アルゴリズム 48－4
（1995．11．17）

順序がない木の最大類似部分問題

劉　紹明†　　田中　栄一††

† 神戸大学 自然科学研究科

†† 神戸大学 工学部

根があり順序がない木を R-木と言い，根がなく順序がない木を単に“木”と言う．本論文では，二つの R-木 T_a，T_b，あるいは二つの木 T_a，T_bについて，T_aと類似している部分を T_b内で探す問題を論じ，T_aと類似しているT_bの最大部分の１つを抽出するアルゴリズムを述べている．アルゴリズムの時間計算量と空間計算量は，R-木の場合も木の場合も，それぞれ $O_T(m^3N_aN_b)$ と $O_S(mN_aN_b)$ である．ここで，$N_a(N_b)$ は $T_a(T_b)$ の頂点数を表し，m は T_aと T_bの最大の頂点次数を表す．

Largest Similar Substructure Problems for Unordered Trees (Extended Abstract)

Shaoming LIU† and Eiichi TANAKA††

† The Graduate School of Science and Technology, Kobe University

†† Faculty of Engineering, Kobe University

This paper discusses the problems of largest similar substructures(in short, LSS) in rooted and unordered trees(in short, *R-trees*) and those in unrooted and unordered trees(in short, *trees*). For two *R-trees*(or *trees*) T_a and T_b, LSS in T_b to T_a is defined, and two algorithms for finding one of the LSSs for *R-trees* and that for *trees* are proposed. The time and space complexities of both algorithms are $O_T(m^3N_aN_b)$ and $O_S(mN_aN_b)$, respectively, where m is the largest degree of a vertex of T_a and T_b, and $N_a(N_b)$ is the number of vertices of $T_a(T_b)$.

1 Introduction

The similar structure search problem has arisen from practical topics. For example, the study of the relationship between the structures of chemical compounds and their properties is one of the most important problems in chemistry[1], and several substructure search systems have already been proposed[2]. However, the similar structure search problem has not been studied mathematically for general graphs except trees embedded in a plane[3].

As similarity measures between trees, several distances for rooted and ordered trees(in short, *RO-trees*)[4]~[7], those for trees embedded in a plane[8], those for unordered trees[9]~[11] have already been proposed. Recently, the problem of largest similar substructures(in short, LSS) in trees embedded in a plane was discussed[3]. An object with tree structure is not always embedded in a plane. However, the computation of the Tai distance[4] between unordered trees is NP-hard[12]. Furthermore, Tai mapping is not appropriate to the evaluation of a similarity between trees[10]. In this paper we use a maximal closest ancestor mapping to discuss the problems of LSS in rooted and unordered trees(in short, *R-tree*) and that in unrooted and unordered trees(in short, *tree*), and propose two efficient algorithms for finding one of the LSSs in *R-trees* and that in *trees*. Those algorithms can be applied to structure-activity studies and structure comparison problems.

2 Mappings

Let $T = (V, E)$ be a *tree*, where V is the set of vertices and E is the set of edges. In this paper, all of the vertices in trees are labeled and numbered. Let $lab(x)$ and $deg(x)$ denote the label and the degree of vertex x, respectively. $m_a(m_b)$ is the largest degree of a vertex in *tree* $T_a(T_b)$, where $T_a = (V_a, E_a)$ and $T_b = (V_b, E_b)$. Vertex x with $deg(x) = 1$ is called a leaf. If a *tree* has a root, we call it an *R-tree*. Let $T_a(u) = (V_a(u), E_a(u))$ be the *R-tree* of T_a with root u.

Consider an *R-tree*. Let $dep(x)$ denote the depth of vertex x. If x is a root, $dep(x) = 0$. If $dep(x) \neq 0$, the neighbour of x with depth $dep(x) - 1$ is called the parent of x and denoted by $pa(x)$. The root does not have its parent. Let $An(x)$ be the set of proper ancestors of x. Let $\hat{An}(x, x') = An(x') - An(x)$ for $x \in An(x')$, where "$A - B$" denotes that removing all the elements in set B from set A. If x is not a leaf, the neighbours of x with depth $dep(x) + 1$ are called the children of x. A leaf does not have children. Let $Ch(x)$ denote the set of children of x. For any vertices x_1 and $x_2 (x_1 \neq x_2)$, if $x_1 \notin An(x_2)$ and $x_2 \notin An(x_1)$, x_1 and x_2 are called to be separated, and it is denoted by $sep(x_1, x_2)$. If $sep(x_1, x_2)$, let $ccan(x_1, x_2)$ be the closest common ancestor of x_1 and x_2.

Let $T_a(u, x) = (V_a(u, x), E_a(u, x))$ be the subtree of $T_a(u)$ with root x. Let $x_i (i = 1, 2, \cdots, |Ch(x)|)$ denote a child of x, and $F_a(u, \bar{x}) = (V_a(u, \bar{x}), E_a(u, \bar{x}))$ denote the forest that consists of trees $T_a(u, x_1)$,$T_a(u, x_2)$, $\cdots$, $T_a(u, x_{|Ch(x)|})$, where |A| indicates the number of elements of set A.

Consider a mapping from vertices of $T_a(u)$ to those of $T_b(v)$. If i maps to j, we write (i, j), where $i \in V_a(u)$ and $j \in V_b(v)$. Let M be the set of (i, j)s. If M satisfies the following conditions, M is called a closest common ancestor mapping(in short, CM) from $T_a(u)$ to $T_b(v)$[9].

For any two pairs (i_1, j_1), $(i_2, j_2) \in M$,

(a1) $i_1 = i_2$ iff $j_1 = j_2$,

(a2) $i_1 \in An(i_2)$ iff $j_1 \in An(j_2)$,

(a3) if $sep(i_1, i_2)$, then
$(ccan(i_1, i_2), ccan(j_1, j_2)) \in M$.

Let M be a CM from $T_a(u)$ to $T_b(v)$. If there is no CM M' from $T_a(u)$ to $T_b(v)$ such that $M' = \{(i, j)\} \cup M$, M is called a maximal CM(in short, MCM) from $T_a(u)$ to $T_b(v)$[11]. Let M be a mapping between forests $F_a(u, \bar{x})$ and $F_b(v, \bar{y})$. If $M \cup \{(x, y)\}$ is an MCM from $T_a(u, x)$ to $T_b(v, y)$, M is called an MCM from $F_a(u, \bar{x})$ to $F_b(v, \bar{y})$.

If M satisfies (a1) and (a2), M is called a Tai mapping(in short, TM). Ref. [6] defined a strongly structure preserving mapping(in short, SSPM). SSPM can make more suitable correspondences between the similar substructures of A and those of B than TM[10], and CM is a subclass of SSPM[9]. From the definitions of CM and MCM, any MCM is a CM, but a CM

is not necessarily an MCM. To express a largest similar substructure in *R-trees*(or *trees*), MCM is better than CM and SSPM, since the number of corresponding vertices between *R-trees*(or *trees*) by MCM is larger than or equal to that by CM or SSPM. Let M be a mapping from *tree* T_a to *tree* T_b. If there are vertices u and v such that M is an MCM from *R-tree* $T_a(u)$ to *R-tree* $T_b(v)$, M is an MCM from T_a to T_b[11].

3 LSS in *R-trees* and *Trees*

Let M be an MCM from $T_a(u)$ to $T_b(v)$. If $(i,j) \in M$ and $lab(i) \neq lab(j)$, we say that $lab(j)$ is substituted to $lab(i)$. For any vertex y of $T_b(v)$ and a vertex i of $T_a(u)$, if $(i,y) \notin M$, i is deleted. For any vertex x of $T_a(u)$ and a vertex j of $T_b(v)$, if $(x,j) \notin M$, j is inserted. By this interpretation, we can see that a mapping defines a transformation from $T_a(u)$ to $T_b(v)$. Let M_{xy} denote an MCM from $T_a(u,x)$ to $T_b(v,y)$. If no confusion occurs, we use M_{uv} to indicate an MCM from $T_a(u)$ to $T_b(v)$. Define $\mathcal{J}(M_{xy}) = \{j \mid (i,j) \in M_{xy}\}$. Let r_{xy} be the root of subtree of $T_b(v)$ such that it includes $\mathcal{J}(M_{xy})$ and the number of its vertices is smallest. If all the vertices of a subtree of $T_b(v)$ are inserted, it is called that the subtree is inserted. Let $I_{M_{xy}}$ denote "the set of vertices of all the inserted subtrees of $T_b(v,y)$ determined by M_{xy}".

$$I_{M_{xy}} = \{j \mid V_b(v,j) \subset (V_b(v,y) - \mathcal{J}(M_{xy}))\}. \quad (1)$$

A similar substructure in $T_b(v,y)$ to $T_a(u,x)$ determined by M_{xy}, denoted by $\mathbf{S}_{M_{xy}}$, is defined as the part of $T_b(v,y)$ obtained by removing the vertices of $\left(I_{M_{xy}} \cup \widehat{A}n(y, r_{xy})\right)$ from $T_b(v,y)$. Let $S_{M_{xy}}$ denote the set of vertices of $\mathbf{S}_{M_{xy}}$. $S_{M_{xy}}$ can be expressed as follows.

$$S_{M_{xy}} = V_b(v,y) - I_{M_{xy}} - \widehat{A}n(y, r_{xy}). \quad (2)$$

That is, one M_{xy} determines one $\mathbf{S}_{M_{xy}}$. Let p, q and r be the weights of a substitution, an insertion and a deletion, respectively. p, q and r are positive and $p = q = r \geq 1$. Assume that M_{xy} determines n_d deleted vertices in $T_a(u,x)$, n_s substituted vertices and n_i inserted vertices in $\mathbf{S}_{M_{xy}}$. The weight to transform $T_a(u,x)$ to $\mathbf{S}_{M_{xy}}$, denoted by $W(M_{xy})$, is defined as $W(M_{xy}) = rn_d + pn_s + qn_i$.

Example 1 Consider an MCM $M_{xy} = \{(x,3'),(1,5'),(2,7')\}$ shown in Figure 1(a). We have $I_{M_{xy}} = \{1',6',8',9',10'\}$, $r_{xy} = 3'$ and $\widehat{A}n(y, r_{xy}) = \widehat{A}n(y,3') = \{y,2'\}$. The similar substructure $\mathbf{S}_{M_{xy}}$ is shown in Figure 1(b). The weight $W(M_{xy})$ is $\delta(x,3') + \delta(1,5') + \delta(2,7') + q$, where q is the weight of inserted vertex $4'$ and

$$\delta(x,y) \triangleq \begin{cases} 0 & : lab(x) = lab(y), \\ p & : lab(x) \neq lab(y). \end{cases}$$

□

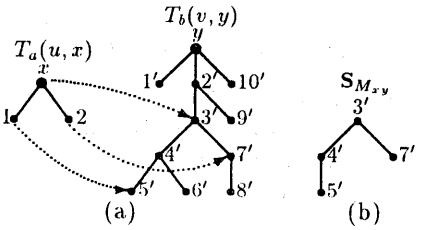

Figure 1 : (a) Mapping $M_{xy} = \{(x,3'),(1,5'),(2,7')\}$, and (b) the similar substructure $\mathbf{S}_{M_{xy}}$ determined by M_{xy}.

The distance from $T_a(u,x)$ to "similar substructure in $T_b(v,y)$ to $T_a(u,x)$", denoted by $D(u,x,v,y)$, is defined as follows.

$$D(u,x,v,y) = \min_{M_{xy}} W(M_{xy}). \quad (3)$$

Note that there is at least one MCM that determines $D(u,x,v,y)$. Let the collection of such mappings be $\{M^1_{xy}, M^2_{xy}, \cdots, M^d_{xy}\}$. An LSS in $T_b(v,y)$ to $T_a(u,x)$, denoted by $\mathbf{S}_{xy}$, is defined as $\mathbf{S}_{M^k_{xy}}$ such that $\left|S_{M^k_{xy}}\right| = \max_{1 \leq i \leq d}\left\{\left|S_{M^i_{xy}}\right|\right\}$. An LSS is a tree. Let S_{xy} denote the set of vertices of $\mathbf{S}_{xy}$.

Let $D(u,v)$ denote the distance from $T_a(u)$ to "similar substructure in $T_b(v)$ to $T_a(u)$", and $\mathbf{S}_{uv}$ be an LSS in $T_b(v)$ to $T_a(u)$. S_{uv} indicates the set of vertices of $\mathbf{S}_{uv}$. Hereafter, SSD stands for "the distance from $T_a(u,x)$ to 'similar substructure in $T_b(v,y)$ to $T_a(u,x)$'". If there are vertices u and v such that a substructure of T_b is a similar substructure in $T_b(v)$ to $T_a(u)$, the substructure of T_b is called a similar substructure in T_b to T_a. The distance from T_a to "similar substructure in T_b to T_a", denoted by D, is defined as follows.

$$D = \min_{u \in V_a, v \in V_b}\{D(u,v)\}. \quad (4)$$

Let Z be the set of pairs of vertices u and v such that $D = D(u, v)$. An LSS in T_b to T_a, denoted by $\mathbf{S}$, is defined as an LSS in $T_b(v)$ to $T_a(u)$ determined by $\max_{(u,v)\in Z}\{|S_{uv}|\}$. S denotes the set of vertices of $\mathbf{S}$.

4 A Computing Method of SSD

Consider two *R-trees* shown in Figure 2. From the definition of MCM, MCM satisfies the mapping conditions of TM. Similar to Ref. [7], TM from $T_a(u, x)$ to $T_b(v, y)$ can be classified into the following four types.

(b1) Vertex x maps to vertex y, and forest $F_a(u, \bar{x})$ maps to forest $F_b(v, \bar{y})$.

(b2) Vertex x' of $F_a(u, \bar{x})$ maps to vertex y, and forest $F_a(u, \bar{x}')$ maps to forest $F_b(v, \bar{y})$. The vertices of $T_a(u, x)$ except those of $T_a(u, x')$ are deleted.

(b3) Vertex x maps to vertex y' of $F_b(v, \bar{y})$, and forest $F_a(u, \bar{x})$ maps to forest $F_b(v, \bar{y}')$.

The vertices of $T_b(v, y)$ except those of $T_b(v, y')$ are inserted.

(b4) Vertex x is deleted, vertex y is inserted, and forest $F_a(u, \bar{x})$ maps to forest $F_b(v, \bar{y})$.

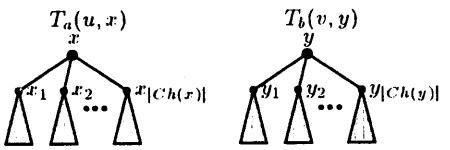

Figure 2: *R-trees* $T_a(u, x)$ and $T_b(v, y)$.

Assume that a mapping from $F_a(u, \bar{x})$ to $F_b(v, \bar{y})$, that from $F_a(u, \bar{x}')$ to $F_b(v, \bar{y})$ and that from $F_a(u, \bar{x})$ to $F_b(v, \bar{y}')$ are MCM. Then, the mappings of type (b1)~(b3) satisfy the conditions of MCM. Let M be an MCM from $F_a(u, \bar{x})$ to $F_b(v, \bar{y})$. Since $M \cup \{(x, y)\}$ is a CM from $T_a(u, x)$ to $T_b(v, y)$, the mapping of type (b4) is not an MCM.

Consider an MCM from $T_a(u, x)$ to $T_b(v, y)$. Let $\Delta 1_{uxvy}$, $\Delta 2_{uxvy}$ and $\Delta 3_{uxvy}$ be the minimum value of $W(M_{xy})$ for all M_{xy} of type (b1), that of type (b2) and that of type (b3), respectively. Note that as described in the previous section, $W(M_{xy})$ means the weight to transform $T_a(u, x)$ to $\mathbf{S}_{M_{xy}}$. From the definition of SSD, we have the following formula.

$$D(u, x, v, y) = \min\{\Delta 1_{uxvy},\ \Delta 2_{uxvy},\ \Delta 3_{uxvy}\}. \tag{5}$$

A similar substructure $\mathbf{S}_{M_{xy}}$ determined by M_{xy} of type (b2) is shown in Figure 3(a). In this case we have

$$\Delta 2_{uxvy} = \min_{x'\in V_a(u,\bar{x})}\{\Delta 1_{ux'vy} + (|V_a(u, x)| - |V_a(u, x')|) \cdot r\}. \tag{6}$$

For the case of type (b3) shown in Figure 3(b), we have

$$\Delta 3_{uxvy} = \min_{y'\in V_b(v,\bar{y})}\{\Delta 1_{uxvy'}\}. \tag{7}$$

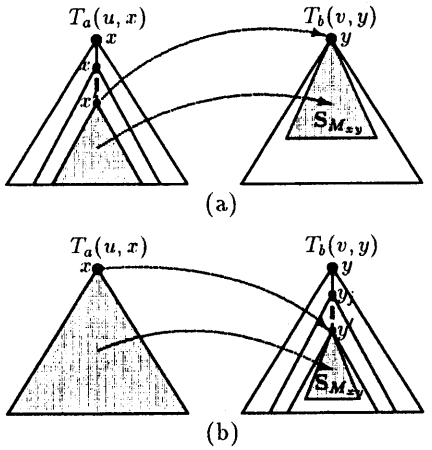

Figure 3: (a) $\mathbf{S}_{M_{xy}}$ determined by M_{xy} of type (b2), and (b) $\mathbf{S}_{M_{xy}}$ determined by M_{xy} of type (b3).

Lemma 1 Assume that all of $\Delta 1_{uxvy}$ and $D(u, x_i, v, y_j)(x_i \in Ch(x),\ y_j \in Ch(y))$ are given. $D(u, x, v, y)$ can be computed by the following formula.

$$D(u, x, v, y) = \begin{cases} \Delta 1_{uxvy}, \\ \min_{x_i\in Ch(x)}\{D(u, x_i, v, y) + (|V_a(u, x)| - |V_a(u, x_i)|) \cdot r\}, \\ \min_{y_j\in Ch(y)}\{D(u, x, v, y_j)\}. \end{cases} \tag{8}$$

Proof. We will prove Lemma 1 for the following three cases.

Case 1: This is the case that $\Delta 1_{uxvy}$ is the mininum value of formula (5). Clearly, we have $D(u, x, v, y) = \Delta 1_{uxvy}$.

Case 2: This is the case that $\Delta 2_{uxvy}$ is the minimum value of formula (5). Consider all M_{xy} of type (b2). Vertex x' maps to y, where x' is a vertex of $T_a(u, x_i)$ and x_i is a child of x. From formula (6), we have

$$\begin{aligned}
&\Delta 2_{uxvy} \\
&= \min_{x' \in V_a(u,\bar{x})} \{\Delta 1_{ux'vy} \\
&\quad +(|V_a(u,x)| - |V_a(u,x')|) \cdot r\} \\
&= \min_{x_i \in Ch(x)} \min_{x' \in V_a(u,x_i)} \{\Delta 1_{ux'vy} \\
&\quad +(|V_a(u,x)| - |V_a(u,x')|) \cdot r\} \\
&= \min_{x_i \in Ch(x)} \Big\{ \min_{x' \in V_a(u,x_i)} \{\Delta 1_{ux'vy} \\
&\quad +(|V_a(u,x_i)| - |V_a(u,x')|) \cdot r\} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r \Big\} \\
&= \min_{x_i \in Ch(x)} \{\min \{\Delta 1_{ux_ivy}, \\
&\quad \min_{x' \in V_a(u,\bar{x}_i)} \{\Delta 1_{ux'vy} \\
&\quad +(|V_a(u,x_i)| - |V_a(u,x')|) \cdot r\}\} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r \} \\
&= \min_{x_i \in Ch(x)} \{\min \{\Delta 1_{ux_ivy}, \ \Delta 2_{ux_ivy}\} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r\}. \qquad (9)
\end{aligned}$$

From the definition of SSD, we have $D(u, x_i, v, y) = \min\{\Delta 1_{ux_ivy}, \ \Delta 2_{ux_ivy}, \ \Delta 3_{ux_ivy}\}$. Assume that "$\min\{\Delta 1_{ux_ivy}, \ \Delta 2_{ux_ivy}\} > \Delta 3_{ux_ivy}$"(*1). Let $L_b(v, y)$ be the set of leaves of $T_b(v, y)$. From the assumption (*1) and formula (9), we have

$$\begin{aligned}
&\Delta 2_{uxvy} \\
&= \min_{x_i \in Ch(x)} \{\min \{\Delta 1_{ux_ivy}, \Delta 2_{ux_ivy}\} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r\} \\
&> \min_{x_i \in Ch(x)} \{\Delta 3_{ux_ivy} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r\} \\
&= \min_{x_i \in Ch(x)} \Big\{ \min_{y' \in V_b(v,\bar{y})} \{\Delta 1_{ux_ivy'}\} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r\} \\
&= \min_{x_i \in Ch(x)} \Big\{ \min_{y' \in V_b(v,\bar{y})} \{\Delta 1_{ux_ivy'} \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)| - 1) \cdot r + p\}\} \\
&\geq \min_{x_i \in Ch(x)} \Big\{ \min_{y' \in V_b(v,\bar{y})} \{\Delta 1_{ux_ivy'} \\
&\quad +\delta(x, pa(y')) \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)| - 1) \cdot r \}\Big\} \\
&\geq \min_{x_i \in Ch(x)} \Big\{ \min_{y' \in V_b(v,\bar{y})} \Big\{\Delta 1_{uxvpa(y')}\Big\}\Big\} \\
&= \min_{y' \in V_b(v,\bar{y})} \Big\{\Delta 1_{uxvpa(y')}\Big\} \\
&= \min \{\Delta 1_{uxvy}, \\
&\quad \min_{y' \in V_b(v,\bar{y}),\ y' \notin L_b(v,y)} \{\Delta 1_{uxvy'}\}\Big\} \\
&\geq \min \Big\{\Delta 1_{uxvy}, \ \min_{y' \in V_b(v,\bar{y})} \{\Delta 1_{uxvy'}\}\Big\} \\
&= \min \{\Delta 1_{uxvy}, \ \Delta 3_{uxvy}\}. \qquad (10)
\end{aligned}$$

Since this contradicts the assumption that "$\Delta 2_{uxvy}$ is the minimum value of formula (5)", the assumption (*1) is wrong. Therefore, we have $\min\{\Delta 1_{ux_ivy}, \ \Delta 2_{ux_ivy}\} \leq \Delta 3_{ux_ivy}$. That is, $D(u, x_i, v, y) = \min\{\Delta 1_{ux_ivy}, \ \Delta 2_{ux_ivy}\}$. From formula (9), we have

$$\begin{aligned}
\Delta 2_{uxvy} &= \min_{x_i \in Ch(x)} \{D(u, x_i, v, y) \\
&\quad +(|V_a(u,x)| - |V_a(u,x_i)|) \cdot r\}. \qquad (11)
\end{aligned}$$

Case 3: This is the case that $\Delta 3_{uxvy}$ is the minimum value of formula (5). Similar to Case 2, we have

$$\Delta 3_{uxvy} = \min_{y_j \in Ch(y)} \{D(u, x, v, y_j)\}. \qquad (12)$$

Summing up the above discussions, we have Lemma 1. □

Hereafter, we will describe a computing method for $\Delta 1_{uxvy}$. If x maps to y, an SSPM from $F_a(u, \bar{x})$ to $F_b(v, \bar{y})$ has the following characteristics.

"For any vertices x_{i_1}, x_{i_2}, y_{j_1} and y_{j_2}($i_1 \neq i_2, x_{i_1}, x_{i_2} \in Ch(x)$, $j_1 \neq j_2$, $y_{j_1}, y_{j_2} \in Ch(y)$), both $T_a(u, x_{i_1})$ and $T_a(u, x_{i_2})$ cannot map to $T_b(v, y_{j_1})$ at the same time, and $T_a(u, x_{i_1})$ cannot map to both $T_b(v, y_{j_1})$ and $T_b(v, y_{j_2})$ at the same time.[6]"(*2)

Since MCM is a subclass of SSPM, (*2) is correct for MCM. Let $M_{\bar{x}\bar{y}}$ denote an MCM from $F_a(u, \bar{x})$ to $F_b(v, \bar{y})$ in the case that $(x, y) \in M_{xy}$. From the definition of $M_{\bar{x}\bar{y}}$, if $(x, y) \in M_{xy}$, we have $M_{\bar{x}\bar{y}} = M_{xy} - \{(x, y)\}$. Let $\langle x_i, y_j \rangle$ denote that $T_a(u, x_i)$ maps to $T_b(v, y_j)$. From (*2), we can express $M_{\bar{x}\bar{y}}$ using set C_{uxvy} of $\langle x_i, y_j \rangle$, that is, $C_{uxvy} = \{\langle x_{i_1}, y_{j_1} \rangle, \ \langle x_{i_2}, y_{j_2} \rangle, \ \cdots, \ \langle x_{i_k}, y_{j_k} \rangle\}$($x_{i_h} \in Ch(x)$, $y_{j_h} \in Ch(y)$, $1 \leq h \leq k$). If $(x, y) \in M_{xy}$, M_{xy} can be expressed as follows.

$$\begin{aligned}
M_{xy} &= \{(x, y)\} \cup M_{\bar{x}\bar{y}}. \\
&= \{(x, y)\} \cup \Big(\bigcup_{\langle x_i, y_j \rangle \in C_{uxvy}} \Big(M_{x_i y_j}\Big)\Big). \qquad (13)
\end{aligned}$$

Consider a C_{uxvy} and an M_{xy} in the case that $(x, y) \in M_{xy}$. Assume that "subtree $T_a(u, x_s)$ ($x_s \in Ch(x)$) is deleted and subtree $T_b(v, y_t)$($y_t \in Ch(y)$) is inserted by M_{xy}"(*3). Since $M_{xy} \cup M_{x_s y_t}$ is a CM from $T_a(u, x)$ to $T_b(v, y)$, M_{xy} is not an MCM. Therefore, the assumption (*3) is incorrect. Then, we have

$$|C_{uxvy}| = \min\{|Ch(x)|, |Ch(y)|\}. \qquad (14)$$

Consider $W(M_{xy})$ in the case that $(x, y) \in M_{xy}$. Assume that $T_a(u, x_i)(x_i \in Ch(x))$ maps

to $T_b(v, y_j)(y_j \in Ch(y))$. Since x maps to y, all the vertices of $\hat{An}(y_j, r_{x_i y_j})$ belong to $S_{M_{xy}}$. From (*2) and formula (13), $W(M_{xy})$ in the case that $(x, y) \in M_{xy}$ and $\Delta 1_{uxvy}$ can be expressed as follows.

$$W(M_{xy}) = \delta(x,y) + \sum_{\langle x_i, y_j\rangle \in C_{uxvy}} \left(W(M_{x_i y_j}) + \left|\hat{An}(y_j, r_{x_i y_j})\right| \cdot q \right) + \sum_{\langle x_i, y_j\rangle \notin C_{uxvy}} |V_a(u, x_i)| \cdot r, \quad (15)$$

$$\begin{aligned}
\Delta 1_{uxvy} &= \min_{M_{xy}} W(M_{xy}) \\
&= \min_{C_{uxvy}} \left\{ \sum_{\langle x_i, y_j\rangle \in C_{uxvy}} \left(\min_{M_{x_i y_j}} \left\{ W(M_{x_i y_j}) + \left|\hat{An}(y_j, r_{x_i y_j})\right| \cdot q \right\} \right) \right. \\
&\quad \left. + \sum_{\langle x_i, y_j\rangle \notin C_{uxvy}} |V_a(u, x_i)| \cdot r \right\} + \delta(x,y) \\
&= \delta(x,y) + \min_{C_{uxvy}} \left\{ \sum_{\langle x_i, y_j\rangle \in C_{uxvy}} \left(\hat{D}(u, x_i, v, y_j) \right.\right. \\
&\quad \left.\left. + \sum_{\langle x_i, y_j\rangle \notin C_{uxvy}} |V_a(u, x_i)| \cdot r \right) \right\}, \qquad (16)
\end{aligned}$$

where

$$\hat{D}(u, x_i, v, y_j) \triangleq \min_{M_{x_i y_j}} \left\{ W(M_{x_i y_j}) + \left|\hat{An}(y_j, r_{x_i y_j})\right| \cdot q \right\}. \quad (17)$$

Before discussing a computing method of $\Delta 1_{uxvy}$, we show a computing method of $\hat{D}(u, x, v, y)$ first. Similar to Lemma 1, we have Lemma 2.

Lemma 2 Assume that $\Delta 1_{uxvy}$, $D(u, x_i, v, y_j)$ and $\hat{D}(u, x_i, v, y_j)(x_i \in Ch(x),\ y_j \in Ch(y))$ are given. $\hat{D}(u, x, v, y)$ can be computed by the following formula.

$$\hat{D}(u, x, v, y) = \begin{cases} \Delta 1_{uxvy}, \\ \min_{x_i \in Ch(x)} \{ D(u, x_i, v, y) + (|V_a(u, x)| - |V_a(u, x_i)|) \cdot r \}, \\ \min_{y_j \in Ch(y)} \left\{ \hat{D}(u, x, v, y_j) + q \right\}. \end{cases} \quad (18)$$

□

Consider a computing method of $\Delta 1_{uxvy}$ based on formula (16). Create a bipartite graph $G(A, B, E)$ from $T_a(u, x)$ and $T_b(v, y)$. $A = A_1 \cup A_2$ and $B = B_1 \cup B_2$ are the sets of vertices of G, and E is the set of edges of G. $A_1 = \{\alpha_1, \alpha_2, \cdots, \alpha_{|Ch(x)|}\}$ and $B_1 = \{\beta_1, \beta_2, \cdots, \beta_{|Ch(y)|}\}$. $\alpha_i(\beta_j)$ denotes subtree $T_a(u, x_i)(T_b(v, y_j))(x_i \in Ch(x),\ y_j \in Ch(y))$. A_2 and B_2 are the sets of dummy vertices. If $|Ch(x)| = |Ch(y)|$, then $A_2 = B_2 = \emptyset$(the empty set). If $|Ch(x)| < |Ch(y)|$, then $A_2 = \{\alpha_{|Ch(x)|+1}, \alpha_{|Ch(x)|+2}, \cdots, \alpha_{|Ch(y)|}\}$ and $B_2 = \emptyset$. If $|Ch(x)| > |Ch(y)|$, then $A_2 = \emptyset$ and $B_2 = \{\beta_{|Ch(y)|+1}, \beta_{|Ch(y)|+2}, \cdots, \beta_{|Ch(x)|}\}$. Let e_{ij} denote the edge that is incident with α_i and β_j, and $E = \{e_{ij} | i = 1, 2, \cdots, |A|,\ j = 1, 2, \cdots, |B|\}$ denote the set of edges that are incident with vertices of A and that of B. Let $w(e_{ij})$ denote the weight of edge e_{ij} and be defined as follows.

$$w(e_{ij}) = \begin{cases} \hat{D}(u, x_i, v, y_j) & : \alpha_i \in A_1,\ \beta_j \in B_1, \\ |V_a(u, x_i)| \cdot r & : \alpha_i \in A,\ \beta_j \in B_2, \\ 0 & : \alpha_i \in A_2,\ \beta_j \in B. \end{cases}$$

A matching $\mathcal{M}$ of G is *perfect* if each vertex is incident with exactly one member of $\mathcal{M}$. From formulae (13) and (14), we can see that a perfect matching of G, denoted by $\mathcal{M}$, can be expressed by a C_{uxvy}. Let $\mathcal{M}_{min}$ denote one of the minimum weight perfect matchings of G, and $W(\mathcal{M}_{min})$ denote its weight. For $\Delta 1_{uxvy}$, we have Lemma 3.

Lemma 3 Assume that all of the distances $\hat{D}(u, x_i, v, y_j)(x_i \in Ch(x),\ y_j \in Ch(y))$ are given. We can compute $\Delta 1_{uxvy}$ by the following formula.

$$\Delta 1_{uxvy} = \delta(x,y) + W(\mathcal{M}_{min}). \quad (19)$$

□

5 An Algorithm for LSS in *R-trees*

Let $N_{xy} = |S_{xy}|$, and let $\hat{N}_{xy}$ denote "$\max_{M_{xy}} \left\{ |S_{M_{xy}}| + |\hat{An}(y, r_{xy})| \right\}$", where M_{xy} is an MCM that determines $\hat{D}(u, x, v, y)$. Let $N1_{xy}$ denote "$\max_{M_{xy}} \{|S_{M_{xy}}|\}$", where M_{xy} is an MCM that determines $\Delta 1_{uxvy}$. Consider Figure 2. Assume that all of $\hat{D}(u, x_i, v, y_j)$ and $N_{x_i y_j}(x_i \in Ch(x),\ y_j \in Ch(y))$ are given. Using Lemma 3, we can compute $\Delta 1_{uxvy}$ and find one of MCMs that determine $\Delta 1_{uxvy}$. To compute $N1_{xy}$, it is necessary to know all M_{xy} that determine $\Delta 1_{uxvy}$. We show an efficient computing method for $\Delta 1$ and $N1_{xy}$.

Lemma 4 Assume that all of $\hat{D}(u, x_i, v, y_j)$ and $\hat{N}_{x_i y_j}(x_i \in Ch(x),\ y_j \in Ch(y))$ are given. We can compute $\Delta 1_{uxvy}$ and $N1_{xy}$ by the following formulae.

$$\Delta 1_{uxvy} = \delta(x,y) + \sum_{(x_i,y_j)\in C'_{uxvy}} \hat{D}(u,x_i,v,y_j) + \sum_{(x_i,y_j)\notin C'_{uxvy}} |V_a(u,x_i)| \cdot r, \quad (20)$$

$$N1_{xy} = 2 + \sum_{(x_i,y_j)\in C'_{uxvy}} \hat{N}_{x_i y_j}, \quad (21)$$

where C'_{uxvy} satisfies the following formula.

$$\begin{aligned}
&\sum_{(x_i,y_j)\in C'_{uxvy}} \Big(\big(\hat{D}(u,x_i,v,y_j)+1\big) \\
&\cdot (|V_a(u,x)| + |V_b(v,y)|) - \hat{N}_{x_i y_j} \Big) \\
&+ \sum_{(x_i,y_j)\notin C'_{uxvy}} |V_a(u,x_i)| \cdot r \\
&\cdot (|V_a(u,x)| + |V_b(v,y)|) \\
&= \min_{C_{uxvy}} \Big\{ \sum_{(x_i,y_j)\in C_{uxvy}} (\\
&\big(\hat{D}(u,x_i,v,y_j)+1\big) \\
&\cdot (|V_a(u,x)| + |V_b(v,y)|) - \hat{N}_{x_i y_j} \Big) \\
&+ \sum_{(x_i,y_j)\notin C_{uxvy}} |V_a(u,x_i)| \cdot r \\
&\cdot (|V_a(u,x)| + |V_b(v,y)|) \Big\}. \quad (22)
\end{aligned}$$

□

Using the algorithm for finding one of the minimum weight perfect matchings of G[13], we can find C'_{uxvy} in formula (22). Using Lemma 4, we can compute $\Delta 1_{uxvy}$ and $N1_{xy}$. Let LSS-Δ1 denote this procedure. The time and space complexities of LSS-Δ1 are $O_T(m^3)$ and $O_S(m^2)$, respectively, where $m = \max(m_a, m_b)$. Using LSS-Δ1, Lemma 1 and Lemma 2, we can compute $D(u,x,v,y)$, $\hat{D}(u,x,v,y)$, N_{xy} and one of MCMs that determin $\mathbf{S}_{xy}$. Since it is easy to find the vertices of $\mathbf{S}_{xy}$, we will omit the details. A procedure LSS-subtree for computing $D(u,x,v,y)$, $\hat{D}(u,x,v,y)$ and N_{xy} is shown in Figure 4. The time complexity of LSS-subtree is $O_T(m^3)$.

Let $N_{uv} = |S_{uv}|$, and let $N_a(N_b)$ be the number of vertices of $T_a(T_b)$. Using procedure LSS-subtree, we can get an algorithm LSS-*R-tree* for finding one of the LSSs in *R-trees* which is shown in Figure 5. The time complexity of LSS-*R-tree* is $O_T(m^3 N_a N_b)$. Since the space for recording C'_{uxvy} for all vertices x of $T_a(u)$ and all vertices y of $T_b(v)$ is at most $2mN_aN_b$, the space complexity of LSS-*R-tree* is $O_S(mN_aN_b)$.

```
Procedure LSS-subtree(u, x, v, y);
begin
  compute Δ1_uxvy and N1_xy using LSS-Δ1;
  compute D̂(u, x, v, y) by formula (18);
  compute D(u, x, v, y) by formula (8);
  if D(u, x, v, y) = Δ1_uxvy, then
    N_xy := N1_xy
  else
    if D(u, x, v, y) = Δ2_uxvy, then
      N_xy := N_{x_i y}
    else
      N_xy := N_{x y_j};
  if D(u, x, v, y) = Δ2_uxvy and N_xy < N_{x_i y}, then
    N_xy := N_{x_i y};
  if D(u, x, v, y) = Δ3_uxvy and N_xy < N_{x y_j}, then
    N_xy := N_{x y_j};
end.
```

Figure 4: Procedure LSS-subtree.

```
Algorithm LSS-R-tree
Input  : Two R-trees T_a(u) and T_b(v)
Output : N_uv

begin
  D(u, v) := N_a · r + N_b · q;
  for vertex y in T_b(v) do
  {y is chosen in decreasing order of the depth of y}
  for vertex x in T_a(u) do
  {x is chosen in decreasing order of the depth of x}
  begin
    N_xy := 1;   N1_xy := 1;
    if both x and y are leaves, then
    begin
      D̂(u, x, v, y) := δ(x, y);   Δ1_uxvy := δ(x, y);
      D(u, x, v, y) := δ(x, y);
    end
    else
      compute D̂(u, x, v, y), D(u, x, v, y) and N_xy
        using LSS-subtree;
  end;
  D(u, v) := D(u, u, v, v);  N_uv := N_xy;
end.
```

Figure 5: Algorithm LSS-*R-tree*.

6 An Algorithm for LSS in *Trees*

If all of distances $D(u,v)$ and all of $\mathbf{S}_{uv}$ are given, for T_a and T_b, we can compute D and find $\mathbf{S}$. In Ref. [8], an efficient algorithm for the distance between *CO-trees* was proposed. Modifing this algorithm, we can obtain an algorithm LSS-*tree* for finding one of the LSSs of *trees*.

(c1) Replace the notations concerning *RO-tree*(*CO-tree*) with *R-tree*(*tree*) defined in this paper.

(c2) Replace the notations concerning the dis-

tances based on SSPM with those concerning the distances between similar substructures.

(c3) Replace the procedure for computing the distance between subtrees with procedure LSS-subtree.

(c4) Add the instructions for computing $|S|$.

The time complexity of LSS-*tree* is $O_T(m^3N_aN_b)$. Since the space for recording C_{uxvy} for all vertices x of $T_a(u)$ and all vertices y of $T_b(v)$ is at most mN_aN_b, the space complexity of LSS-*tree* is $O_S(mN_aN_b)$.

7 Conclusions

We discussed the problems of largest similar substructures in T_b to T_a, where both T_a and T_b are *R-trees*(*trees*). An efficient algorithm for finding one of the LSSs in *R-trees* and that in *trees* were proposed. The time and space complexities of both algorithms are $O_T(m^3N_aN_b)$ and $O_S(mN_aN_b)$, respectively. Those algorithms can be applied to structure-activity studies and structure comparison problems. One of the future problems is to enumerate all subtrees of tree T_b with at most k differences to tree T_a.

References

[1] P.Willett, "Similarity and clustering in chemical information systems," *Research Studies Press*, 1987.

[2] M.G.Hicks and C.Jochum, "Substructure search system. 1. Performance comparison of the MACCS, DARC, HTSS, CAS registry MVSSS, and S4 substructure search systems," *J. Chem. Inf. Comput. Sci.*, vol.**30**, pp.191-199, 1990.

[3] S.M.Liu, E.Tanaka and S.Masuda, "Largest similar substructure problems for treer and their algorithms," *IEICE Trans.*, 1995 (to appear).

[4] K.C.Tai, "The tree-to-tree correction problem," *JACM*, vol.**26**, no.3, pp.422-433, 1979.

[5] E.Tanaka and K.Tanaka, (a)"A tree metric and its computing method," *IECE Trans.*, vol.**J65-D**, no.5, pp.511-518, 1982. (b)"Correction to 'A tree metric and its computing method'," *IEICE Trans.*, vol.**J76-D-I**, no.11, p.635, 1993.

[6] E.Tanaka, "The metric between rooted and ordered trees based on strongly structure preserving mapping and its computing method," *IECE Trans.*, vol.**J67-D**, no.6, pp.722-723, 1984.

[7] K.Ohmori and E.Tanaka, "A unified view on tree metrics," *Proc. Workshop on Syntactic and Structural Pattern Recognition, Barcelona*, pp.85-100, 1986. (Eds. G. Ferraté et al. Syntactic and Structural Pattern Recognition, Springer 1988.)

[8] S.M.Liu, E.Tanaka and S.Masuda, "The distances between unrooted and cyclically ordered trees and their computing methods," *IEICE Trans. Inf. & Syst.*, vol.**E77-D**, no.10, pp.1094-1105, 1994.

[9] T.Muguruma, E.Tanaka and S.Masuda, "A metric between unrooted and unordered trees and its top-down computing method," *IEICE Trans. Inf. & Syst.*, vol.**E77-D**, no.5, pp.555-566, 1994.

[10] E.Tanaka, "A metric between unrooted and unordered trees and its bottom-up computing method," *IEEE Trans. Pattern Anal. & Mach. Intell.*, vol.**16**, no.12, pp.1233-1238, 1994.

[11] S.M.Liu and E.Tanaka, "Algorithms for computing the distances between unordered trees," *IEICE Trans.*, 1995 (to appear).

[12] K.Zhang, R.Statman and D.Shasha, "On the editing distance between unordered labeled trees," *Inf. Process. Lett.*, vol.**42**, pp.133-139, 1992.

[13] E.L.Lawler, Combinatorial Optimization: Networks and Matroids, *Holt Rinehalt and Winston, New York, NY*, 1976.