INVITED PAPER *Special Section on Information-Based Induction Sciences and Machine Learning*

Kernel Methods for Chemical Compounds: From Classification to Design

Tatsuya AKUTSU[†a)] ***and*** **Hiroshi NAGAMOCHI**[††], ***Members***

SUMMARY In this paper, we briefly review kernel methods for analysis of chemical compounds with focusing on the authors' works. We begin with a brief review of existing kernel functions that are used for classification of chemical compounds and prediction of their activities. Then, we focus on the pre-image problem for chemical compounds, which is to infer a chemical structure that is mapped to a given feature vector, and has a potential application to design of novel chemical compounds. In particular, we consider the pre-image problem for feature vectors consisting of frequencies of labeled paths of length at most K. We present several time complexity results that include: NP-hardness result for a general case, polynomial time algorithm for tree structured compounds with fixed K, and polynomial time algorithm for $K = 1$ based on graph detachment. Then we review practical algorithms for the pre-image problem, which are based on enumeration of chemical structures satisfying given constraints. We also briefly review related results which include efficient enumeration of stereoisomers of tree-like chemical compounds and efficient enumeration of outerplanar graphs.

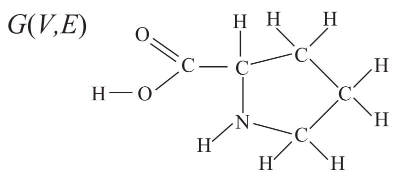
key words: *chemoinformatics, kernel method, pre-image, dynamic programming, enumeration, graph detachment*

1. Introduction

As a result of extensive studies done in these two decades, *kernel methods* have become one of the standard tools in machine learning, data mining, and bioinformatics. Kernel methods have also been applied to chemoinformatics [7], [9], [19], [21], especially to Quantitative Structure-Activity Relationship (QSAR) and Quantitative Structure-Property Relationship (QSPR) problems whose purposes are to predict the chemical activity and property for a given chemical compound respectively [12], [14]. In most of these applications, chemical compounds are usually defined as graph structures and then these graphs are mapped to feature vectors in a feature space, to which such prediction methods as Support Vector Machines (SVMs) and Support Vector Regression (SVR) are applied. Though several methods have been proposed for design of feature vectors, those based on *frequency of small fragments* [7], [9] and *frequency of labeled paths* [19], [21] have been widely used, where weights/probabilities are sometimes put on paths/fragments, and other traditional features such as molecular weights, partial charges and logP might also be combined with them. Examples of feature vectors based on frequency of labeled paths and frequency of small fragments are given in Fig. 1, where the formal definitions are given in Sect. 2.

While classification of chemical compounds and prediction of their activities and properties are still important, design of new chemical compounds is becoming very important because of increasing need of development of novel drugs. In the field of machine learning, the *pre-image problem* has been studied [4], [5]. In this approach, a desired object is specified or computed as a vector in a feature space using a suitable objective function or other methods and then the vector is mapped back to the input space, where this mapped back object is called a *pre-image*. Let ϕ be a mapping from an input space to a feature space. Then, the problem is, given a vector $\mathbf{v}$ in the feature space, to find a pre-image x in the input space such that $\mathbf{v} = \phi(x)$. It is to be noted that ϕ is not necessarily injective or surjective. If ϕ is not surjective, we need to compute an approximate pre-image, for example x^* defined by $x^* = \arg\min_x \ dist(\mathbf{v}, \phi(x))$ (see Fig. 2).

Bakir, Weston and Scölkopf proposed a method to find pre-images in a general setting by using Kernel Principal Component Analysis and regression [4]. Bakir, Zien and Tsuda developed a stochastic search algorithm to find pre-images for graphs [5]. Akutsu and Fukagawa studied the problem of inferring graphs from the frequency of vertex labeled paths of length at most K, which corresponds to a pre-image problem on the path frequency-based feature space [1]–[3]. They proved that this problem can be solved by *dynamic programming* in polynomial time of the size of an output graph if graphs are trees of bounded degree and K



	C	N	O	H	CC	CN	CO	CH	NC	NN	• • •	OH	• • •
$\phi^1_{\mathrm{PF}}(G)$	5	1	2	9	8	2	3	7	2	0	• • •	1	• • •

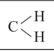
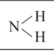
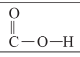
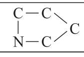
	C	N	O	H					• • •
$\phi_{\mathrm{FF}}(G)$	5	1	2	9	3	0	1	1	• • •

Fig. 1 Path frequency-based feature vector $\phi^K_{\mathrm{PF}}(G)$ and fragment-based feature vector $\phi_{\mathrm{FF}}(G)$.

Manuscript received December 28, 2010.
Manuscript revised April 23, 2011.

[†]The author is with Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji-shi, 611–0011 Japan.
[††]The author is with the Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan.
a) E-mail: takutsu@kuicr.kyoto-u.ac.jp

DOI: 10.1587/transinf.E94.D.1846

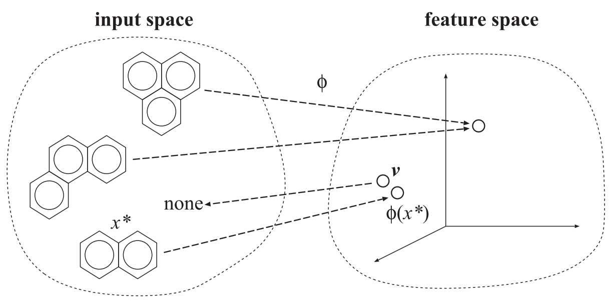


Fig. 2 Pre-image problem for chemical compounds. Multiple compounds may correspond to the same feature vector, or no compound may correspond to a given feature vector.

is fixed, whereas this problem is NP-hard even for trees of bounded degree for general K [1]. They extended the polynomial time algorithm for outerplanar graphs [2] and for feature vectors based on frequencies of small fragments [3], where a graph is *outerplanar* if it can be drawn on a plane such that all vertices lie on the outer face without crossing of edges and it is reported that 94.3% of chemical compounds in the NCI chemical database have outerplanar graph structures [15].

Though these algorithms work in polynomial time, the degree of polynomial is too high to be applied to real instances. Nagamochi developed an efficient polynomial time algorithm for the case of $K = 1$ which can be applied to general graph structures as well as tree structures, based on *graph detachment* [24].

In order to develop practical algorithms, Akutsu and Fukagawa proposed a *branch-and-bound* algorithm for tree-like chemical structures [1]. Then, Fujiwara et al. developed a much more efficient branch-and-bound algorithm [13], which combines an existing tree enumeration algorithm [25] with several bounding operations. Ishida et al. developed an improved algorithm by introducing a novel and strong bounding operation named *detachment cut* [18], which is based on Nagamochi's work on detachment [24].

The pre-image problem has also been studied as a part of *inverse QSAR/QSPR* studies. Kier et al. developed methods for reconstructing molecular structures from the count of paths of a length up to two and the count of paths of a length up to three, by combining enumeration and bounding operations [20]. Skvortsova et al. developed a similar method, in which paths of the same length are further classified into several classes based on atom and bond types [28]. Faulon et al. defined another descriptor based on trees and developed methods for enumerating all the structures consistent with a given descriptor [10].

As mentioned above, enumeration of structures consistent with given constraints plays a key role in practical algorithms for the pre-image problem for chemical compounds. In fact, the enumeration of chemical graphs is one of the fundamental problems in chemoinformatics [11], [12] and has a long history going back to Cayley's work on the enumeration of structural isomers of alkanes in the 19th Century [8] and including seminal group theoretic studies by Polỳa and others [27]. Our approach provides somewhat different methodology to these existing approaches. In addition to enumeration of chemical graphs, it is also important to enumerate stereoisomers. For this stereoisomer enumeration problem, several methods have been proposed, which mostly follow the work by Nourse [26]. The basic strategy employed in these methods is to create a list of all 2^m combinations after identification of m stereocenters and then remove duplicated structures. Imada et al. recently developed an alternative approach using dynamic programming [16], [17] whereas the applicability of this approach is currently limited to chemical graphs having tree-like structures and outerplanar structures.

In this paper, we review some of existing kernels for chemical compounds, algorithms for the pre-image problem for chemical compounds, and algorithms for enumeration of isomers and stereoisomers, with focusing on the authors' works.

2. Kernels for Chemical Compounds

In this section, we briefly review some of existing kernels for chemical compounds. Other recent kernels for chemical compounds can be found in [23].

We begin with feature vectors based on frequency of vertex labeled paths and its probabilistic extension called *marginalized graph kernel* or *random walk kernel*. Let $G(V, E)$ be an undirected connected multigraph without self loops[†]. Let Σ_V and Σ_E be sets of vertex labels and edge labels, respectively. In this paper, we mainly consider vertex labels and ignore edge labels in many cases. Since we are considering chemical structures, we reasonably assume that the maximum degree of vertices and the sizes of Σ_V and Σ_E are bounded by constants. Then, we use $n = |V|$ to denote the size of graph $G(V, E)$ since the number of bonds connecting to each vertex is bounded by a constant and thus $|E|$ is $O(n)$. For each $a \in \Sigma_V$, its valence $val(a)$ is assigned (e.g., $val(C) = 4$, $val(N) = 3$, $val(O) = 2$). Let $\Sigma_V^{\leq k}$ be the set of label sequences (i.e., the set of strings) over Σ_V whose lengths are between 1 and k. Let $\ell(v)$ be the label of vertex v. For each path $\pi = (v_0, \ldots, v_h)$ of G, $\ell(\pi)$ denotes the label sequence of π (i.e., $\ell(\pi) = (\ell(v_0), \ldots, \ell(v_h))$). For graph G and label sequence s, $occ(s, G)$ denotes the number of paths π in G such that $\ell(\pi) = s$. It is to be noted that we consider directed paths for π although G is an undirected graph. Then, the level K feature vector $\phi_{\mathrm{PF}}^K(G)$ for $G(V, E)$ is defined by

$$\phi_{\mathrm{PF}}^K(G) = (occ(s, G))_{s \in \Sigma_V^{\leq K+1}}$$

and the corresponding kernel $\mathcal{K}(G, G')$ between graphs $G(V, E)$ and $G'(V', E')$ is defined by

$$\mathcal{K}_{\mathrm{PF}}^K(G, G') = \phi_{\mathrm{PF}}^K(G) \cdot \phi_{\mathrm{PF}}^K(G')$$

[†]Multigraph means that there can be multiple edges between the same pair of vertices. Though multigraphs were not explicitly considered in [1]–[3], the algorithms can be modified to cope with multigraphs.

where $\mathbf{x} \cdot \mathbf{y}$ denotes the inner product between two vectors $\mathbf{x}$ and $\mathbf{y}$. See Fig. 1 for an example. It should be noted that the size (i.e., number of vertices) n of the original graph can be obtained from $\phi^K_{\mathrm{PF}}(G)$.

The frequency-based kernel can be extended to the *marginalized graph kernel* as follows. For two sequences s and s' over Σ. we define $\mathcal{K}_{\mathrm{ID}}(s, s')$ by

$$\mathcal{K}_{\mathrm{ID}}(s, s') = \begin{cases} 1 & \text{if } s = s', \\ 0 & \text{otherwise.} \end{cases}$$

We assume that each path π in G has the probability $Pr(\pi)$, where $\sum_{\pi \in V^*} Pr(\pi) = 1$ holds. This probability is usually given as the probability of generating π by random walk on G under some probabilistic model. Then, the marginalized graph kernel $\mathcal{K}_{\mathrm{MG}}(G, G')$ is defined by

$$\mathcal{K}_{\mathrm{MG}}(G, G') = \sum_{(\pi,\pi') \in V^* \times (V')^*} Pr(\pi) Pr(\pi') \mathcal{K}_{\mathrm{ID}}(\pi, \pi').$$

Though the number of paths appearing in this kernel is infinite, the kernel value can be computed efficiently by matrix inversion under a certain probabilistic model of random walk [19]. In the following, we assume that *tottering paths* (paths for which there exists some i such that $v_i = v_{i+2}$) are not counted in feature vectors, where it is suggested that avoiding tottering paths is useful in practice [21].

Next we briefly review feature vectors based on small fragments, which have been traditionally used in chemoinformatics (see also Fig. 1). Let $\mathcal{F} = \{F_1, \ldots, F_M\}$ be a set of graphs (chemical substructures). Since information on the number of occurrences of each atom type is usually included in feature vectors, we assume that all single atoms are included in $\mathcal{F}$. We also assume that the size of each F_i is bounded by a constant K because small fragments are usually employed. Let $occ(F_i, G)$ denote the number of subgraphs of G that are isomorphic to F_i. Different from the case of path frequency, subgraphs consisting of the same vertices are counted only once for each F_i because the number of automorphisms may become large whereas it is at most two for each path. Then, a feature vector $\phi_{\mathrm{FF}}(G)$ for G is defined by

$$\phi_{\mathrm{FF}}(G) = (occ(F_i, G))_{F_i \in \mathcal{F}}.$$

The kernel function for this feature vector is simply defined as the inner product $\phi_{\mathrm{FF}}(G) \cdot \phi_{\mathrm{FF}}(G')$.

As a variant of path frequency-based kernels and fragment-based kernels, *tree pattern kernels* have been proposed [22]. Let $\mathcal{T} = \{T_1, \ldots, T_{|\mathcal{T}|}\}$ be a set of trees. Let $occ(T, G)$ denote the number of occurrences of T in G (see [22] for the meaning of *occurrences*). Assume that *weight* $w(T)$ is given for each tree. Then, the tree-pattern kernel is defined as

$$\mathcal{K}_{\mathrm{TR}}(G, G') = \sum_{T \in \mathcal{T}} w(T) occ(T, G) occ(T, G').$$

It is shown in [22] that $\mathcal{K}_{\mathrm{TR}}(G, G')$ can be computed in polynomial time if $\mathcal{T}$ and $w(T)$ satisfy some reasonable conditions.

Chemical compounds are regarded as undirected labeled graphs in the above. However, two chemical compounds with the same graph structure may have different three-dimensional configurations due to asymmetry around carbon atoms and many other structural asymmetries. Such compounds are called *stereoisomers*. Since stereoisomers often exhibit different chemical properties, it is also important to develop kernel functions incorporating such stereochemical information. Brown et al. developed such a kernel based on the tree pattern kernel [6].

3. Dynamic Programming Algorithms for Pre-image Problem

As mentioned in Introduction, the pre-image problem is to find a graph $G(V, E)$ such that $\phi(G) = \mathbf{v}$ for a given feature vector $\mathbf{v}$ under some fixed feature map ϕ. The most basic version of the pre-image problem is defined as follows.

Definition 1: (**GIPF**: Graph Inference from Path Frequency)
Given a path frequency-based feature vector $\mathbf{v}$ of level K, output a graph $G(V, E)$ satisfying $\phi^K_{\mathrm{PF}}(G) = \mathbf{v}$. If there does not exist such $G(V, E)$, output "no solution."

We also consider several variants of GIPF by taking into account the valence condition, the distance between a given vector and a possible feature vector, and fragment-based features[†].

GIFV (Graph Inference from path Frequency and label Valence)
Given a path frequency-based feature vector $\mathbf{v}$ of level K, output a graph $G(V, E)$ satisfying $\phi^K_{\mathrm{PF}}(G) = \mathbf{v}$ and $\sum_{w:\{v,w\} \in E} m(\{v, w\}) = val(\ell(v))$ for all $v \in V$, where m denotes the multiplicity of an edge $\{v, w\}$ (e.g., $m(\{v, w\}) = 2$ if there exist two edges (i.e., double bond) between v and w). If there does not exist such $G(V, E)$, output "no solution."

GIFV-M (Graph Inference from path Frequency and label Valence with Minimum distance)
Given a path frequency-based feature vector $\mathbf{v}$ of level K, output a graph $G(V, E)$ which minimizes L_1-distance between $\phi^K_{\mathrm{PF}}(G)$ and $\mathbf{v}$ under the condition that $\sum_{w:\{v,w\} \in E} m(\{v, w\}) = val(\ell(v))$ holds for all $v \in V$.

GIFF (Graph Inference from Fragment Frequency)
Given a feature vector $\mathbf{v}$ based on a set of fragments $\mathcal{F}$, output a graph $G(V, E)$ satisfying $\phi_{\mathrm{FF}}(G) = \mathbf{v}$ and $\sum_{w:\{v,w\} \in E} m(\{v, w\}) = val(\ell(v))$ for all $v \in V$. If there does not exist such $G(V, E)$, output "no solution."

GIFF-M (Graph Inference from Fragment Frequency with Minimum distance)
Given a feature vector $\mathbf{v}$ based on a set of fragments $\mathcal{F}$, output a graph $G(V, E)$ which minimizes L_1-distance between $\phi_{\mathrm{FF}}(G)$ and $\mathbf{v}$ under the condition that $\sum_{w:\{v,w\} \in E} m(\{v, w\}) = val(\ell(v))$ holds for all $v \in V$.

[†]The names of some problems are changed from the original ones [3].

GIULF (Graph Inference from Upper and Lower bounds of Fragment frequency)
Given feature vectors ***ub*** and ***lb*** based on a set of fragments $\mathcal{F}$, output a graph $G(V,E)$ satisfying $\boldsymbol{lb} \leq \phi_{\mathrm{FF}}(G) \leq \boldsymbol{ub}$ and $\sum_{w:\{v,w\}\in E} m(\{v,w\}) \leq val(l(v))$ for all $v \in V$, where $\mathbf{x} \leq \mathbf{y}$ denotes that $x_i \leq y_i$ holds for any i-th elements of $\mathbf{x}$ and $\mathbf{y}$. If there does not exist such $G(V,E)$, output "no solution."

Interestingly, all of the above mentioned problem can be solved in polynomial time for reasonably wide classes of constraints and graph structures [1]–[3]

Theorem 1: GIPF, GIFV, GIFV-M, GIFF, GIFF-M and GIULF for outerplanar graphs can be solved in polynomial time in n if K, M and Σ are fixed, and the number of edges of each face and the maximum degree of graphs are bounded by constants.

On the other hand, it is known that the most basic version GIPF is NP-hard even for trees of unbounded degree [1], [2]

Theorem 2: GIPF is strongly NP-hard (i) even for $K = 3$ and trees of unbounded degree, and (ii) even for trees of bounded degree and for a fixed Σ.

Here, we briefly present the basic idea of the above mentioned polynomial time algorithms, all of which use dynamic programming. We consider a simple case where $\Sigma = \{0,1\}$, $K = 1$, and $m(e) = 1$ holds for any edge e. We construct a dynamic programming table $D(\ldots)$ defined by

$$D(n_0,n_1,n_{00},n_{01},n_{10},n_{11}) = \begin{cases} 1, & \text{if there exists tree } T \text{ such that} \\ & \quad \phi^1_{\mathrm{PF}}(T) = (n_0,n_1,n_{00},n_{01},n_{10},n_{11}), \\ 0, & \text{otherwise,} \end{cases}$$

where n_i and n_{ij} denote the numbers of vertices (i.e., paths of length 0) having label i and edges (i.e., paths of length 1) having vertex labels i, j (i.e., each edge is directed from vertex labeled i to vertex labeled j), respectively. This table can be constructed by the following procedure where we omit the initialization part:

$$\begin{aligned} &D(n_0,n_1,n_{00},n_{01},n_{10},n_{11}) = 1 \textbf{ iff} \\ &\quad D(n_0-1,n_1,n_{00}-2,n_{01},n_{10},n_{11}) = 1 \quad \textbf{or} \\ &\quad D(n_0-1,n_1,n_{00},n_{01}-1,n_{10}-1,n_{11}) = 1 \quad \textbf{or} \\ &\quad D(n_0,n_1-1,n_{00},n_{01}-1,n_{10}-1,n_{11}) = 1 \quad \textbf{or} \\ &\quad D(n_0,n_1-1,n_{00},n_{01},n_{10},n_{11}-2) = 1. \end{aligned}$$

The correctness of the algorithm follows from the fact that any tree can be constructed incrementally by adding a vertex (leaf) one by one. For example, the second and third lines in this procedure correspond to cases where a new vertex labeled 0 is attached to an existing vertex labeled 0 and a new vertex labeled 0 is attached to an existing vertex labeled 1, respectively. The required tree (if exists) can be obtained by means of a standard *traceback* procedure. Since the value of each element of the feature vector is $O(n)$, the table size is $O(n^6)$. Since it takes a constant time per entry to construct the dynamic programming table, we can see that the computation time is $O(n^6)$.

As seen from this analysis, the degree of polynomial is quite high in the above mentioned dynamic programming-based algorithms. Therefore, the polynomial time results in this section are not practical and thus alternative ways are required as mentioned in the following sections.

4. Detachment Algorithms for Level One GIFV

In this section, we review that GIPF and GIFV for general graphs admit efficient algorithms if $K = 1$. For a subset $X \subseteq E$ (resp., $X \subseteq V$) of a multigraph $G(V,E)$, let $G - X$ denote the multigraph obtained by removing the edges in X (resp., the vertices in X together with the incident edges) from G. Let $comp(G)$ denote the number of connected components in G.

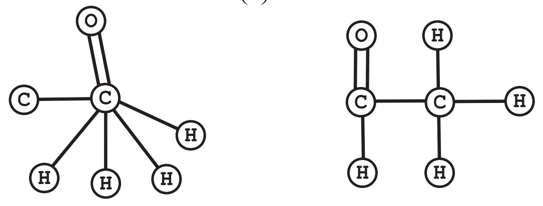
Let $\mathbf{v} = \mathbf{v}^1 + \mathbf{v}^2$ be a given path frequency-based feature vector of level $K = 1$, where $\mathbf{v}^i$ ($i = 1,2$) denotes the vector consisting of the elements for label sequences of length i in $\mathbf{v}$. Let n (n') denote the sum of all elements in $\mathbf{v}^1$ ($\mathbf{v}^2$). Thus n and n' are the numbers of vertices and edges of a possible solution to GIPF, i.e., a graph G satisfying $\phi^1_{\mathrm{PF}}(G) = \mathbf{v}$.

For an example of feature vector $\mathbf{v}$ in Fig. 3 (a), both graphs G_1 and G_2 in Fig. 3 (b) and (c) are solutions to GIPF. Let us consider what graph will appear from these graphs if we merge all the atoms of the same element into a single vertex. Figure 4 (a) shows such a graph $G_{\mathbf{v}}(\Sigma_V, \mathcal{E})$ resulting from G_1 (or G_2). Note that $G_{\mathbf{v}}$ is a mulitgraph on the vertex set Σ_V such that edges are determined by $\mathbf{v}^2$.

Conversely by splitting each vertex $t \in \Sigma_V$ in $G_{\mathbf{v}}$ into a specified number $\mathbf{v}[t]$ of vertices, we can get a solution G to GIPF such as G_1 and G_2 as long as G is connected and has no self-loops. Such an operation is called *detachment*. To find a detachment that generates a loopless and connected graph G from $G_{\mathbf{v}}$, we construct the expansion H of $G_{\mathbf{v}}$, which is a loopless multigraph obtained from $G_{\mathbf{v}}$ by splitting each vertex $t \in \Sigma_V$ into $\mathbf{v}[t]$ vertices $t^1, t^2, \ldots, t^{\mathbf{v}[t]}$ and putting $\mathbf{v}[st]$ multiple edges between every vertices s^i and t^j. See Fig. 4 (b) for an example of H.

H	O	C	HH	HO	HC	OH	OO	OC	CH	CO	CC
4	1	2	0	0	4	0	0	2	4	2	2

(a) $\mathbf{v}$



(b) G_1 (c) G_2

Fig. 3 (a) A vector $\mathbf{v}$ of level $K = 1$; (b) a solution G_1 to GIPF with $\mathbf{v}$; and (c) a solution G_2 to GIFV with $\mathbf{v}$ and the valence $val(\mathtt{H}) = 1$, $val(\mathtt{O}) = 2$ and $val(\mathtt{C}) = 4$.

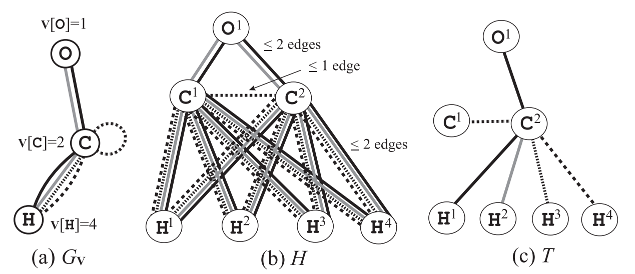


Fig. 4 (a) The multigraph $G_{\mathbf{v}}(\Sigma_V, \mathcal{E})$; (b) the expansion H of $G_{\mathbf{v}}$ with $\mathbf{v}[\mathtt{H}] = 4$, $\mathbf{v}[\mathtt{O}] = 1$ and $\mathbf{v}[\mathtt{C}] = 2$; and (c) a spanning tree T of H such that $\phi^1_{\mathrm{PF}}(T) \le \mathbf{v}^2$.

Now GIPF has a solution G if and only if H contains a spanning tree T whose edge set does not exceed the one specified by $\mathbf{v}^2$ (obviously any spanning tree of G satisfies the condition; on the other hand, given such a tree T, G can be obtained by adding the remaining edges arbitrarily). The example H in Fig. 4 (b) has such a tree T as shown in Fig. 4 (c), in which the number of edges between $\mathtt{O}^1$ and $\{\mathtt{C}^1, \mathtt{C}^2\}$ is at most $\mathbf{v}[\mathtt{OC}] = 2$, that between $\mathtt{C}^1$ and $\mathtt{C}^2$ is at most $\mathbf{v}[\mathtt{CC}] = 1$, and that between $\{\mathtt{C}^1, \mathtt{C}^2\}$ and $\{\mathtt{H}^1, \mathtt{H}^2, \mathtt{H}^3, \mathtt{H}^4\}$ is at most $\mathbf{v}[\mathtt{CH}] = 4$. In this case, $\mathbf{v}[\mathtt{OC}] = 2$ and $occ(\mathtt{OC}, T) = 1$, and we add one more edge to T to obtain a solution G to GIPF, such as G_1 in Fig. 3 (b).

Finally, we convert a solution to GIPF to that to GIFV by modifying the edge-vertex incidence relationship until the valence condition is satisfied. For example, G_2 in Fig. 3 (c) is obtained from G_1 by changing the end-vertex $\mathtt{C}$ of three edges $\mathtt{HC}$ to the other carbon atom $\mathtt{C}$.

A mathematical characterization of vectors $\mathbf{v}$ that admit solutions to GIPF (GIFV) together with polynomial algorithms have been obtained as follows [24].

Theorem 3: GIPF of level 1 has a solution if and only if the following conditions hold:

(1) for every nonempty subset $X \subseteq \Sigma_V$, the number of edges between X and Σ_V in $G_{\mathbf{v}}$ is not smaller than $\sum_{t \in X} \mathbf{v}[t] + comp(G_{\mathbf{v}} - X) - 1$; and
(2) for each $tt \in \Sigma^2_V$ with $\mathbf{v}[tt] \neq 0$, it holds $\mathbf{v}[t] \ge 2$.

GIFV of level 1 has a solution if and only if (1), (2) and the following condition hold:

(3) for each $t \in \Sigma_V$, the number of edges incident to the vertex $t \in \Sigma_V$ in $G_{\mathbf{v}}$ is at least $val(t)$.

Whether GIPF (GIFV) has a solution or not can be tested in $O(\min\{n + |\Sigma_V|^2 2^{|\Sigma|}, n^{3.5} + n'\})$ time, and a solution G to GIPF (GIFV), if any, can be constructed in $O(n^{3.5} + n'n^2)$ time.

The result has been extended to a variant of GIFV with $K = 1$ which requires a graph G satisfying the valence condition, $\phi^0_{\mathrm{PF}}(G) = \mathbf{v}^1$ and $\phi^1_{\mathrm{PF}}(G) \ge \mathbf{v}^2$ [24].

5. Enumeration of Tree-Like Chemical Graphs

In this section, we give a sketch of our branch-and-bound algorithm for enumerating all solutions to GIFV for multitrees [13], [18]. We first observe that the multiplicity $m(\{u, v\})$ of two adjacent vertices u and v in a multitree is uniquely determined due to the valence of vertices. We assume that a given vector $\mathbf{v}$ represents path frequency in a possible *simple* tree, and consider how to compute all solutions to GIFV, i.e., simple trees G satisfying $\phi^K_{\mathrm{PF}}(G) = \mathbf{v}$ and the valence condition.

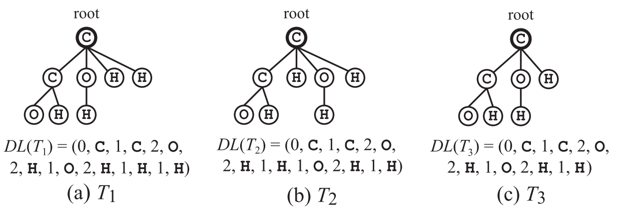

root
root
root
$DL(T_1) = (0, \mathtt{C}, 1, \mathtt{C}, 2, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{H}, 1, \mathtt{H})$
$DL(T_2) = (0, \mathtt{C}, 1, \mathtt{C}, 2, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{H}, 1, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{H})$
$DL(T_3) = (0, \mathtt{C}, 1, \mathtt{C}, 2, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{O}, 2, \mathtt{H}, 1, \mathtt{H})$
(a) T_1
(b) T_2
(c) T_3

Fig. 5 Ordered trees and their depth-label sequences, where T_1 is left-heavy, and T_3 is the parent $P(T_1)$ of T_1.

Canonical Forms To avoid generating duplications of the same simple tree, we introduce a canonical form of trees based on "left-heavy trees" due to Nakano and Uno [25].

A *rooted tree* is a tree G with a designated vertex r, called the *root*, which introduces a parent-child relationship among vertices and defines the *depth* $d(v)$ of each vertex v to be the length of the path from r to v. An *ordered tree* is a rooted tree $T = (G, r, \pi)$ with a left-right relationship π, a total order over the children of each vertex. We denote the vertices in T by $v_0, v_1, \ldots, v_{n-1}$ which are indexed by the depth-first search (DFS) that starts from $r = v_0$ and visits vertices from the left to the right. An ordered tree T is encoded into the *depth-label sequence* $DL(T) = (d(v_0), \ell(v_0), d(v_1), \ell(v_1), \ldots, d(v_{n-1}), \ell(v_{n-1}))$, which is an alternating sequence of the depth $d(v)$ and the label $\ell(v)$ of all vertices v_i. Fix a total order over Σ_V. An ordered tree $T = (G, r, \pi)$ is called *left-heavy* if $DL(T)$ is lexicographically larger than that of any other ordered tree $T' = (G, r, \pi')$ of (G, r). Figure 5 (a) and (b) show two ordered trees T_1 and T_2 of the same rooted tree, where T_1 is left-heavy.

To regard "unrooted trees" as rooted trees, we use the fact that every tree G possesses the *centroid*, which is a unique vertex v^*, called the *unicentroid*, or a unique edge e^*, called the *bicentroid*, such that any subtree in $G - v^*$ $(G - e^*)$ contains at most $(n-1)/2$ $(n/2)$ vertices. In what follows, we treat only trees which have the unicentroids and regard them as trees rooted at the unicentroids. The canonical form of a tree G is defined to be the left-heavy tree $T = (G, v^*, \pi)$ of the tree (G, v^*) rooted at the unicentroid v^*.

Branching In a branch-and-bound method, *branching* is an algorithm for generating all candidates for the solutions. Our branching generates all the left-heavy trees with at most n vertices labeled by Σ_V. To generate all left-heavy trees, we introduce a parent-child relationship between left-heavy trees. The *parent* $P(T)$ of a left-heavy tree T is defined to be the ordered tree obtained by removing the *rightmost* leaf of T. Notice that the resulting tree $P(T)$ is also left-heavy. Figure 5 (a) and (c) show a left-heavy tree T_1 and its parent $T_3 = P(T_1)$. Hence all the left-heavy trees T with at most n vertices labeled by Σ_V are connected into a tree structure, called *family tree* $\mathcal{F}(n, \Sigma_V)$ whose leaves correspond to left-heavy trees with exactly n vertices. The left-heavy tree of each tree rooted at its unicentroid with n vertices must correspond to a leaf in the family tree. The task of our branching is to visit all nodes in $\mathcal{F}(n, \Sigma_V)$ in a DFS manner after starting from the empty tree (the root node of $\mathcal{F}(n, \Sigma_V)$). When we visit a child node v of the current node u in $\mathcal{F}(n, \Sigma_V)$, we append a new leaf to an appropriate place on the rightmost path of the left-heavy tree T_u corresponding to u to generate a left-heavy tree T_v. It is shown that traversing a link between two nodes in $\mathcal{F}(n, \Sigma_V)$ can be executed in *constant* time [25].

Bounding In a branch-and-bound method, *bounding* implies several procedures for discarding part of process for producing candidates that do not lead to any solutions. Our bounding operation skips the task of appending leaves to the current left-heavy tree T with at most n vertices if at least one of the following criteria is violated:

(1) The root of T remains the unicentroid of an output (the centroid constraint);
(2) $\phi^K_{\mathrm{PF}}(T) \leq \mathbf{v}$ (the feature vector constraint);
(3) $\sum_{w:\{v,w\}\in E(T)} m(\{v, w\}) \leq val(\ell(v))$ for all $v \in V(T)$ (the valence constraint);
(4) T can be extended to a connected and loopless tree with n vertices (the detachment constraint).

Testing whether criterion (4) holds or not can be checked by the algorithms in Theorem 3.

It is our future work to extend our branch-and-bound algorithm to a wider class of graphs than multitrees. Recently we have developed an algorithm for generating all rooted outerplanar graphs with at most n vertices in constant time per output [29].

6. Enumeration of Stereoisomers of Tree-Like Chemical Graphs

In this section, we describe an outline of our algorithm for enumerating all stereoisomers of a given tree-like chemical compound composed of carbon, hydrogen, oxygen and nitrogen atoms, where a tree-like chemical compound is modeled as a multitree [16], and it was further extended for chemical compounds having outerplanar graph structures [17]. The algorithm generates each stereoisomer in linear time and space. We assume that stereoisomers in a multitree G arise from the three-dimensional configurations around "asymmetric" carbon atoms (or double bonds between them). Informally, a carbon atom v is said to be *asymmetric* if one of the following cases occurs:

(i) v is adjacent to four subtrees T_i, $i = 1, 2, 3, 4$, and their configurations are all distinct;
(ii) v is adjacent to a subtree T_1 by a double bond and two subtrees T_2 and T_3 by single bonds, the configuration of T_1 is asymmetric along the double bond, and the configurations of T_2 and T_3 are distinct to each other; and

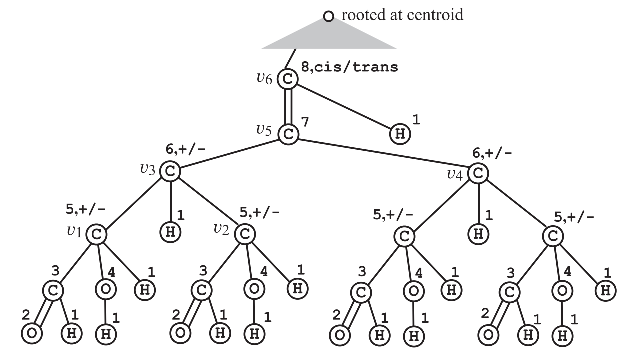


Fig. 6 A multitree rooted at the centroid, where the number beside each vertex v shows the index $id(v)$ of T_v, and the symbol `+/-` (`cis/trans`) indicates that the corresponding carbon atom (double bond) can be asymmetric.

(iii) v is adjacent to two subtrees T_1 and T_2 by double bonds, and the configuration of T_i is asymmetric along the double bond for each $i = 1, 2$.

We regard a given multitree $G(V, E)$ as a tree rooted at its centroid. For each non-root vertex $v \in V$, let T_v denote the tree rooted at v induced from G by the descendants of v. Our algorithm consists of two phases. The first phase counts the total number $f^*(G)$ of stereoisomers of G by using dynamic programming. For a given number $i \in \{1, 2, \ldots, f^*(G)\}$, the second phase constructs the i-th stereoisomer by backtracking the computation process in the first phase. We illustrate the two phases using an example in Fig. 6.

Counting Phase For each non-root carbon atom v, one of the subtrees adjacent to v contains the centroid and its configuration is always distinct from that of any of the other subtrees. Hence we only need to examine whether the subtrees in the tree T_v can take distinct configurations or not. Let $g(v)$ (resp., $h(v)$) denote the number of combinations of configurations of subrees in tree T_v so that v (or a double bond incident to v) becomes symmetric (resp., asymmetric). Then the number $f(v)$ of stereoisomers of T_v is given as $f(v) = g(v) + 2h(v)$. Counting phase computes the numbers $g(v)$, $h(v)$ and $f(v)$ for all carbon atoms v in a bottom-up manner along tree G. As a preprocessing, we first compute the index $id(v)$ of T_v for all vertices $v \in V$ such that $id(u) = id(v)$ if and only if T_v and T_u are structurally isomorphic without considering any three-dimensional information (see Fig. 6 for an example of id).

In Fig. 6, the three subtrees in tree T_{v_1} rooted at carbon atom v_1 have structurally distinct configurations (i.e., distinct indices id), and v_1 is always asymmetric. Hence $g(v_1) = 0$, $h(v_1) = 1$ and $f(v_1) = g(v_1) + 2h(v_1) = 2$. Similarly, we have $g(v_2) = 0$, $h(v_2) = 1$ and $f(v_2) = g(v_2) + 2h(v_2) = 2$. For carbon atom v_3, its rooted tree T_{v_3} has two subtree T_{v_1} and T_{v_2}, which have the structurally same configuration $id(T_{v_1}) = id(T_{v_2}) = 5$. But T_{v_1} can take two sterically distinct configurations, say $T^+_{v_1}$ and $T^-_{v_1}$. Similarly $T^+_{v_2}$ and $T^-_{v_2}$ for T_{v_2}. Hence there are three combinations of them, i.e., $\{T^+_{v_1}, T^+_{v_2}\}$, $\{T^-_{v_1}, T^-_{v_2}\}$ and $\{T^+_{v_1}, T^-_{v_2}\}$, where the first two give $g(v_3) = f(v_1) = 2$ and the last one shows $h(v_3) = \binom{f(v_1)}{2} = 1$. Hence $f(v_3) = g(v_3) + 2h(v_3) = 4$, which indicates the four stereoisomers around v_3, determined by $(T^+_{v_1}, T^+_{v_2})$, $(T^-_{v_1}, T^-_{v_2})$, $(T^+_{v_1}, T^-_{v_2})$ and $(T^-_{v_1}, T^+_{v_2})$, respectively. Analogously, we have $g(v_4) = 2$, $h(v_4) = 1$ and $f(v_4) = g(v_4) + 2h(v_4) = 4$.

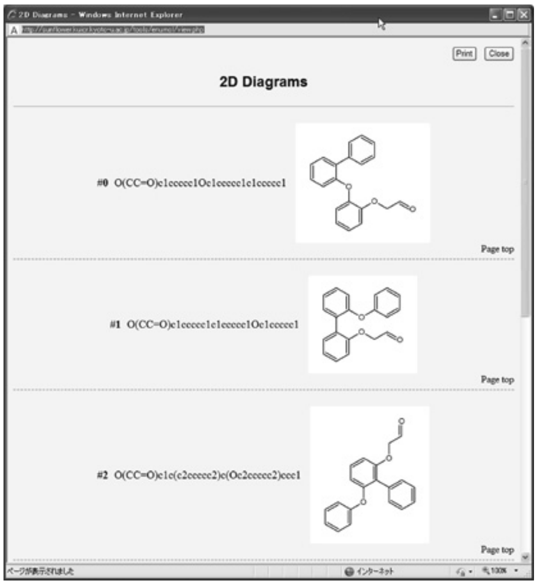
For the double bond between carbon atoms v_5 and v_6, we let $g(v_5)$ (resp., $h(v_5)$) store the number of combinations of configurations in tree T_{v_5} so that no cis-trans isomer (resp., a cis-trans isomer) arises around the double bond. Since T_{v_3} and T_{v_4} have the structurally same configuration, we have $h(v_5) = \binom{f(v_3)}{2} = 6$ and $g(v_5) = f(v_3) = 4$. We set $f(v_6) = g(v_5) + 2h(v_5) = 16$, which shows the number of all cis-trans isomers around the double bond between v_6 and v_5.

Output Phase After the first phase, $f(v), g(v)$ and $h(v)$ are stored for each non-root vertex v. For each number $i = 1, 2, \ldots, f^*(G)$, the second phase outputs the i-th stereoisomer of G by backtracking the process of computing these functions f, g and h in a top-down manner along tree G. When we visit a carbon atom v, we wish to compute the k-th stereoisomer $T^{(k)}_v$ of the subtree T_v for a specified number $k \in \{1, 2, \ldots, f(v)\}$. For this, we detect the integer k_u for each child u of v such that $T^{(k)}_v$ consists of the k_u-th stereoisomer of T_u for all children u of v. We repeat this process until an appropriate configuration of T_w is determined for all carbon atoms w in G.

We explain such a computation using the example in Fig. 6. Suppose that we wish to compute the k-th stereoisomer of T_{v_3}, where $k \in \{1, 2, 3, 4 = f(v_3)\}$. Recall that the four stereoisomers of T_{v_3} are determined by a combination of configurations of T_{v_1} and T_{v_2}, where we call $T^+_{v_1}$ and $T^-_{v_1}$ the first and second stereoisomers of T_{v_1}. Similarly for the first $T^+_{v_2}$ and second $T^-_{v_2}$ of T_{v_2}. We here fix a mapping μ from $\{1, 2, 3, 4 = f(v_3)\}$ to $\{1, 2 = f(v_1)\} \times \{1, 2 = f(v_2)\}$ such as $\mu(1) = (1, 1)$, $\mu(2) = (2, 2)$, $\mu(3) = (1, 2)$ and $\mu(4) = (2, 1)$, where $\mu(k_{v_3}) = (k_{v_1}, k_{v_2})$ means that the k-th stereoisomer of T_{v_3} is composed of the k_{v_1}-th stereoisomer of T_{v_1} and the k_{v_2}-th stereoisomer of T_{v_2}. To represent the symmetry of carbon atom v_1, we set a label of v_1 to be $[id(v_1), \texttt{nil}]$ for $\mu(1) = (1, 1)$ and $\mu(2) = (2, 2)$, $[id(v_1), +]$ for $\mu(3) = (1, 2)$, and $[id(v_1), -]$ for $\mu(4) = (2, 1)$, respectively. Similarly the label of carbon atom v_6 in Fig. 6 is set to be one of $[id(v_6), \texttt{nil}]$, $[id(v_6), \texttt{cis}]$ and $[id(v_6), \texttt{trans}]$ depending on the type of a specified configuration of T_{v_6}.

7. Conclusion

In this paper, we have reviewed kernel functions for chemical graphs and algorithms for pre-image and enumeration problems for chemical structures. In order to make the developed enumeration algorithms easily available, we have been developing the EnuMol system (see Fig. 7) [18]. Currently, EnuMol includes algorithms for enumeration of tree-like chemical structures from path frequency-based feature vectors and enumeration of their stereoisomers. Though the current version of EnuMol can handle benzene rings, each benzene ring is treated as a vertex in a chemical graph.



2D Diagrams

#0 O(CC=O)c1ccccc1Oc1ccccc1c1ccccc1

#1 O(CC=O)c1ccccc1c1ccccc1Oc1ccccc1

#2 O(CC=O)c1c(c2ccccc2)c(Oc2ccccc2)ccc1

Fig. 7 Snapshot of EnuMol.

Although kernel methods provided a new approach to QSAR/QSPR, it seems that their performances are not significantly better than those of traditional descriptor-based approaches. Therefore, further improvements are required.

For enumeration problems, we have shown novel approaches: branch-and-bound algorithms with strong cut operations for enumeration of chemical graphs, and dynamic programming-based algorithms for enumeration of stereoisomers. Though the theoretical and practical performances are better than existing algorithms, the classes of graphs covered by these algorithms are limited. Therefore, extension of algorithms to wider graph classes is left as future work.

For the pre-image problem on chemical compounds, we presented a dynamic programming-based approach and an enumeration-based approach. It seems that the latter approach is much more practical than the former approach. However, unless strong constraints are given, the number of possible structures would be quite large and thus it would be impossible to enumerate all possible structures. In such a case, it would be useful to sample non-similar structures instead of enumerating all possible structures. Therefore, algorithms for sampling non-similar structures should be developed whereas there exist several studies in chemoinformatics [11], [12]. Though we have developed enumeration-based methods for the pre-image problem from path frequency, we have not yet developed such methods for fragment frequency. Therefore, development of enumeration-based methods for fragment frequency is left as future work. Another important future work is to develop efficient methods for specifying a desirable compound in a feature space. Again, some heuristic methods have been proposed in chemoinformatics [11], [12]. However, existing methods can only be applied to design of small compounds and thus novel methods should be developed. For this design problem, machine learning approaches could be useful and should be explored.

Acknowledgments

This work was partially supported by Grants-in-Aid #19200022 and #22240009 from Mext, Japan.

References

[1] T. Akutsu and D. Fukagawa, "Inferring a graph from path frequency," Lect. Notes Comput. Sci., no.3537, pp.371–392, 2005.

[2] T. Akutsu and D. Fukagawa, "On inference of a chemical structure from path frequency," Proc. 2005 International Joint Conference of InCoB, AASBi and KSBI, pp.96–100, 2005.

[3] T. Akutsu and D. Fukagawa, "Inferring a chemical structure from a feature vector based on frequency of labeled paths and small fragments," Proc. 5th Asia Pacific Bioinformatics Conference, pp.165–174, 2007.

[4] G.H. Bakir, J. Weston, and B. Schölkopf, "Learning to find preimages," Advances in Neural Information Processing Systems, vol.16, pp.449–456, 2003.

[5] G.H. Bakir, A. Zien, and K. Tsuda, "Learning to find graph preimages," Lect. Notes Comput. Sci., no.3175, pp.253–261, 2004.

[6] J.B. Brown, T. Urata, T. Tamura, M.A. Arai, T. Kawabata, and T. Akutsu, "Compound analysis via graph kernels incorporating chirality," J. Bioinformatics and Computational Biology, vol.8, suppl. 1, pp.63–81, 2010.

[7] E. Byvatov, U. Fechner, J. Sadowski, and G. Schneider, "Comparison of support vector machine and artificial neural network systems for drug/nondrug classification," J. Chemical Information and Computer Sciences, vol.43, pp.1882–1889, 2003.

[8] A. Cayley, "On the mathematical theory of isomers," Philosophical Magazine, Series 5, vol.47, pp.444–446, 1874.

[9] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent substructure-based approaches for classifying chemical compounds," IEEE Trans. Knowl. Data Eng., vol.17, no.8, pp.1036–1050, 2005.

[10] J-L. Faulon, C.J. Churchwell, and D.P. Visco, Jr., "The signature molecular descriptor. 2. Enumerating molecules from their extended valence sequences," J. Chemical Information and Computer Science, vol.43, pp.721–734, 2003.

[11] J-L. Faulon, D.P. Visco Jr., and D. Roe, "Enumerating molecules," Reviews in Computational Chemistry, vol.21, pp.209–286, 2005.

[12] J-L. Faulon and A. Bender eds., Handbook of Chemoinformatics Algorithms, CRC Press, Boca Raton, FL, 2010.

[13] H. Fujiwara, J. Wang, L. Zhao, H. Nagamochi, and T. Akutsu, "Enumerating tree-like chemical graphs with given path frequency," J. Chemical Information and Modeling, vol.48, pp.1345–1357, 2008.

[14] J. Gasteiger and T. Engel eds., Chemoinformatics, Wiley-VCH, 2003.

[15] T. Horvath, T. Akutsu, and S. Wrobel, "A refinement operator for outerplanar graphs," Proc. 16th International Conference of Inductive Logic Programming, pp.95–97, 2006.

[16] T. Imada, S. Ota, H. Nagamochi, and T. Akutsu, "Efficient enumeration of stereoisomers of tree structured molecules using dynamic programming," J. Mathematical Chemistry, vol.49, pp.910–970, 2011.

[17] T. Imada, Efficient enumeration of stereoisomers of outerplanar chemical graphs using dynamic programming, Department of Applied Mathematics and Physics. Graduate School of Informatics, Kyoto University, Master Thesis, March 2011.

[18] Y. Ishida, Y. Kato, L. Zhao, H. Nagamochi, and T. Akutsu, "Branch-and-bound algorithms for enumerating treelike chemical graphs with given path frequency using detachment-cut," J. Chemical Information and Modeling, vol.50, pp.934–946, 2010.

[19] H. Kashima, K. Tsuda, and A. Inokuchi, "Marginalized kernels between labeled graphs," Proc. 20th Int. Conf. Machine Learning, pp.321–328, 2003.

[20] L.B. Kier, L.H. Hall, and J.W. Frazer, "Design of molecules from quantitative structure-activity relationship models. 1. Information transfer between path and vertex degree counts," J. Chemical Information and Computer Science, vol.33, pp.143–147, 1993.

[21] P. Mahé, N. Ueda, T. Akutsu, J-L. Perret, and J-P. Vert, "Graph kernels for molecular structure-activity relationship analysis with support vector machines," J. Chemical Information and Modeling, vol.45, pp.939–951, 2005.

[22] P. Mahé and J-P. Vert, "Graph kernels based on tree patterns for molecules," Mach. Learn., vol.75, pp.3–35, 2009.

[23] J. Mohr, B. Jain, A. Sutter, A.T. Laak, T. Steger-Hartmann, N. Heinrich, and K. Obermayer, "A maximum common subgraph kernel method for predicting the chromosome aberration test," J. Chemical Information and Modeling, vol.50, pp.1821–1838, 2010.

[24] H. Nagamochi, "A detachment algorithm for inferring a graph from path frequency," Algorithmica, vol.53, pp.207–224, 2009.

[25] S. Nakano and T. Uno, "Generating colored trees," Lect. Notes Comput. Sci., no.3787, pp.249–260, 2005.

[26] J.G. Nourse, "The configuration symmetry group and its application to stereoisomer generation, specification, and enumeration," J. American Chemical Society, vol.101, pp.1210–1216, 1979.

[27] G. Pólya, "Kombinatorische anzahlbestimmungen für gruppen, graphen, un chemische verbindungen," Acta Mathematica, vol.68, pp.145–253, 1937.

[28] M.I. Skvortsova, I.I. Baskin, O.L. Slovokhotova, V.A. Palyulin, and N.S. Zefirov, "Inverse problem in QSAR/QSPR studies for the case of topological indices characterizing molecular shape (Kier indices)," J. Chemical Information and Computer Science, vol.33, pp.630–634, 1993.

[29] J. Wang and H. Nagamochi, "Constant time generation of rooted and colored outerplanar graphs," Graduate School of Informatics, Kyoto University, Techinical Report, 2010-007, 2010.

Tatsuya Akutsu received his M.Eng degree in Aeronautics in 1986 and a Dr. Eng. degree in Information Engineering in 1989 both from University of Tokyo, Japan. From 1989 to 1994, he was with Mechanical Engineering Laboratory, Japan. He was an associate professor in Gunma University from 1994 to 1996 and in Human Genome Center, University of Tokyo from 1996 to 2001 respectively. He joined Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan as a professor in Oct. 2001. His research interests include bioinformatics and discrete algorithms.

Hiroshi Nagamochi was born in Tokyo, on January 1, 1960. He received the B.A., M.E. and D.E. degrees from Kyoto University, in 1983, in 1985 and in 1988, respectively. He is a Professor in the Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University. His research interests include network flow problems and graph connectivity problems. Dr. Nagamochi is a member of the Operations Research Society of Japan, the Information Processing Society, and the Japan Society for Industrial and Applied Mathematics.