trees

```
  INPUT : Semi-ordered trees $T_1, T_2$, and a Local Mapping $M$.
  $LocalStructure_1, LocalStructure_2 = \emptyset$
  Calculate $M|_1$ and $M|_2$ from $M$.
  for $i = 1, 2$ do
    for $v$ in $M|_i$ do
      $V$ is the set of vertices in the paths of from $v$ and $v_{before}$ to $nca(v, v_{before})$ in $T$
      for $w$ in $V$ do
        if $w \notin LocalStructure_i$ then
          Add $w$ to $LocalStructure_i$.
        end if
      end for
      $v_{before} = v$
    end for
  end for
  OUTPUT : $LocalStructure_1, LocalStructure_2$
```

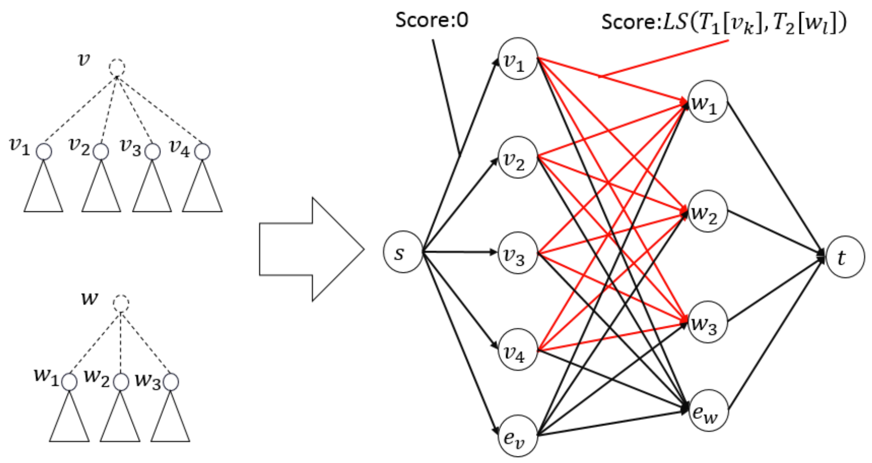

Fig. 6. An example of a graph which reduces our problem on two forests into an instance of the bipartite maximum score matching problem.

4 Concluding Remarks

In this paper, we propose a polynomial algorithm which calculates the local mapping between two semi-ordered trees through constrained mappings. The time complexity is the same as the time complexity to compute the constrained edit distance between semi-ordered trees [7], that is,

$$O(|V_1||V_2| \max\{D(T_1), D(T_2)\} \log_2 \max\{D(T_1), D(T_2)\}),$$

where $D(T)$ represents the maximum degrees of T, because

$$O(\sum_{x \in T_1} \sum_{y \in T_2} \sum_{X_1 \in C(x)_\equiv} \sum_{X_2 \in C(y)_\equiv} |X_1||X_2|(|X_1| + |X_2|) \log_2 (|X_1| + |X_2|)) \\ \leq O(|V_1||V_2| \max\{D(T_1), D(T_2)\} \log_2 \max\{D(T_1), D(T_2)\}).$$

We can improve the time complexity to

$$O(|V_1||V_2| \min\{D(T_1), D(T_2)\})$$

by using the same technique as in [15].

Modelling plant architectures is a typical example of an application of semi-ordered trees. Calculating the local score is useful for measuring the relevance between biological objects. Therefore we can measure the local similarity between semi-ordered trees which are better models than ordered trees by using the proposed method more effective.

In our future work, we should compute the edit distance among multiple semi-ordered trees and calculate the local simirality between semi-ordered trees through various mappings other than the constrained mapping, especially the less-constrained mapping because the problem of computing the local similarity between semi-ordered trees through less-constrained mappings is a MAX-SNP hard problem.

References

1. P. Bille : A survey on tree edit distance and related problems, *Theoretical Computer Science*, Vol. 337, pp.217-239 (2005).
2. J. Edmonds and R. M. Karp : Theoretical improvements in algorithmic efficiency for network flow problems, *Journal of the Association for Computing Machinery*, Vol. 19, pp. 248-264 (1972).
3. P. Ferraro, L. Tichit and S. Dulucq : Local similarity between trees, In *Conference JOBIM'04, 5th Open Days in Biology, Computer Science and Mathematics*, pp. 19-30 (2004).
4. P. Ferraro and A. Ouangraoua : Local mapping between unordered trees, *Tech.Report RR-105*, pp. 1-13 (2005).
5. C. Godin and Y. Caraglio : A multiscale model of plant topological structures, *Journal of Theoretical Biology*, Vol. 191, pp. 1-46 (1998).

6. A. Ouangraoua and P. Ferraro : Local similarity between quotiented ordered trees, *Journal of Discrete Algorithms*, Vol. 5, pp. 23-35 (2007).
7. A. Ouangraoua and P. Ferraro : A constrained edit distance algorithm between semi-ordered trees, *Theoretical Computer Science*, Vol. 410, pp. 837-846 (2009).
8. T. Smith and M. Waterman : Identification of common molecular subsequences, *Journal of Molecular Biology*, Vol. 147, pp. 195-197 (1981).
9. K. C. Tai : The tree-to-tree correction problem, *J. Assoc. Comput.*, pp. 422-433 (1979).
10. R. E. Tarjan : Data Structures and Network algorithms, *CBMS-NFS-Regional Conference Series In Applied Mathematics*, (1983).
11. R. A. Wagner and M. J. Fisher : The string-to-string correction problem, *J. Assoc. Comput.*, Vol. 21, pp. 168-173 (1974).
12. K. Zhang and T. Jiang : Some max SNP-hard results concerning unordered labeled trees and related problems, *Pattern Recognition*, Vol. 15, pp. 205-222 (1994).
13. K. Zhang : Algorithms for constrained editing distance between ordered labeled trees and related problems, *Pattern Recognition*, Vol. 28(3), pp. 463-474 (1995).
14. K. Zhang : A constrained edit distance between unordered labeled trees, *Algorithmica*, Vol. 15, pp. 205-222 (1996).
15. Y. Yamamoto, K. Hirata, T. Kuboyama : Tractable and intractable variations of unordered tree edit distance, *Internat. J. Found. Comput. Sci.*, 25, 307-329 (2014)