# Branch-and-Bound Algorithms for Enumerating Treelike Chemical Graphs with Given Path Frequency Using Detachment-Cut

Yusuke Ishida,[†] Yuki Kato,*[,‡] Liang Zhao,[†] Hiroshi Nagamochi,[†] and Tatsuya Akutsu[‡]

Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, Yoshida, Kyoto 606-8501, Japan, and Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan

Received November 18, 2009

Computational methods of enumerating chemical graphs have attained great importance in chemoinformatics since they lead to a variety of useful applications including structure determination of novel chemical compounds. Recently, Fujiwara et al. have presented an efficient branch-and-bound algorithm for enumerating treelike chemical graphs with given path frequency. In this paper, we augment Fujiwara et al.'s algorithm by introducing a new bounding operation called detachment-cut to reduce further the search space in the branch-and-bound framework. Experimental results on much chemical compound data show that our proposed algorithm achieves better performance than Fujiwara et al.'s algorithm in computation time. A program that implements our algorithm can be used freely via Web server.

## INTRODUCTION

Development of novel and useful chemical structures is one of the major goals in chemoinformatics and bioinformatics, and it is important to enumerate chemical structures under various constraints to achieve this purpose. Chemical structures can be represented by *chemical graphs* where vertices and edges represent atoms and chemical bonds, respectively. The pioneering work for enumerating chemical graphs performed by Cayley[1] was dedicated to enumerating structural isomers of alkanes, and a century later several studies based on computational methods followed.[2−6] Other applications of enumerating chemical graphs include reconstruction of molecular structures from their signatures,[7,8] classification of chemical compounds,[9] and virtual exploration of the chemical universe.[10,11] In these applications, it is important to enumerate chemical graphs that satisfy given constraints.[12]

In the field of machine learning, the *preimage problem*[13] has been studied where given an arbitrary feature vector in a feature space, the corresponding preimage in an input space is computed. One of the successful methods of giving the "feature" for chemical graphs is to consider the frequency of labeled paths.[14,15] It must be noted that the preimage problem can be considered as the problem of enumerating chemical graphs from a given path frequency. Akutsu and Fukagawa[16] addressed the problem of inferring graphs from the frequency of paths of labeled vertices, which corresponds to the preimage problem, and proved that the problem is NP-hard even if the graph with bounded degrees is planar. They also developed a branch-and-bound algorithm for treelike chemical graphs.[12] Recently, Fujiwara et al.[17] presented a much more efficient branch-and-bound algorithm, which combines a branching operation of generating nonisomorphic trees based on the tree enumeration algorithm proposed by Nakano and Uno[18,19] with bounding operations of avoiding the generation of invalid trees based on the path frequency and the atom−atom bonds. To improve the computation efficiency, Fujiwara et al. provided further an alternative problem formulation by removing all hydrogens and replacing each multiple edge with a new virtual atom and two new single edges incident to the atom.

The preimage problem has also been studied as a part of inverse QSAR/QSPR (quantitative structure−activity relationship/quantitative structure−property relationship) studies. In particular, it is almost the same as reconstruction and/or enumeration of molecules from their descriptors in inverse QSAR/QSPR, where the descriptors correspond to feature vectors in the preimage problem. Kier et al.[20] 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.[8] Their methods first compute all the possible degree sequences matching the count of paths up to a length of two then generate all the structures from each degree sequence. In the case of a length up to three, structures that do not match the count of length three paths are rejected. Skvortsova et al.[21] developed a similar method, in which paths of the same length are further classified into several classes based on atom and bond types. However, it seems that applications of these methods are restricted to small size molecular structures. A decade later, Faulon et al.[7] defined another descriptor and developed methods for enumerating all the structures consistent with a given descriptor, where a descriptor is a vector of the occurrence number of atomic signatures, each of which is a canonical treelike representation of the atom's environment up to a predefined height *h*. Furthermore, they demonstrated the usefulness of their approach by applying it to the design of novel peptides[22] and novel polymers.[23] Although their approach is useful, its efficiency strongly depends on the descriptor. In particular, their approach is not efficient for enumerating large structures

* Corresponding author e-mail: ykato@kuicr.kyoto-u.ac.jp.
[†] Graduate School of Informatics.
[‡] Institute for Chemical Research.

if a small $h$ is used, and the use of a large $h$ may tend to cause overfitting.

This work is intended to achieve further efficiency of enumerating treelike chemical graphs with a given path frequency. For this purpose, we introduce a new bounding operation called *detachment-cut* in Fujiwara et al.'s branch-and-bound algorithm. It checks the connectivity of graphs with a given path frequency via the detachment operation, which is based on Nagamochi's work.[24] We introduce an additional bounding operation called *H-cut*, which uses information of removed hydrogens for the second problem formulation described above. This paper augments the preliminary work that adopts the above idea, which was presented in a conference book series.[25] Specifically, major augmentations of this paper are summarized as follows:

• We propose a method of reducing the search space by considering the label order in the branching operation of the enumeration algorithm.

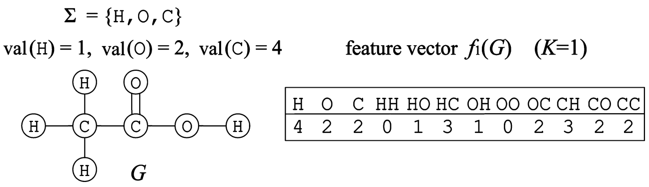
• We extend the detachment-cut in the bounding operation to deal with the case where the maximum path length is two. Note that the preliminary version[25] can handle only the case where the maximum path length is one.

• More exhaustive comparative experiments are carried out, where the performance of our algorithm is scrutinized in terms of utilizing label order and detachment-cut. In addition, we show that our proposed algorithm runs much faster than the Fujiwara et al.'s algorithm does on the same instances.

• A Web server EnuMol that implements our proposed algorithm is developed and available online.

It is to be noted that our proposed method has several novel points compared with existing studies on enumeration of structures from descriptors.[7,8,20,21] First, the descriptor (i.e., feature vector) of our method uses the count of vertex labeled paths. Although descriptors in the existing methods[8,20,21] use information on paths, these do not explicitly use vertex labels (i.e., atom labels). Explicit use of vertex labels greatly contributes to enhancement of the efficiency of our bounding operations. Second, our proposed method employs a recently developed sophisticated technique for nonredundant enumeration of labeled trees.[18,19] Third, our proposed method uses a strong bounding operation named detachment-cut, which makes explicit use of vertex labels. By means of these modifications, our method can enumerate medium-sized tree-structured molecules from feature vectors corresponding to the count of short vertex labeled paths.

It should also be noted that our proposed method has a potential application to inverse QSAR/QSPR. For that purpose, as discussed by Faulon et al.[7] and Churchwell et al.,[22] we should first employ some forward QSAR/QSPR method of obtaining a regression function, then get feature vectors that give desired activity/property values, and finally compute structures from these feature vectors. Our proposed method can be used for the last step, whereas we may employ existing methods[7,22] for the first and second steps. Since our proposed method is restricted to tree-structured molecules, the types of molecules to be designed are also restricted and thus the method itself is not very practical, though there exist some practical cases where treelike or even linear structures are useful.[22] However, the techniques of bounding operations shown in this paper are not restricted to tree-like structures.


$\Sigma = \{H, O, C\}$
val(H) = 1, val(O) = 2, val(C) = 4
feature vector $f_1(G)$ ($K$=1)
$G$

| H | O | C | HH | HO | HC | OH | OO | OC | CH | CO | CC |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 2 | 2 | 0 | 1 | 3 | 1 | 0 | 2 | 3 | 2 | 2 |

**Figure 1.** An illustration of a ($\Sigma$, val)-labeled multitree $G$ and $f_1(G)$, where multiple edges are treated as one edge and paths are considered "directed."

Therefore, these techniques might also be used for developing enumeration methods for more general structures.

## PRELIMINARIES AND PROBLEM FORMULATION

In this section, we will give some basic definitions on graphs and formalize the problem to be addressed in this work. A graph is called a *multigraph* if multiple edges and self-loops are allowed; otherwise, it is *simple*. A *multitree* is a multigraph with no cycle or self-loop. A *path* $P$ is a sequence $v_0, e_1, v_1, e_2, v_2, ..., e_k, v_k$ of distinct vertices $v_i$ ($i = 0, ..., k$) and edges $e_j$ that join $v_{j-1}$ and $v_j$ ($j = 1, ..., k$). Without confusion, we may write $P = (v_0, v_1, ..., v_k)$. The length $|P|$ of the path $P$ is defined as $k$, i.e., the number of edges in the path.

We are given a set $\Sigma = \{l_1, l_2, ..., l_s\}$ of $s$ labels, which correspond to chemical elements. Let each label $l$ be associated with a valence $\mathrm{val}(l) \in \mathbb{Z}_+$, where $\mathbb{Z}_+$ denotes the set of non-negative integers. A multigraph $G$ is said to be $\Sigma$-*labeled* if each vertex $v$ has a label $l(v) \in \Sigma$ and is called ($\Sigma$, val)-*labeled* if, in addition, the degree of each vertex $v$ is $\mathrm{val}(l(v))$, i.e., the valence of the element $l(v)$. Chemical compounds can be viewed as ($\Sigma$, val)-labeled, self-loopless, and connected multigraphs, where vertices and labels represent atoms and elements, respectively. For a path $P = (v_0, v_1, ..., v_k)$, we call $l(P) = l(v_0), l(v_1), ..., l(v_k)$ the *label sequence* of $P$. Given a label sequence $t$, let $\#t$ denote the number of paths $P$ with $l(P) = t$ in the graph, where multiple edges are treated as a single edge and paths are considered "directed." The *feature vector* $f_K(G)$ *of level* $K \in \mathbb{Z}_+$ of $G$ is defined as the $p(K, s)$-dimensional vector whose entry $f_K(G)[t]$ ($|t| \leq K$) represents $\#t$, where $p(K, s) = (s^{K+2} - s)/(s - 1)$ for $s > 1$ and $p(K, 1) = K + 1$. Figure 1 illustrates an example.

Let $\deg(v; G)$ denote the degree of a vertex $v$ in a multigraph $G$. The problem can be formulated as follows.

**Problem 1.** Given a set $\Sigma$ of $s$ labels, a valence function $\Sigma \rightarrow \mathbb{Z}_+$, and a feature vector $g$ of level $K$, find all ($\Sigma$, val)-labeled multitrees $T$ such that $f_K(T) = g$ and $\deg(v;T) = \mathrm{val}(l(v))$ for all vertices $v \in T$.

Note that the case where $K = 0$ corresponds to the enumeration of constitutional isomers of a given molecular formula.

For a given feature vector $g$, the entry $g(t)$ specifies $\#t$ in an output graph. In particular, the number $n$ of vertices is decided by $\Sigma_{l \in \Sigma} g(l)$. To solve the problem, we start with an empty graph and repeatedly extend the current tree $T$ by appending a new vertex with each label $l \in \Sigma$ to obtain a *valid* tree (a tree that has not violated any constraints on output trees) by one vertex until we get $n$ vertices. In order to avoid duplicate outputs, we follow the branch-and-bound framework presented by Fujiwara et al.,[17] which first defines a canonical representation for isomorphic trees, then lists

```
Main:
  FOR all labels $\ell \in \Sigma$ DO
    Let $T$ be the tree consisting of one (root) vertex labeled by $\ell$.
    Gen($T$)
  DONE

Gen($T$):
  IF $T$ has $n$ vertices THEN
    Check that $T$ is valid. If so, output it.
  ELSE
    Let $T_1, T_2, \ldots, T_p$ be a set of trees obtained by the branching operation.
    FOR all trees $T_i$ $(1 \le i \le p)$ DO
      Check that $T_i$ is valid. If so, call Gen($T_i$) (do nothing otherwise).
    DONE
  ENDIF
```

**Figure 2.** The framework of the enumeration algorithm.

them using the algorithm of Nakano and Uno[18,19] (the branching operation) and discards invalid trees using some bounding operations.

## ENUMERATION ALGORITHM

This section provides details of the branch-and-bound algorithm for enumerating treelike structures. Given a simple tree with $n$ vertices, the valence constraint uniquely determines the multiplicities of all edges. Thus, we list all nonisomorphic $\Sigma$-labeled *simple* trees and then get/check the corresponding multitrees by the valence constraint. The framework of the enumeration algorithm is shown in Figure 2, and we will refer to the algorithm designed for Problem 1 as Algorithm $A_D$ in the rest of the paper. The description of the branch-and-bound algorithm that includes duplication of a reference[17] is indispensable for understanding our new approach, and we present it concisely in this paper.

The Canonical Representation of Trees and Branching Operation section reviews the way of extending trees (the branching operation), which is exactly the same as the method by Fujiwara et al.[17] The Bounding Operations section describes how the validity is checked by four bounding operations, three of which come from the earlier work[17] and the other being a new operation utilizing *detachment-cut*.

## CANONICAL REPRESENTATION OF TREES AND BRANCHING OPERATION

First of all, we need a representation for the output that must be unique for isomorphic trees. For this purpose, we use the idea of a *centroid-rooted left-heavy tree*,[17] where the centroid is defined by the next theorem (see also ref 26).

**Theorem 1** (Jordan[27]). For any tree with $n$ vertices, either there exists a unique vertex $v^*$ such that each subtree obtained by removing $v^*$ contains at most $\lfloor (n-1)/2 \rfloor$ vertices or there exists a unique edge $e^*$ such that both of the subtrees obtained by removing $e^*$ contain exactly $n/2$ vertices.

Such a vertex $v^*$ (edge $e^*$) is called the *unicentroid* (*bicentroid*, respectively) of the tree. For example, the tree in Figure 1 has a bicentroid (the C−C edge). To introduce "left-heaviness," we need an *ordering* among rooted trees.

Let $T$ be a tree of $n$ vertices rooted at a vertex $v_0$ (which is not necessarily its centroid). Suppose that it is embedded in the plane, where $v_0$ is the top. Let $v_0, v_1, ..., v_{n-1}$ be indexed by the depth-first search (DFS) that starts from $v_0$ and visits vertices from the left to the right. The *depth* $d(v)$ of a vertex $v$ is defined as the length of the path from $v_0$ to $v$ in $T$. The *depth-label sequence* of $T$ is defined as

$$\mathrm{DL}(T) = (d(v_0), l(v_0), d(v_1), l(v_1), ..., d(v_{n-1}), l(v_{n-1}))$$

We say that $T$ is rooted at an edge $(v_0, v_1)$ if $v_0$ and $v_1$ are the two tops, where we define $d(v)$ as the minimum of the lengths of the $v_0$, $v$ path and the $v_1$, $v$ path. Then, DL($T$) for trees rooted at edges can be defined as before. Now we have a one-to-one mapping between plane-embedded trees and label sequences. See Figure 3 for an illustration.

Given an (arbitrary) order of labels, define the order of depth-label sequences as follows. For any $T_1$ and $T_2$, we say $\mathrm{DL}(T_1) > \mathrm{DL}(T_2)$ if $\mathrm{DL}(T_1)$ is *lexicographically* larger than $\mathrm{DL}(T_2)$. Similarly, we can define $\mathrm{DL}(T_1) \ge \mathrm{DL}(T_2)$ if $\mathrm{DL}(T_1) > \mathrm{DL}(T_2)$ or $\mathrm{DL}(T_1) = \mathrm{DL}(T_2)$. In Figure 3, we have $\mathrm{DL}(T_1) > \mathrm{DL}(T_2) > \mathrm{DL}(T_3)$, supposing C > O > H. The *canonical representation* of a rooted tree is defined as the *largest* depth-label sequence among all its plane embeddings. This is equivalent to the *left-heavy* plane embedding (see refs 18 and 19); i.e., any two siblings (vertices having the same parent or the two vertices of the edge root) $v_i$ and $v_j$ with $i < j$ satisfy $\mathrm{DL}(\mathbf{T}(v_i)) \ge \mathrm{DL}(\mathbf{T}(v_j))$, where $\mathbf{T}(v)$ denotes the subtree consisting of $v$ and all its descendants. For example, $T_1$ and $T_3$ in Figure 3 are left-heavy, whereas $T_2$ is not.

Thus, our branching task is to list all centroid-rooted left-heavy trees with $n$ vertices and $s$ or less labels. Following the scheme of Nakano and Uno,[18,19] we define a *parent−child* relation between two left-heavy trees. The *parent* $P(T)$ of a left-heavy tree $T$ is obtained from $T$ by removing its *right-most* leaf. If $T$ is rooted at a vertex or an edge $(v_0, v_1)$ but $v_1$ is not the right-most leaf, then the root of $P(T)$ remains unchanged. Otherwise, we change the root to *vertex* $v_0$ since $v_1$ is removed. Clearly, $P(T)$ is still left-heavy. In this way we can define a *family tree* $\mathscr{F}(n, s)$ of left-heavy trees whose leaves are exactly what we want, i.e., the centroid-rooted left-heavy trees with $n$ vertices and $s$ or less labels. Notice that, in general, a nonleaf node in the family tree may not be rooted at its own centroid.

Therefore, we only need to enumerate the (leaf) nodes of $\mathscr{F}(n, s)$. This can be performed by starting from the empty tree (the root node of $\mathscr{F}(n, s)$) and repeatedly appending a new leaf to the current tree. In the current tree $T$, a vertex to which a new leaf can be attached is called *active* and is called *fixed* otherwise. We let $\mathrm{RP}(T) = (r_0, r_1, ..., r_k)$ denote the right-most path of $T$. It is known that the set of all active vertices appears as a subpath $r_0, ..., r_h$ $(h \le k)$ of $\mathrm{RP}(T)$.[18,19] Our branching operation employs the algorithm of Nakano and Uno,[18,19] which extends the current tree $T$ (i.e., finds a child of $T$) in *constant* time. See a ref 17 for details.

## BOUNDING OPERATIONS

Next, we explain how to check the validity of a tree $T$ generated during the branching operation. If we can conclude that $T$ and all its descendants are not valid, then we can discard $T$, i.e., skip the task of appending leaves to $T$. Our bounding operation discards $T$ if at least one of the following criteria is violated:

**(C1)** The root of $T$ remains the centroid of an output (the centroid constraint).

**(C2)** $f_K(T) \le g$ (the feature vector constraint).

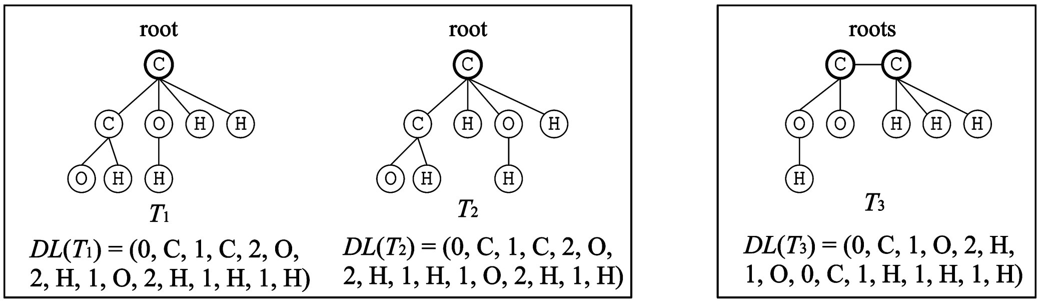

root
root
roots
$T_1$
$T_2$
$T_3$
$DL(T_1)$ = (0, C, 1, C, 2, O, 2, H, 1, O, 2, H, 1, H, 1, H)
$DL(T_2)$ = (0, C, 1, C, 2, O, 2, H, 1, H, 1, O, 2, H, 1, H)
$DL(T_3)$ = (0, C, 1, O, 2, H, 1, O, 0, C, 1, H, 1, H, 1, H)

**Figure 3.** Rooted trees and their depth-label sequences. Notice that $T_1$ and $T_2$ are isomorphic.

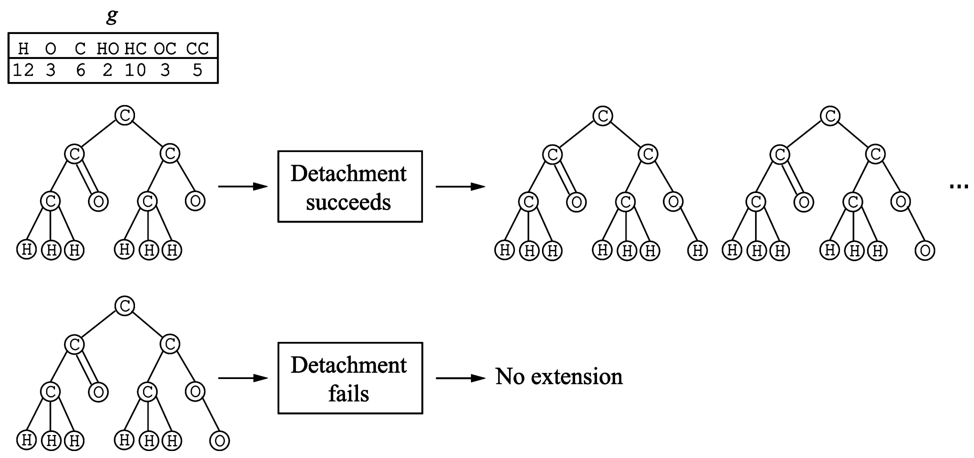


**Figure 4.** Overview of the detachment-based bounding operation.

**(C3)** $\deg(v; T) \leq \mathrm{val}(l(v))$ for all $v \in T$ (the valence constraint).

**(C4)** $T$ can be extended to a connected and loopless tree with $n$ vertices (the detachment constraint).

The first three criteria are the same as the work by Fujiwara et al.[17] and not difficult to check. In the following, we explain how to check the last criterion (C4). Figure 4 shows an overview of how the detachment-based bounding operation works while the algorithm is extending trees.

## DETACHMENT-CUT PROCEDURE

This subsection describes the definition of detachment and presents a new detachment-based bounding operation. Let $G$ be a multigraph that may have self-loops, obtained from a chemical graph $H$ by contracting the vertices with the same label into a single vertex, where each vertex in $G$ corresponds to a label in $H$ (note that we do not eliminate any edges in $H$ in contracting vertices to obtain $G$). A process of regaining $H$ from $G$ is described as follows. Given a function $r$: $V(G) \to \mathbb{Z}_+$, an *r-detachment* $H$ of $G$ is defined as a multigraph obtained from $G$ by splitting each vertex $v \in V(G)$ into a set of $r(v)$ copies of $v$, denoted by $W_v = \{v^1, v^2, ..., v^{r(v)}\}$, so that each edge $\{u, v\} \in E(G)$ joins some vertices, $u^i \in W_u$ and $v^j \in W_v$. Hence, an $r$-detachment $H$ of $G$ is not unique in general. A self-loop $\{u, u\}$ in $G$ may be mapped to a self-loop $\{u^i, u^i\}$ or a nonloop edge $\{u^i, u^j\}$ in a detachment $H$ of $G$. Note that, for all vertex pairs $\{u, v\} \in V(G)$, the number of edges between subsets $W_u$ and $W_v$ in $H$ is equal to that of edges between vertices $u$ and $v$ in $G$.

To obtain a chemical graph $H$ as an $r$-detachment $H$ of $G$, we need to specify the degree of vertices (with the same label) in $H$. For a function $r$: $V(G) \to \mathbb{Z}_+$, an *r-degree specification* is a set $\rho$ of vectors $\rho(v) = (\rho_1^v, \rho_2^v, ..., \rho_{r(v)}^v)$ for $v \in V(G)$ such that

$$\sum_{i=1}^{r(v)} \rho_i^v = \deg(v;G)$$

which is necessary for all the edges incident to the vertex $v$ in $G$ to be assigned to split vertices $v^i \in W_v$ completely. An $r$-detachment $H$ of $G$ is called a *ρ-detachment* if each $v \in V$ satisfies

$$\deg(v^i;H) = \rho_i^v \text{ for all } v^i \in W_v = \{v^1, v^2, ..., v^{r(v)}\}$$

which is a requirement that each vertex $v_i$ in $H$ must have the prescribed degree $\rho_i^v$. Figure 5 illustrates a $\rho$-detachment $H$ for a graph $G = (V, E)$ with $V = \{a, b, c, d\}$; a function $r$ with $r(a) = 1$, $r(b) = 3$, $r(c) = 4$, and $r(d) = 2$; and a degree specification $\rho$ with $\rho(a) = (3)$, $\rho(b) = (3, 3, 1)$, $\rho(c) = (1, 3, 2, 3)$, and $\rho(d) = (2, 3)$.

The next theorem gives a characterization of a multigraph $G$ that admits a connected and loopless $\rho$-detachment.

**Theorem 2** (Nagamochi[24]). Let $G = (V, E)$ be a multigraph, $r$: $V \to \mathbb{Z}_+$ and $\rho$: $V \to \mathbb{Z}_+^{r(v)}$ ($v \in V$). Then, $G$ has a connected and loopless $\rho$-detachment $H$ if and only if the following hold:

$$r(X) + c(G - X) - d(X, V;G) \leq 1 \quad (\emptyset \neq \forall X \subseteq V) \tag{1}$$

$$1 \leq \rho_i^v \leq d(v;G) + d(\{v\}, \{v\};G) \quad (\forall v \in V, i = 1, 2, ..., r(v)) \tag{2}$$

where $r(X) = \Sigma_{v \in X} r(v)$, $c(G')$ denotes the number of connected components of a graph $G'$, $G - X$ denotes the graph

$g$

| $a$ | $b$ | $c$ | $d$ | $aa$ | $ab$ | $ac$ | $bb$ | $bc$ | $cc$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 4 | 2 | 0 | 1 | 2 | 1 | 3 | 1 |

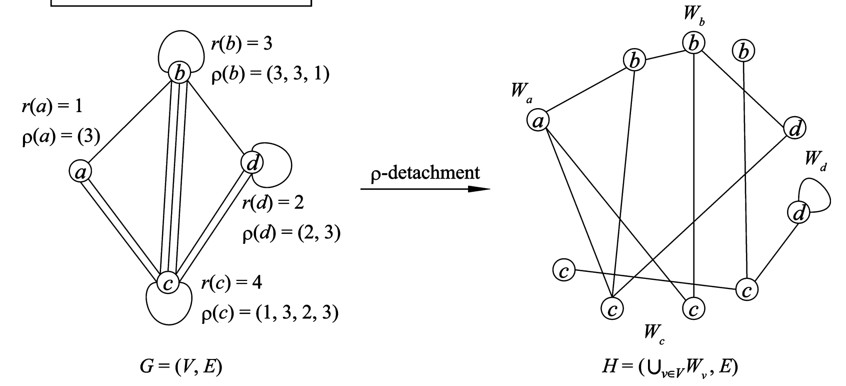


**Figure 5.** A multigraph $G$ with a feature vector $g$ and a $\rho$-detachment $H$ of $G$. Note that $G$ satisfies eqs 1 and 2 in Theorem 2, and $H$ satisfies the feature vector constraint.

$g$

| $a$ | $b$ | $c$ | $aa$ | $ab$ | $ac$ | $bb$ | $bc$ | $cc$ |
|---|---|---|---|---|---|---|---|---|
| 4 | 9 | 3 | 0 | 6 | 2 | 0 | 4 | 3 |

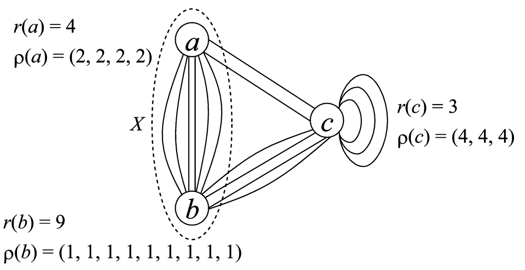


**Figure 6.** A multigraph $G$ with a feature vector $g$ that violates eq 1 in Theorem 2.

obtained from graph $G$ by removing the vertices in $X$ together with all edges incident to vertices in $X$, and $d(A, B; G)$ denotes the number of edges $(u, v) \in E$ with $u \in A$ and $v \in B$.

The inequality (eq 1) in Theorem 2 means that, if there is a connected detachment $H$ of $G$, then the graph $H/(V - X)$ obtained from $H$ by contracting the vertices of each label $l \notin X$ into a single vertex remains connected. The number of vertices in $H/(V - X)$ is given by $r(X) + c(G - X)$, and $H/(V - X)$ must have at least $r(X) + c(G - X) - 1$ edges to be connected. Since $d(X, V; G)$ is the number of edges in $H/(V - X)$, it must hold that $r(X) + c(G - X) - 1 \le d(X, V; G)$ for $G$ to admit the connected detachment $H$. The inequality (eq 2) in Theorem 2 requires the degree of each vertex $v_i$ to be at least one, and a detachment $H$ of $G$ to have no self-loops (note that if the specified degree $\rho_j^v$ of a vertex $v_j \in W_v$ is larger than the number of edges $d(v; G) + d(\{v\},\{v\}; G)$ incident to $v$, then the vertex $v_j$ must have a self-loop in any detachment $H$ of $G$).

Here, we will see two examples to illustrate how to use the inequality (eq 1). To take Figure 5 as the first example, we can check that the graph $G$ satisfies eq 1 regardless of the choice of a subset $X \subseteq V$. Therefore, $G$ has a connected and loopless $\rho$-detachment $H$ as shown in Figure 5. On the other hand, the graph $G$ shown in Figure 6 has a subset $X = \{a, b\} \subseteq V$ that violates the eq 1. In fact, $r(X) = r(a) + r(b) = 4 + 9 = 13$, $c(G - X) = 1$, and $d(X, V; G) = 6 + 2 + 4 = 12$ where 6 comes from the number of edges that connect $a \in X$ and $b \in X$. This tells us that the second example does not satisfy eq 1 and $G$ has no connected and loopless $\rho$-detachment that satisfies the constraint of the feature vector $g$.

Using Theorem 2, we can check that a partial multitree $T$ violates C4. This bounding operation, called *detachment-cut*, terminates the search of any descendant of $T$ if there is no $\rho$-detachment in the multigraph $G$ introduced below.

Let $l_1, l_2, ..., l_s$ be input labels and $g$: $\Sigma^{\le K+1} \to \mathbb{Z}_+$ be a feature vector. Let $n_i^R$ $(1 \le i \le s)$ denote the number of active vertices $r_j$ $(0 \le j \le h)$ with $l(r_j) = l_i$. For each label sequence $t$, $\#t$ denotes the number of paths $P$ in $T$ with $l(P) = t$. Introducing a vertex with a new label $l_{s+1}$ of valence $h + 1$ (see the label $A = l_{s+1}$ in Figure 7), which represents the fixed vertices in $T$, and new edges between the vertex labeled $l_{s+1}$ and active vertices, we define a new feature vector $g'$ for level $K = 1$ as

$$g'(l_i) = \begin{cases} g(l_i) - \#l_i + n_i^R & (1 \le i \le s) \\ 1 & (i = s + 1) \end{cases}$$

$$g'(l_il_j) = \begin{cases} g(l_il_j) - \#l_il_j & (1 \le i,j \le s) \\ n_i^R & (1 \le i \le s, j = s + 1) \end{cases}$$

For a partial multitree $T$, $g'(l_i)$ denotes the sum of the number $g(l_i) - \#l_i$ of vertices labeled $l_i$ that have not been appended to $T$ and the number $n_i^R$ of active vertices labeled $l_i$ in $T$, and $g'(l_il_j)$ denotes the number of edges with end vertices labeled $l_i$ and $l_j$ that have not been appended to $T$. If there exists a multitree $T'$ such that $f_1(T') = g'$, then we can obtain a multitree $T^*$ that satisfies $f_1(T^*) = g$ by combining $T$ and $T'$. More specifically, we can construct $T^*$ by replacing the vertex labeled $l_{s+1}$ and edges incident to it in $T'$ with the fixed vertices and all edges in $T$, respectively. See Figure 7 for an illustration. To check whether such a multitree $T'$ exists using Theorem 2, we construct an auxiliary graph $G = (V, E)$ with a vertex set $V = \{v_1, ..., v_s, v_{s+1} \mid l(v_i) = l_i,\ 1 \le i \le s + 1\}$ and an edge set $E = \{e_{ij} \mid e_{ij} = \{v_i, v_j\},\ d(\{v_i\},\{v_j\}; G) = g'(l_il_j),\ 1 \le i,j \le s + 1\}$ where $d(\{v_i\},\{v_j\}; G)$ means the multiplicity of the edge $e_{ij}$. The vertex set $V$ and the edge set $E$ of $G$ correspond to the label set and the frequency of each edge in the feature vector $g'$, respectively. We check that $G$ has a $\rho$-detachment satisfying the feature vector constraint $g'$ and the valence constraint. Since a

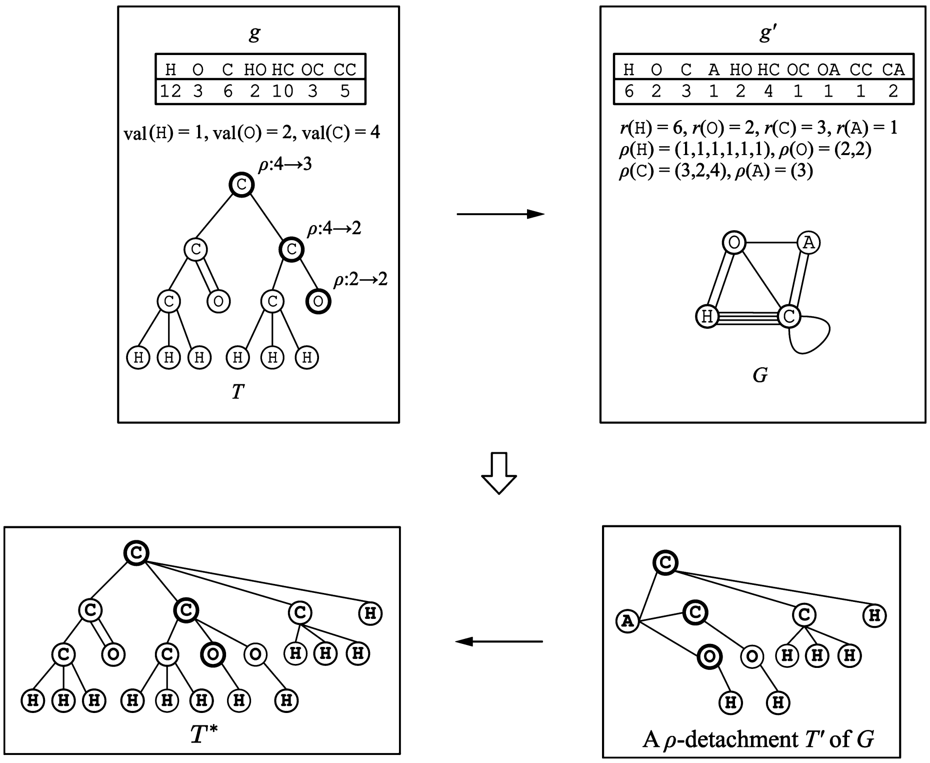

g
H O C HO HC OC CC
12 3 6 2 10 3 5
val(H) = 1, val(O) = 2, val(C) = 4
ρ:4→3
ρ:4→2
ρ:2→2
T
g′
H O C A HO HC OC OA CC CA
6 2 3 1 2 4 1 1 1 2
r(H) = 6, r(O) = 2, r(C) = 3, r(A) = 1
ρ(H) = (1,1,1,1,1,1), ρ(O) = (2,2)
ρ(C) = (3,2,4), ρ(A) = (3)
G
T*
A ρ-detachment T′ of G

**Figure 7.** A partial multitree $T$, a $\rho$-detachment $T'$ of $G$, and the multitree $T^*$ obtained by combining $T$ and $T'$. The multitree $T^*$ satisfies the feature vector constraint $g$. Note that we do not construct $T'$ and $T^*$ explicitly.

function $r$ in detachments specifies the number of copies of each vertex in $G$ that represents a label, we set $r(v_i)$ for each $v_i \in V$ to be the frequency $g'(l_i)$. Hence, each copy $v^j \in \{v^1, ..., v^{r(v)}\}$ of a vertex $v = v_i \in V$ has not been appended to $T$ or an active vertex in $T$. For the former $v^j$, the degree $\rho_j^v$ of the copy $v^j$ is given by the original valence of the label $l_i$. For the latter $v^j$, the degree $\rho_j^v$ of the copy $v^j$ in a detachment $T'$ is given from the valence of $l_i$ by subtracting the number $\deg(v^j; T)$ of edges incident to $v^j$ in $T$ and by increasing by one, which stands for the edge that joins $v^j$ and the vertex labeled $l_{s+1}$. Therefore, a function $r$ and a degree specification $\rho$ for $v = v_i \in V$ are defined as follows:

$$r(v_i) = g'(l_i) \quad (1 \le i \le s+1)$$

$$\rho_j^v = \begin{cases} \mathrm{val}(l_i) & (v^j \notin T, 1 \le j \le r(v)) \\ \mathrm{val}(l_i) - \deg(v^j;T) + 1 & (v^j \in T, 1 \le j \le r(v)) \end{cases}$$

The detachment-cut terminates the search of any descendant of $T$ if there is no $\rho$-detachment of the above graph $G = (V, E)$ with $r$ and $\rho$. By Theorem 2, we need to check that one or more of the next two conditions is violated.

$$\sum_{1 \le i \le r(v)} \rho_i^v \ge \deg(v;G) \quad (\forall v \in V) \tag{a}$$

$$r(X) + c(G - X) - d(X, V;G) \le 1 \quad (\forall X \subseteq V, X \ne \emptyset) \tag{b}$$

Notice that condition a is not equality since the feature vector counts multiple edges as one edge. Our detachment-cut discards $T$ if any of a and b is violated. Let $n$ and $s$ denote the number of vertices and the number of labels given by the input, respectively. The time complexity of executing the detachment-cut is $O(n + s^2 2^s)$ due to Nagamochi's algorithm,[24] where $s \le 5$ in our problem setting in this paper.

Next, we describe the detachment-cut for level $K = 2$. Considering an edge and a path of length two to be a new vertex and a new edge, respectively, enables us to apply the detachment-cut for $K = 1$ described above.

We treat an edge between vertices $v_i$ and $v_j$ as a virtual vertex $v_{ij}$ called *divertex* and treat a path of length two as a virtual edge joining two divertices called *diedge*. The label of a divertex $v_{ij}$ with $l(v_i) = l_i$ and $l(v_j) = l_j$ is defined as $l_{ij}$. A divertex $v_{ij}$ (a label $l_{ij}$) may be written as $v_{i,j}$ ($l_{i,j}$, respectively). For the vertices $r_0$, ..., $r_h$ in the right-most path of $T$ to which a new vertex can be attached, a divertex is called *active* if it is incident to any of those vertices and is called *fixed* otherwise. In Figure 8, active divertices in $T$ are indicated by thick lines. Let $n_{ij}^R$ denote the number of active divertices labeled $l_{ij}$. Introducing a divertex with a new label $l_{s+1,s+1}$ of valence $\Sigma_i\Sigma_j n_{ij}^R$ (see the label $A = l_{s+1,\,s+1}$ in Figure 8), which represents the fixed divertices in $T$, and new diedges between the divertex labeled $l_{s+1,s+1}$ and active divertices, we define a new feature vector $g''$ for level $K = 2$ as

$$g''(l_{ij}) = \begin{cases} g(l_il_j) - \#l_il_j + n_{ij}^R & (1 \le i,j \le s) \\ 1 & (i = j = s+1) \end{cases}$$

$$g''(l_{ij}l_{hk}) = \begin{cases} g(l_il_jl_k) - \#l_il_jl_k & (1 \le i,j,k \le s, h = j) \\ n_{ij}^R & (1 \le i,j \le s, h = k = s+1) \end{cases}$$

For a partial multitree $T$, $g''(l_{ij})$ denotes the sum of the number $g(l_il_j) - \#l_il_j$ of divertices labeled $l_{ij}$ that have not been appended to $T$ and the number $n_{ij}^R$ of active divertices labeled $l_{ij}$ in $T$, and $g''(l_{ij}l_{hk})$ denotes the number of diedges with end divertices labeled $l_{ij}$ and $l_{hk}$ that have not been appended to $T$. If there exists a multitree $T'$ such that $f_2(T')$

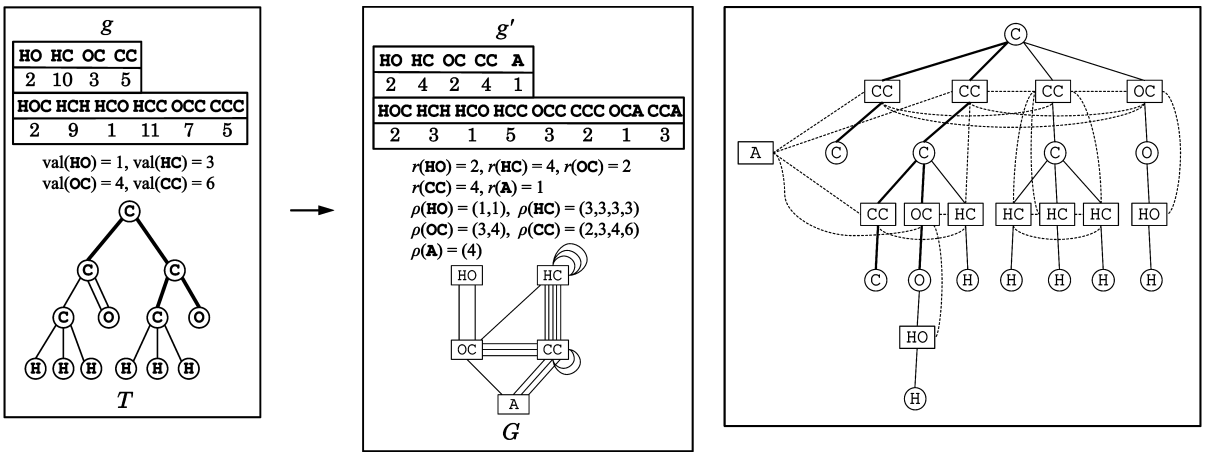


**Figure 8.** Example of detachment-cut for level $K = 2$. The right-most tree shows a $\rho$-detachment of $G$. Note that each divertex is surrounded by a square and each diedge is indicated by a dashed line.

$= g''$, then we can obtain a multitree $T^*$ that satisfies $f_2(T^*) = g$ by combining $T$ and $T'$. More specifically, we can construct $T^*$ by replacing the divertex labeled $l_{s+1,s+1}$ and diedges incident to it in $T'$ with the fixed divertices and all diedges in $T$, respectively. To check whether such a multitree $T'$ exists using Theorem 2, we construct an auxiliary graph $G = (V, E)$ with a vertex set $V = \{v_{1,1}, v_{1,2}, ..., v_{s,s}, v_{s+1,s+1} \mid l(v_{i,j}) = l_{i,j}\}$ and an edge set $E = \{e_{ijk} \mid e_{ijk} = \{v_{ij}, v_{jk}\}, d(\{v_{ij}\},\{v_{jk}\}; G) = g''(l_{ij}l_{jk}),\ 1 \le i, j, k \le s\} \cup \{e_{ijk} \mid e_{ijk} = \{v_{ij}, v_{kk}\},\ d(\{v_{ij}\},\{v_{kk}\}; G) = g''(l_{ij}l_{kk}),\ 1 \le i, j \le s,\ k = s + 1\}$. The vertex set $V$ and the edge set $E$ of $G$ correspond to the label set of divertices and the frequency of each diedge in the feature vector $g''$, respectively. We check whether $G$ has a $\rho$-detachment that satisfies the feature vector constraint $g''$ and the valence constraint. Since a function $r$ in detachments specifies the number of copies of each vertex in $G$ that represents the label of a divertex, we set $r(v_{ij})$ for each $v_{ij} \in V$ to be the frequency $g''(l_{ij})$. Hence, each copy $v^k \in\{v^1, ..., v^{r(v)}\}$ of a divertex $v = v_{ij} \in V$ has not been appended to $T$ or an active divertex in $T$. For the former $v^k$, the degree $\rho_k^v$ of the copy $v^k$ is given from the sum of the valences of the labels $l_i$ and $l_j$ by subtracting two, which stands for the edge that joins the vertices $v_i$ and $v_j$. For the latter $v^k$, the degree $\rho_k^v$ of the copy $v^k$ in a detachment $T'$ is given from the sum of the valences of $l_i$ and $l_j$ by subtracting the number $\deg(v_i; T)$ of edges incident to $v_i$ and the number $\deg(v_j; T)$ of edges incident to $v_j$ in $T$, and increasing by one, which stands for the edge that joins $v^k$ and the divertex labeled $l_{s+1,s+1}$. Therefore, a function $r$ and a degree specification $\rho$ for $v = v_{ij} \in V$ are defined as follows:

$$r(v_{ij}) = g''(l_{ij})(1 \le i,j \le s \text{ or } i = j = s + 1)$$

$$\rho_k^v = \begin{cases} \text{val}(l_i) + \text{val}(l_j) - 2 & (v^k \notin T, 1 \le k \le r(v)) \\ \text{val}(l_i) + \text{val}(l_j) - (\deg(v_i;T) + \deg(v_j;T)) + 1 & (v^k \in T, 1 \le k \le r(v)) \end{cases}$$

If the graph $G$ has no connected and loopless $\rho$-detachment for the above $r$ and $\rho$, we discard the tree $T$. In fact, the algorithm checks whether the inequality (eq 1) of Theorem 2 holds for sets $X = \{v\}$, $\forall v \in V$ since much computation might be required for all $X \subseteq V$ with an increase in the number of labels. The time complexity of the detachment-cut for $K = 2$ is evaluated as $O(n + s^4 2^{s^2})$ since the number of vertices that correspond to the original edges is $O(n)$ and the number of labels is $O(s^2)$.

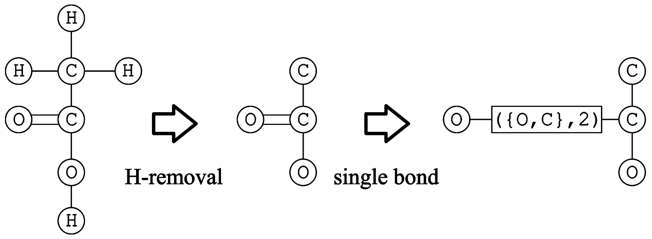


**Figure 9.** An illustration of the H-removal and single-bond transformations.

## ALTERNATIVE PROBLEM FORMULATION

In order to improve the computation efficiency, we consider the second problem formulation as in the earlier study,[17] where two kinds of graph transformations are used. One is the *H-removal transformation*, which reduces the size of compounds by removing hydrogens. The other is the *single-bond transformation*, which replaces each multiple edge with a new virtual atom and two new simple edges joining the same end points. Figure 9 illustrates these two transformations.

In the single-bond transformation that replaces a multiple edge $(u, v)$ with a new vertex $w$ and two new simple edges $(u, w)$ and $(w, v)$, we define the *bond label* $l(w)$ of $w$ by $l(w) = (\{l(u),l(v)\})$ and define the *bond valence* of $l(w)$ by the multiplicity of $(u, v)$. Let $C_\Sigma$ be the set of all such bond labels and $\Sigma' = \Sigma \cup C_\Sigma$. Remember that $\Sigma$ is a set of labels representing atoms. For each vertex $v \in \Sigma'$, its *bond degree* $\widetilde{\deg}(v; T)$ is defined as the number of vertices adjacent to $v$ in a simple tree $T$. We then consider the next formulation.

**Problem 2.** Given a set of labels $\Sigma'$, a feature vector $g$ of level $K$ and a valence function val: $\Sigma \to \mathbb{Z}_+$, find all $\Sigma'$-labeled simple trees $T' = (V', E')$ that satisfy $f_K(T') = g$ and $\widetilde{\deg}(v;T') \le \text{val}(l(v))$ for all $v \in V'$.

For this problem, we develop an extension of Algorithm $A_D$ for Problem 1, called Algorithm $B_D$, where the branching operation is the same as that of Algorithm $A_D$, but we use the bounding operation with a slight modification. In fact, we can still employ the bounding operations based on the

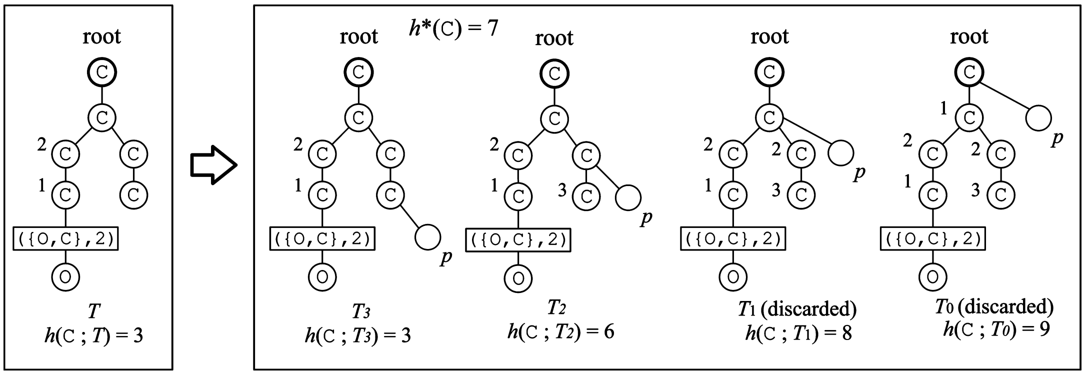


**Figure 10.** An illustration of the H-cut procedure, where only label C is being considered and numbers val($l(v)$) − deg($v$; $T$) are shown near each carbon not on the right-most path.

**Table 1.** Comparison of the Cases When Using and Not Using Prioritized Label Order (Without the Detachment-Cut for $K = 2$)[a]

| | | | without prioritized label order | | | with prioritized label order | | |
|---|---|---|---|---|---|---|---|---|
| entry formula | $n$ | $K$ | time (sec) | #nodes | #sol. | time (sec) | #nodes | #sol. |
| | | 1 | 4.10 | 3,909,283 | 70,170 | 3.34 | 2,754,980 | 70,170 |
| | | 2 | 0.01 | 4,321 | 16 | 0.01 | 4,040 | 16 |
| C07178 | | 3 | 0.01 | 2,984 | 2 | 0.01 | 2,972 | 2 |
| | 19 | 4 | 0.00 | 1,062 | 1 | 0.00 | 1,065 | 1 |
| $C_{21}H_{28}N_2O_5$ | | 5 | 0.00 | 819 | 1 | 0.00 | 813 | 1 |
| | | 6 | 0.00 | 794 | 1 | 0.00 | 788 | 1 |
| | | 7 | 0.00 | 794 | 1 | 0.00 | 788 | 1 |
| | | 1 | T.O. | 1,619,035,666 | 192,778,589 | T.O. | 1,776,727,111 | 517,954,091 |
| | | 2 | 4.09 | 1,113,024 | 1,198 | 1.84 | 743,911 | 1,198 |
| C03690 | | 3 | 2.93 | 570,616 | 8 | 1.66 | 402,390 | 8 |
| | 25 | 4 | 1.58 | 197,027 | 4 | 0.91 | 150,565 | 4 |
| $C_{24}H_{38}O_4$ | | 5 | 1.08 | 120,718 | 2 | 0.71 | 96,781 | 2 |
| | | 6 | 0.66 | 60,310 | 1 | 0.45 | 50,750 | 1 |
| | | 7 | 0.51 | 46,319 | 1 | 0.35 | 38,761 | 1 |
| | | 1 | T.O. | 1,325,409,183 | 19,616,711 | T.O. | 1,760,206,835 | 24,462,156 |
| | | 2 | T.O. | 474,561,419 | 78,347 | T.O. | 511,779,857 | 88,382 |
| C13806 | | 3 | 140.88 | 22,875,334 | 360 | 132.92 | 22,906,789 | 360 |
| | 28 | 4 | 19.21 | 2,262,602 | 8 | 18.33 | 2,266,533 | 8 |
| $C_{23}H_{41}N_5O_3$ | | 5 | 7.53 | 724,018 | 2 | 7.16 | 725,648 | 2 |
| | | 6 | 3.76 | 296,235 | 1 | 3.64 | 296,846 | 1 |
| | | 7 | 1.82 | 131,075 | 1 | 1.77 | 131,350 | 1 |
| | | 1 | T.O. | 1,631,866,312 | 30,005,406 | T.O. | 1,964,038,054 | 215,027,700 |
| | | 2 | 748.13 | 300,524,875 | 2,520 | 476.01 | 164,518,561 | 2,520 |
| C04036 | | 3 | 24.10 | 4,745,395 | 1 | 16.00 | 3,478,952 | 1 |
| | 29 | 4 | 2.04 | 262,162 | 1 | 1.31 | 207,994 | 1 |
| $C_{19}H_{39}O_7P$ | | 5 | 1.33 | 107,378 | 1 | 0.89 | 100,001 | 1 |
| | | 6 | 0.96 | 60,557 | 1 | 0.68 | 57,164 | 1 |
| | | 7 | 0.73 | 40,493 | 1 | 0.57 | 38,265 | 1 |
| | | 1 | T.O. | 1,368,233,968 | 10 | T.O. | 1,368,168,649 | 35,248,476 |
| | | 2 | T.O. | 347,931,274 | N.F. | T.O. | 434,706,136 | N.F. |
| C13829 | | 3 | 283.40 | 39,195,891 | 330 | 220.06 | 39,212,240 | 330 |
| | 32 | 4 | 11.38 | 1,288,623 | 6 | 9.44 | 1,289,820 | 6 |
| $C_{24}H_{41}NO_2$ | | 5 | 5.12 | 470,265 | 4 | 4.51 | 470,650 | 4 |
| | | 6 | 2.14 | 168,624 | 2 | 1.93 | 168,704 | 2 |
| | | 7 | 1.46 | 103,669 | 1 | 1.28 | 103,695 | 1 |
| | | 1 | T.O. | 1,350,441,908 | 856,729 | T.O. | 1,694,837,933 | 20,888,287 |
| | | 2 | T.O. | 495,815,834 | 3,030 | T.O. | 521,186,457 | 3,502 |
| C03630 | | 3 | 230.95 | 37,803,253 | 13 | 161.11 | 30,976,896 | 13 |
| | 33 | 4 | 15.00 | 1,519,286 | 11 | 10.65 | 1,316,554 | 11 |
| $C_{21}H_{39}O_7P$ | | 5 | 7.21 | 515,752 | 9 | 5.40 | 470,746 | 9 |
| | | 6 | 3.80 | 225,620 | 7 | 2.96 | 207,332 | 7 |
| | | 7 | 1.88 | 92,431 | 5 | 1.50 | 87,036 | 5 |

[a] Note: (1) C07178, C03690, C13806, C04036, C13829, and C03630 are the entries of trimethobenzamide, bis(2-ethylhexyl)phthalate, philanthotoxin 343, 1-palmitoylglycerol 3-phosphate, 7,10,13,16-docosatetraenoylethanolamine, and oleoylglycerone phosphate, respectively, in the KEGG LIGAND database. (2) $n$ is the number of vertices in an instance preprocessed by replacing each benzene ring with a new atom having six valences, and by the H-removal and single-bond transformations. (3) $K$ is the level of a given feature vector. (4) "time" is the CPU time. (5) T.O. means time over (the time limit is set to 1800 seconds). (6) #nodes is the number of nodes of the family trees that are traversed. (7) #sol. is the number of all possible solutions within the time limit. (8) N.F. means "not found."

four criteria C1−C4 as stated in the previous section (notice that Problem 2 considers only simple trees). Moreover, we introduce a new *H-cut* bounding operation, which discards the partial tree $T$ being checked if the number of hydrogens

**Table 2.** Comparison of the Cases when Using and Not Using the Detachment-Cut for $K = 2$ (with Prioritizing the Label Order)

| | | | without detachment-cut for $K = 2$ | | | with detachment-cut for $K = 2$ | | |
|---|---|---|---|---|---|---|---|---|
| entry formula | $n$ | $K$ | time (sec) | #nodes | #sol. | time (sec) | #nodes | #sol. |
| | | 1 | 3.34 | 2,754,980 | 70,170 | 3.34 | 2,754,980 | 70,170 |
| | | 2 | 0.01 | 4,040 | 16 | 0.02 | 3,994 | 16 |
| C07178 | | 3 | 0.01 | 2,972 | 2 | 0.02 | 2,972 | 2 |
| | 19 | 4 | 0.00 | 1,065 | 1 | 0.00 | 1,065 | 1 |
| $C_{21}H_{28}N_2O_5$ | | 5 | 0.00 | 813 | 1 | 0.01 | 813 | 1 |
| | | 6 | 0.00 | 788 | 1 | 0.00 | 788 | 1 |
| | | 7 | 0.00 | 788 | 1 | 0.01 | 788 | 1 |
| | | 1 | T.O. | 1,776,727,111 | 517,954,091 | T.O. | 1,776,727,111 | 517,954,091 |
| | | 2 | 1.84 | 743,911 | 1,198 | 0.61 | 201,521 | 1,198 |
| C03690 | | 3 | 1.66 | 402,390 | 8 | 0.63 | 154,226 | 8 |
| | 25 | 4 | 0.91 | 150,565 | 4 | 0.44 | 76,612 | 4 |
| $C_{24}H_{38}O_4$ | | 5 | 0.71 | 96,781 | 2 | 0.37 | 54,747 | 2 |
| | | 6 | 0.45 | 50,750 | 1 | 0.24 | 28,877 | 1 |
| | | 7 | 0.35 | 38,761 | 1 | 0.16 | 20,269 | 1 |
| | | 1 | T.O. | 1,760,206,835 | 24,462,156 | T.O. | 1,760,206,835 | 24,462,156 |
| | | 2 | T.O. | 511,779,857 | 88,382 | T.O. | 360,354,641 | 125,693 |
| C13806 | | 3 | 132.92 | 22,906,789 | 360 | 171.57 | 21,545,743 | 360 |
| | 28 | 4 | 18.33 | 2,266,533 | 8 | 24.13 | 2,240,879 | 8 |
| $C_{23}H_{41}N_5O_3$ | | 5 | 7.16 | 725,648 | 2 | 9.50 | 721,618 | 2 |
| | | 6 | 3.64 | 296,846 | 1 | 4.92 | 296,015 | 1 |
| | | 7 | 1.77 | 131,350 | 1 | 2.43 | 131,169 | 1 |
| | | 1 | T.O. | 1,964,038,054 | 215,027,700 | T.O. | 1,964,038,054 | 215,027,700 |
| | | 2 | 476.01 | 164,518,561 | 2,520 | 89.15 | 20,918,476 | 2,520 |
| C04036 | | 3 | 16.00 | 3,478,952 | 1 | 14.17 | 2,121,663 | 1 |
| | 29 | 4 | 1.31 | 207,994 | 1 | 1.59 | 168,365 | 1 |
| $C_{19}H_{39}O_7P$ | | 5 | 0.89 | 100,001 | 1 | 1.04 | 85,232 | 1 |
| | | 6 | 0.68 | 57,164 | 1 | 0.74 | 50,114 | 1 |
| | | 7 | 0.57 | 38,265 | 1 | 0.62 | 34,736 | 1 |
| | | 1 | T.O. | 1,368,168,649 | 35,248,476 | T.O. | 1,368,168,649 | 35,248,476 |
| | | 2 | T.O. | 434,706,136 | N.F. | T.O. | 453,312,436 | 4,744 |
| C13829 | | 3 | 220.06 | 39,212,240 | 330 | 71.58 | 12,410,641 | 330 |
| | 32 | 4 | 9.44 | 1,289,820 | 6 | 7.38 | 897,569 | 6 |
| $C_{24}H_{41}NO_2$ | | 5 | 4.51 | 470,650 | 4 | 4.13 | 375,619 | 4 |
| | | 6 | 1.93 | 168,704 | 2 | 1.92 | 140,805 | 2 |
| | | 7 | 1.28 | 103,695 | 1 | 1.27 | 89,296 | 1 |
| | | 1 | T.O. | 1,694,837,933 | 20,888,287 | T.O. | 1,694,837,933 | 20,888,287 |
| | | 2 | T.O. | 521,186,457 | 3,502 | 1570.01 | 274,374,673 | 12,240 |
| C03630 | | 3 | 161.11 | 30,976,896 | 13 | 159.26 | 18,104,731 | 13 |
| | 33 | 4 | 10.65 | 1,316,554 | 11 | 12.81 | 1,052,571 | 11 |
| $C_{21}H_{39}O_7P$ | | 5 | 5.40 | 470,746 | 9 | 6.44 | 401,429 | 9 |
| | | 6 | 2.96 | 207,332 | 7 | 3.49 | 181,330 | 7 |
| | | 7 | 1.50 | 87,036 | 5 | 1.79 | 80,904 | 5 |

that must be appended to $T$ and any of its descendants for restoring the compound exceeds a precalculated limit.

Specifically, we first calculate the number $h^*(l)$ ($l \in \Sigma$) of hydrogens that must be appended to a vertex labeled $l$. It is easy to see that this can be done from the input feature vector of level 1 and the valence function. The H-cut checks if (a lower bound on) the number of hydrogens that must be appended to the $l$-labeled vertices in $T$ exceeds $h^*(l)$ for each $l \in \Sigma$. We use the following lower bound:

$$h(l;T) = \sum_{v \in T \backslash \mathrm{RP}(T), l(v)=l} (\mathrm{val}(l(v)) - \deg(v;T))$$

(Recall that $T$ and all descendants of $T$ in the family tree share the common structure of $T\backslash\mathrm{RP}(T)$.) See the illustration in Figure 10.

## EXPERIMENTAL RESULTS

In this section, we demonstrate the performance of our proposed algorithms. Tests were carried out on a Linux PC with an AMD Athlon 2.60 GHz CPU, using the instances constructed from some chemical compounds selected from the KEGG LIGAND database at http:/www.gnome.jp/kegg/ligand.html. Note that we treat a benzene ring contained in these compounds as a new virtual atom of valence six. Also, all instances for Problem 2 are preprocessed by the H-removal and the single-bond transformations. In the following, we will show the results of comparison in terms of label order, detachment-cut, and with earlier algorithms. It is especially noteworthy that the detachment-cut is very effective in the efficiency of enumeration, as shown in the results of the comparison with the earlier algorithms.

**Comparison in Label Order.** We figured out a way to raise the efficiency of the search by considering the label order in the branching operation and compared the cases when adopting and not adopting this scheme. In general, the computation efficiency of the branch-and-bound algorithm often varies with the search order of a tree. In the algorithm for enumerating treelike chemical compounds, the branching operation depends on the label of a vertex to be added to the partial tree, and elaborating the label order of such a vertex can make the search space small. More specifically, when the label of an input atom

**Table 3.** Comparison between Fujiwara et al.'s Algorithm $A$ and Our Algorithm $A_D$ for Problem 1[a]

| | | | Algorithm $A$ | | | Algorithm $A_D$ | | |
|---|---|---|---|---|---|---|---|---|
| entry formula | $n_1$ | $K$ | time (sec) | #nodes | #sol. | time (sec) | #nodes | #sol. |
| | | 1 | T.O. | 2,761,279,559 | N.F. | 286.11 | 29,658,916 | 70,170 |
| | | 2 | 26.06 | 15,827,372 | 16 | 2.80 | 185,593 | 16 |
| C07178 | | 3 | 2.27 | 915,962 | 2 | 0.29 | 20,741 | 2 |
| | 46 | 4 | 0.54 | 146,789 | 1 | 0.12 | 7,946 | 1 |
| $C_{21}H_{28}N_2O_5$ | | 5 | 0.58 | 123,251 | 1 | 0.12 | 6,776 | 1 |
| | | 6 | 0.67 | 118,295 | 1 | 0.12 | 6,660 | 1 |
| | | 7 | 0.64 | 93,947 | 1 | 0.13 | 6,177 | 1 |
| | | 1 | T.O. | 3,163,866,349 | N.F. | T.O. | 701,805,483 | N.F. |
| | | 2 | T.O. | 1,030,128,639 | N.F. | 107.86 | 20,491,818 | 1,198 |
| C03690 | | 3 | T.O. | 638,297,197 | N.F. | 52.81 | 8,668,135 | 8 |
| | 61 | 4 | T.O. | 463,342,181 | 1 | 8.20 | 1,081,724 | 4 |
| $C_{24}H_{38}O_4$ | | 5 | T.O. | 331,765,664 | 1 | 6.74 | 737,684 | 2 |
| | | 6 | T.O. | 222,811,935 | 1 | 3.14 | 264,785 | 1 |
| | | 7 | 763.86 | 77,002,582 | 1 | 2.17 | 169,071 | 1 |
| | | 1 | T.O. | 2,659,749,382 | N.F. | T.O. | 356,850,153 | N.F. |
| | | 2 | T.O. | 936,286,613 | N.F. | T.O. | 232,066,628 | N.F. |
| C13806 | | 3 | T.O. | 601,709,417 | N.F. | T.O. | 178,463,247 | 192 |
| | 67 | 4 | T.O. | 411,221,165 | N.F. | 331.22 | 29,749,320 | 8 |
| $C_{23}H_{41}N_5O_3$ | | 5 | T.O. | 296,677,899 | N.F. | 90.79 | 7,196,553 | 2 |
| | | 6 | T.O. | 248,139,066 | N.F. | 54.62 | 3,937,782 | 1 |
| | | 7 | T.O. | 184,624,435 | N.F. | 23.19 | 1,438,490 | 1 |
| | | 1 | T.O. | 2,536,517,458 | N.F. | T.O. | 804,294,593 | N.F. |
| | | 2 | T.O. | 1,057,838,606 | N.F. | T.O. | 465,994,772 | 720 |
| C04036 | | 3 | T.O. | 530,220,520 | N.F. | 415.31 | 81,239,220 | 1 |
| | 66 | 4 | T.O. | 350,726,961 | N.F. | 39.61 | 6,102,637 | 1 |
| $C_{19}H_{39}O_7P$ | | 5 | T.O. | 254,336,314 | N.F. | 26.41 | 3,208,595 | 1 |
| | | 6 | T.O. | 197,169,238 | N.F. | 23.44 | 2,373,205 | 1 |
| | | 7 | T.O. | 167,248,397 | N.F. | 21.57 | 1,901,333 | 1 |
| | | 1 | T.O. | 3,146,334,630 | N.F. | T.O. | 982,564,067 | N.F. |
| | | 2 | T.O. | 1,097,952,835 | N.F. | T.O. | 585,189,456 | N.F. |
| C13829 | | 3 | T.O. | 558,155,006 | N.F. | T.O. | 387,963,648 | N.F. |
| | 68 | 4 | T.O. | 372,345,257 | N.F. | T.O. | 290,019,992 | N.F. |
| $C_{24}H_{41}NO_2$ | | 5 | T.O. | 259,625,733 | N.F. | T.O. | 206,873,646 | N.F. |
| | | 6 | T.O. | 197,777,048 | N.F. | T.O. | 164,523,040 | N.F. |
| | | 7 | T.O. | 159,902,472 | N.F. | T.O. | 137,165,879 | N.F. |
| | | 1 | T.O. | 2,635,645,408 | N.F. | T.O. | 808,369,464 | N.F. |
| | | 2 | T.O. | 1,103,007,536 | N.F. | T.O. | 531,093,972 | N.F. |
| C03630 | | 3 | T.O. | 605,089,731 | N.F. | T.O. | 307,935,706 | 9 |
| | 68 | 4 | T.O. | 350,596,933 | N.F. | 1066.66 | 143,223,737 | 11 |
| $C_{21}H_{39}O_7P$ | | 5 | T.O. | 252,457,654 | N.F. | 624.50 | 68,644,407 | 9 |
| | | 6 | T.O. | 184,868,704 | N.F. | 544.56 | 49,670,247 | 7 |
| | | 7 | T.O. | 154,332,627 | N.F. | 499.72 | 39,596,054 | 5 |

[a] Note: $n_1$ is the number of vertices in an instance preprocessed by replacing each benzene ring with a new atom having six valences.

with high occurrence has priority over that of an atom with low occurrence, the number of vertices to be searched tends to fall, which will lead to a decrease in computation time. Table 1 shows the results of such a comparison for Problem 2. We can see a speed-up in computation for all instances when adopting the prioritized label order, especially for small $K$. It is noteworthy that only the results for Problem 2 are reported here since the Problem 2 formulation with hydrogen removal can handle larger sizes of instances than the Problem 1 formulation does, which can reveal remarkable improvement. This also holds for the case when using the detachment-cut for $K = 2$ described below.

**Comparison in Detachment-Cut for $K = 2$.** After prioritizing the label order, we compared the cases when using and not using the detachment-cut for level $K = 2$, which is shown in Table 2. It must be noted that both cases make use of the detachment-cut for $K = 1$, and this comparison is intended to investigate the effect of using the detachment-cut for $K = 2$. We should also notice that this detachment-cut cannot be applied to an instance where $K = 1$ since information on the feature vector of length two is necessary. As the table shows, the detachment-cut pays off in computation time for compounds with a large number of atoms without a benzene ring when $K = 2$ and 3 (see the results on the last three instances C04036, C13829, and C03630). However, the computation with the detachment-cut for C13806 takes slightly longer time than that without the detachment-cut.

**Comparison with Fujiwara et al.'s Algorithms.** We made a comparison between our algorithms (called Algorithms $A_D$ and $B_D$ for Problems 1 and 2, respectively) and Fujiwara et al.'s algorithms[17] (called Algorithms $A$ and $B$ for Problems 1 and 2, respectively) in terms of computation time. Note that Algorithms $A_D$ and $B_D$ used in the comparison adopt the detachment-cut for $K = 2$ as well as the prioritized label order. As Tables 3 and 4 show, our proposed algorithms run significantly faster than Fujiwara et al.'s algorithms do for almost all instances. This shows that the detachment-cut operation works well to improve enumeration efficiency.

**Table 4.** Comparison between Fujiwara et al.'s Algorithm $B$ and Our Algorithm $B_D$ for Problem 2[a]

| | | | Algorithm $B$ | | | Algorithm $B_D$ | | |
|---|---|---|---|---|---|---|---|---|
| Entry Formula | $n_2$ | $K$ | time (sec) | #nodes | #sol. | time (sec) | #nodes | #sol. |
| | | 1 | 102.49 | 96,006,467 | 70,170 | 3.34 | 2,754,980 | 70,170 |
| | | 2 | 0.05 | 21,460 | 16 | 0.02 | 3,994 | 16 |
| C07178 | | 3 | 0.04 | 11,950 | 2 | 0.02 | 2,972 | 2 |
| | 19 | 4 | 0.02 | 3,152 | 1 | 0.00 | 1,065 | 1 |
| $C_{21}H_{28}N_2O_5$ | | 5 | 0.01 | 2,143 | 1 | 0.01 | 813 | 1 |
| | | 6 | 0.02 | 2,088 | 1 | 0.00 | 788 | 1 |
| | | 7 | 0.02 | 2,088 | 1 | 0.01 | 788 | 1 |
| | | 1 | T.O. | 1,451,017,726 | 13,416,358 | T.O. | 1,776,727,111 | 517,954,091 |
| | | 2 | 11.42 | 2,984,162 | 1,198 | 0.61 | 201,521 | 1,198 |
| C03690 | | 3 | 8.70 | 1,464,436 | 8 | 0.63 | 154,226 | 8 |
| | 25 | 4 | 4.16 | 509,870 | 4 | 0.44 | 76,612 | 4 |
| $C_{24}H_{38}O_4$ | | 5 | 2.89 | 283,418 | 2 | 0.37 | 54,747 | 2 |
| | | 6 | 1.55 | 132,434 | 1 | 0.24 | 28,877 | 1 |
| | | 7 | 1.25 | 101,097 | 1 | 0.16 | 20,269 | 1 |
| | | 1 | T.O. | 1,107,551,121 | 1,787,759 | T.O. | 1,760,206,835 | 24,462,156 |
| | | 2 | T.O. | 375,837,782 | N.F. | T.O. | 360,354,641 | 125,693 |
| C13806 | | 3 | 658.59 | 93,742,772 | 360 | 171.57 | 21,545,743 | 360 |
| | 28 | 4 | 62.98 | 6,445,846 | 8 | 24.13 | 2,240,879 | 8 |
| $C_{23}H_{41}N_5O_3$ | | 5 | 18.61 | 1,578,420 | 2 | 9.50 | 721,618 | 2 |
| | | 6 | 8.40 | 564,758 | 1 | 4.92 | 296,015 | 1 |
| | | 7 | 3.47 | 219,050 | 1 | 2.43 | 131,169 | 1 |
| | | 1 | T.O. | 1,558,179,957 | 7,045,290 | T.O. | 1,964,038,054 | 215,027,700 |
| | | 2 | T.O. | 450,784,492 | 1,735 | 89.15 | 20,918,476 | 2,520 |
| C04036 | | 3 | 104.13 | 14,517,014 | 1 | 14.17 | 2,121,663 | 1 |
| | 29 | 4 | 7.29 | 638,457 | 1 | 1.59 | 168,365 | 1 |
| $C_{19}H_{39}O_7P$ | | 5 | 3.64 | 225,966 | 1 | 1.04 | 85,232 | 1 |
| | | 6 | 2.68 | 127,250 | 1 | 0.74 | 50,114 | 1 |
| | | 7 | 2.06 | 81,532 | 1 | 0.62 | 34,736 | 1 |
| | | 1 | T.O. | 1,172,000,340 | N.F. | T.O. | 1,368,168,649 | 35,248,476 |
| | | 2 | T.O. | 274,949,165 | N.F. | T.O. | 453,312,436 | 4,744 |
| C13829 | | 3 | 929.29 | 113,381,773 | 330 | 71.58 | 12,410,641 | 330 |
| | 32 | 4 | 25.22 | 2,606,092 | 6 | 7.38 | 897,569 | 6 |
| $C_{24}H_{41}NO_2$ | | 5 | 11.18 | 894,669 | 4 | 4.13 | 375,619 | 4 |
| | | 6 | 3.95 | 276,201 | 2 | 1.92 | 140,805 | 2 |
| | | 7 | 2.30 | 155,976 | 1 | 1.27 | 89,296 | 1 |
| | | 1 | T.O. | 1,501,238,732 | 11,877 | T.O. | 1,694,837,933 | 20,888,287 |
| | | 2 | T.O. | 318,970,855 | 116 | 1570.01 | 274,374,673 | 12,240 |
| C03630 | | 3 | 1655.00 | 189,813,489 | 13 | 159.26 | 18,104,731 | 13 |
| | 33 | 4 | 69.13 | 5,104,899 | 11 | 12.81 | 1,052,571 | 11 |
| $C_{21}H_{39}O_7P$ | | 5 | 30.95 | 1,554,928 | 9 | 6.44 | 401,429 | 9 |
| | | 6 | 17.02 | 673,426 | 7 | 3.49 | 181,330 | 7 |
| | | 7 | 7.50 | 244,166 | 5 | 1.79 | 80,904 | 5 |

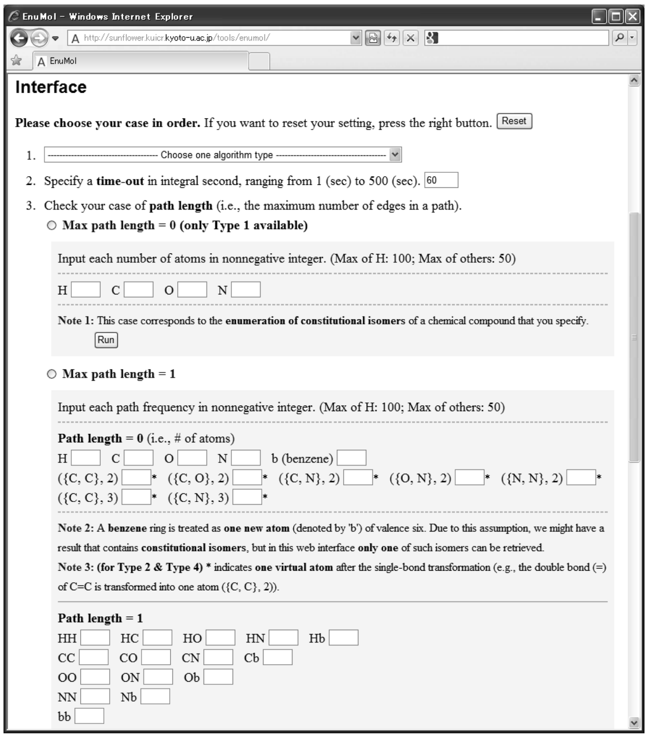
[a] Note: $n_2$ is the number of vertices in an instance preprocessed by replacing each benzene ring with a new atom having six valences and by the H-removal and single-bond transformations.

## WEB SERVER

We provide EnuMol, a Web server for ENUmerating treelike MOLecular structures from a given path frequency, implementing our proposed algorithms. EnuMol is available at http://sunflower.kuicr.kyoto-u.ac.jp/tools/enumol/, and a user can choose either Algorithm $A_D$ for Problem 1 or Algorithm $B_D$ for Problem 2 in the input page. Moreover, the user has an option to run the algorithm according to the value of the level $K$ for a feature vector. If $K = 0$ or $K = 1$, each element of the feature vector can be directly input into the form on the Web interface. In particular, the case where $K = 0$ corresponds to enumerating constitutional isomers of chemical compounds consistent with the numbers of atoms that the user specifies (i.e., the molecular formula). If $K \geq 2$, a feature vector file must be uploaded. Figure 11 shows a partial view of EnuMol for input. After the server finishes the computation, we can obtain chemical compounds in SMILES (simplified molecular input line entry specification) format if any (see Figure 12). Note that SMILES is a simplified chemical language that represents a molecular structure as a labeled graph, and thus it is essentially the two-dimensional diagram of the molecule. In the output page, we can also view the two-dimensional diagram of a set of SMILES (see Figure 13).

## CONCLUSION

We considered the problem of enumerating all treelike chemical graphs from a given feature vector based on frequencies of paths and augmented the existing branch-and-bound algorithm by introducing two new bounding operations (the detachment-cut and the H-cut). The previous branch-and-bound algorithm by Fujiwara et al.[17] employed only bounding operations that investigate whether the current tree $T$ violates given structural conditions, whereas our new bounding operation (detachment-cut) tests whether the current tree $T$ can have a solution as its descendant using information on the nodes/edges that have not been introduced


EnuMol - Windows Internet Explorer
http://sunflower.kuicr.kyoto-u.ac.jp/tools/enumol/
EnuMol

Interface

**Please choose your case in order.** If you want to reset your setting, press the right button. Reset

1. ---------- Choose one algorithm type ----------
2. Specify a **time-out** in integral second, ranging from 1 (sec) to 500 (sec). 60
3. Check your case of **path length** (i.e., the maximum number of edges in a path).

○ **Max path length = 0 (only Type 1 available)**

Input each number of atoms in nonnegative integer. (Max of H: 100; Max of others: 50)

H C O N

**Note 1:** This case corresponds to the **enumeration of constitutional isomers** of a chemical compound that you specify.

Run

○ **Max path length = 1**

Input each path frequency in nonnegative integer. (Max of H: 100; Max of others: 50)

**Path length = 0** (i.e., # of atoms)

H C O N b (benzene)

({C, C}, 2) * ({C, O}, 2) * ({C, N}, 2) * ({O, N}, 2) * ({N, N}, 2) *

({C, C}, 3) * ({C, N}, 3) *

**Note 2:** A **benzene** ring is treated as **one new atom** (denoted by 'b') of valence six. Due to this assumption, we might have a result that contains **constitutional isomers**, but in this web interface **only one** of such isomers can be retrieved.

**Note 3: (for Type 2 & Type 4) *** indicates **one virtual atom** after the single-bond transformation (e.g., the double bond (=) of C=C is transformed into one atom ({C, C}, 2)).

**Path length = 1**

HH HC HO HN Hb

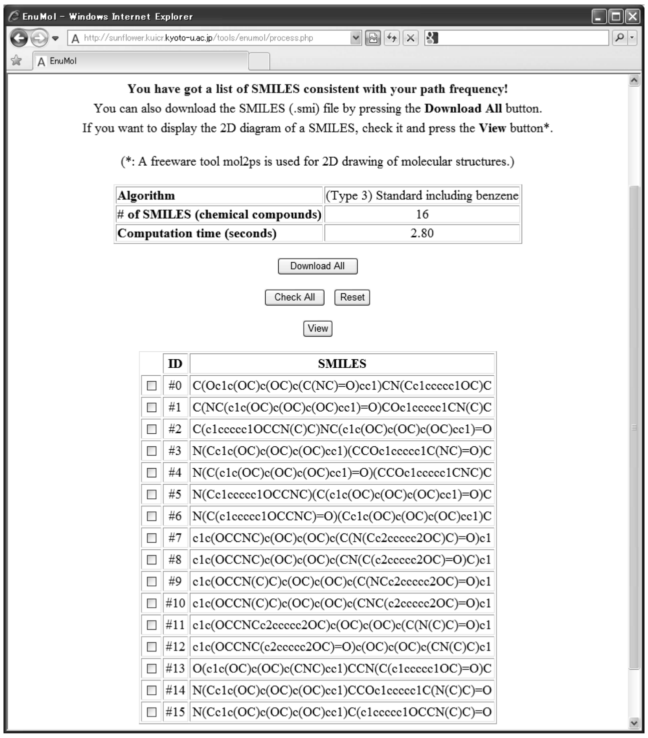
CC CO CN Cb

OO ON Ob

NN Nb

bb

**Figure 11.** Partial view of EnuMol for input.


EnuMol - Windows Internet Explorer
http://sunflower.kuicr.kyoto-u.ac.jp/tools/enumol/process.php
EnuMol
Download All
Check All
Reset
View

**You have got a list of SMILES consistent with your path frequency!**
You can also download the SMILES (.smi) file by pressing the **Download All** button.
If you want to display the 2D diagram of a SMILES, check it and press the **View** button*.

(*: A freeware tool mol2ps is used for 2D drawing of molecular structures.)

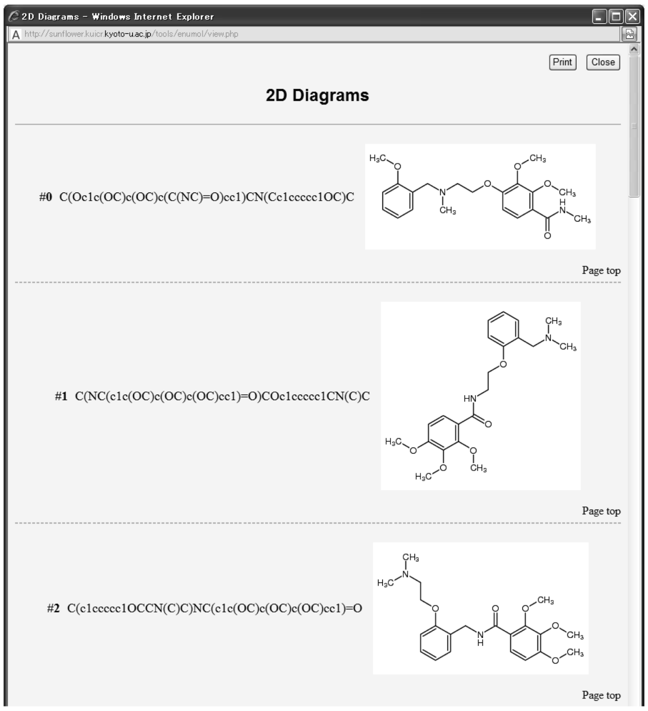
| | |
|---|---|
| **Algorithm** | (Type 3) Standard including benzene |
| **# of SMILES (chemical compounds)** | 16 |
| **Computation time (seconds)** | 2.80 |

| | ID | SMILES |
|---|---|---|
| ☐ | #0 | C(Oc1c(OC)c(OC)c(C(NC)=O)cc1)CN(Cc1ccccc1OC)C |
| ☐ | #1 | C(NC(c1c(OC)c(OC)c(OC)cc1)=O)COc1ccccc1CN(C)C |
| ☐ | #2 | C(c1ccccc1OCCN(C)C)NC(c1c(OC)c(OC)c(OC)cc1)=O |
| ☐ | #3 | N(Cc1c(OC)c(OC)c(OC)cc1)(CCOc1ccccc1C(NC)=O)C |
| ☐ | #4 | N(C(c1c(OC)c(OC)c(OC)cc1)=O)(CCOc1ccccc1CNC)C |
| ☐ | #5 | N(Cc1ccccc1OCCNC)(C(c1c(OC)c(OC)c(OC)cc1)=O)C |
| ☐ | #6 | N(C(c1ccccc1OCCNC)=O)(Cc1c(OC)c(OC)c(OC)cc1)C |
| ☐ | #7 | c1c(OCCNC)c(OC)c(OC)c(C(N(Cc2ccccc2OC)C)=O)c1 |
| ☐ | #8 | c1c(OCCNC)c(OC)c(OC)c(CN(C(c2ccccc2OC)=O)C)c1 |
| ☐ | #9 | c1c(OCCN(C)C)c(OC)c(OC)c(C(NCc2ccccc2OC)=O)c1 |
| ☐ | #10 | c1c(OCCN(C)C)c(OC)c(OC)c(CNC(c2ccccc2OC)=O)c1 |
| ☐ | #11 | c1c(OCCNCc2ccccc2OC)c(OC)c(OC)c(C(N(C)C)=O)c1 |
| ☐ | #12 | c1c(OCCNC(c2ccccc2OC)=O)c(OC)c(OC)c(CN(C)C)c1 |
| ☐ | #13 | O(c1c(OC)c(OC)c(CNC)cc1)CCN(C(c1ccccc1OC)=O)C |
| ☐ | #14 | N(Cc1c(OC)c(OC)c(OC)cc1)CCOc1ccccc1C(N(C)C)=O |
| ☐ | #15 | N(Cc1c(OC)c(OC)c(OC)cc1)C(c1ccccc1OCCN(C)C)=O |

**Figure 12.** Partial view of EnuMol for output, showing a list of SMILES.

in the tree *T*. Our experimental results show that our algorithm outperforms Fujiwara et al.'s algorithm.

Remember that our algorithm can deal with treelike structures, not cyclic ones. A benzene ring has been treated as one virtual atom of valence six, but the algorithm may not work for more complex cyclic structures. To cover a wider class of chemical graphs, we will consider enumerating the more general class of graphs, including outerplanar graphs that are known to cover most of the chemical


2D Diagrams - Windows Internet Explorer
http://sunflower.kuicr.kyoto-u.ac.jp/tools/enumol/view.php
Print
Close

**2D Diagrams**

**#0** C(Oc1c(OC)c(OC)c(C(NC)=O)cc1)CN(Cc1ccccc1OC)C

Page top

**#1** C(NC(c1c(OC)c(OC)c(OC)cc1)=O)COc1ccccc1CN(C)C

Page top

**#2** C(c1ccccc1OCCN(C)C)NC(c1c(OC)c(OC)c(OC)cc1)=O

Page top

**Figure 13.** Partial view of EnuMol for output, showing a list of 2D diagrams.

graphs,[28] which is left as our future work. In addition, the depth label sequences introduced in this work describe only the graphical structures of compounds from the viewpoint of planarity and thus may lose information of stereochemistry.[29] Therefore, designing better representation is another interesting task. Furthermore, we would like to consider enumeration under constraints other than path frequency, where it is important to investigate what kind of constraint works out.

## ACKNOWLEDGMENT

The authors thank Mr. Hiroki Fujiwara and Miss Jiexun Wang for their helpful discussions. The authors also thank three anonymous reviewers for giving helpful comments, which greatly helped improve the quality of the paper. This work was supported in part by Grant-in-Aid for Scientific Research (KAKENHI) #19200022 from the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan.

## REFERENCES AND NOTES

(1) Cayley, A. On the Analytic Forms Called Trees with Applications to the Theory of Chemical Combinations. *Rep. Brit. Assoc. Adv. Sci.* **1875**, *45*, 257–305.
(2) Balaban, T. S.; Filip, P. A.; Ivanciuc, O. Computer Generation of Acyclic Graphs Based on Local Vertex Invariants and Topological Indices. Derived Canonical Labeling and Coding of Trees and Alkanes. *J. Math. Chem.* **1992**, *11*, 79–105.
(3) Jackson, M. D.; Bieber, T. I. Applications of Degree Distribution. 2. Construction and Enumeration of Isomers in the Alkane Series. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 701–708.
(4) Knop, J. V.; Müeller, W. R.; Jeričević, Z.; Trinajstić, N. Computer Enumeration and Generation of Trees and Rooted Trees. *J. Chem. Inf. Comput. Sci.* **1981**, *21*, 91–99.
(5) Masinter, L. M.; Sridharan, N. S.; Lederberg, J.; Smith, D. H. Applications of Artificial Intelligence for Chemical Inference. XII. Exhaustive Generation of Cyclic and Acyclic Isomers. *J. Am. Chem. Soc.* **1974**, *96*, 7702–7714.
(6) Trinajstić, N.; Nicolić, S.; Knop, J. V.; Müller, W. R.; Szymanski, K. *Computational Chemical Graph Theory: Characterization, Enumera-*

*tion and Generation of Chemical Structures by Computer Methods*; Simon Schuster/Horwood: Chichester, U. K., 1991.

(7) Faulon, J.-L.; Churchwell, C. J.; Visco, D. P., Jr. The Signature Molecular Descriptor. 2. Enumerating Molecules from Their Extended Valence Sequences. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 721–734.

(8) Hall, L. H.; Dailey, R. S.; Kier, L. B. Design of Molecules from Quantitative Structure Activity Relationship Models. 3. Role of Higher Order Path Counts: Path 3. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 598–603.

(9) Deshpande, M.; Kuramochi, M.; Wale, N.; Karypis, G. Frequent Substructure-Based Approaches for Classifying Chemical Compounds. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 1036–1050.

(10) Fink, T.; Reymond, J. L. Virtual Exploration of the Chemical Universe up to 11 Atoms of C, N, O, F: Assembly of 26.4 Million Structures (110.9 Million Stereoisomers) and Analysis for New Ring Systems, Stereochemistry, Physicochemical Properties, Compound Classes, and Drug Discovery. *J. Chem. Inf. Comput. Sci.* **2007**, *47*, 342–353.

(11) Mauser, H.; Stahl, M. Chemical Fragment Spaces for De Novo Design. *J. Chem. Inf. Comput. Sci.* **2007**, *47*, 318–324.

(12) Akutsu, T.; Fukagawa, D. Inferring a Chemical Structure from a Feature Vector Based on Frequency of Labeled Paths and Small Fragments. In *Series on Advances in Bioinformatics and Computational Biology, Proceedings of the Fifth Asia Pacific Bioinformatics Conference*, Hong Kong, China, January 15−17, 2007; Sankoff, D., Wang, L., Chin, F., Eds.; Imperial College Press: London, 2007; pp 165−174.

(13) Bakır, G. H.; Zien, A.; Tsuda, K. Learning to Find Graph Pre-Images. *Lect. Notes Comput.Sci* **2004**, *3175*, 253–261.

(14) Mahé, P.; Ueda, N.; Akutsu, T.; Perret, J. L.; Vert, J. P. Graph Kernels for Molecular Structure Activity Relationship Analysis with Support Vector Machines. *J. Chem. Inf. Model.* **2005**, *45*, 939–951.

(15) Kashima, H.; Tsuda, K.; Inokuchi, A. Marginalized Kernels between Labeled Graphs. *Proceedings of 20th International Conference on Machine Learning*, Washington, DC, August 21−24, 2003; Fawcett, T., Mishra, N., Eds.; The AAAI Press: Menlo Park, CA, 2003; pp 321−328.

(16) Akutsu, T.; Fukagawa, D. Inferring a Graph from Path Frequency. *Lect. Notes Comput. Sci.* **2005**, *3537*, 371–382.

(17) Fujiwara, H.; Wang, J.; Zhao, L.; Nagamochi, H.; Akutsu, T. Enumerating Treelike Chemical Graphs with Given Path Frequency. *J. Chem. Inf. Model.* **2008**, *48*, 1345–1357.

(18) Nakano, S.; Uno, T. *Efficient Generation of Rooted Trees*; Technical Report NII-2003−005E, National Institute of Informatics: Tokyo, Japan, 2003.

(19) Nakano, S.; Uno, T. Generating Colored Trees. *Lect. Notes Comput. Sci.* **2005**, *3787*, 249–260.

(20) Kier, L. B.; Hall, L. H.; Frazer, J. W. Design of Molecules from Quantitative Structure-Activity Relationship Models. 1. Information Transfer between Path and Vertex Degree Counts. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 143–147.

(21) Skvortsova, M. I.; Baskin, I. I.; Slovokhotova, O. L.; Palyulin, V. A.; Zefirov, N. S. Inverse Problem in QSAR/QSPR Studies for the Case of Topological Indices Characterizing Molecular Shape (Kier Indices). *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 630–634.

(22) Churchwell, C. J.; Rintoul, M. D.; Martin, S.; Visco, D. P., Jr.; Kotu, A.; Larson, R. S.; Sillerud, L. O.; Brown, D. C.; Faulon, J.-L. The Signature Molecular Descriptor: 3. Inverse-Quantitative Structure-Activity Relationship of ICAM-1 Inhibitory Peptides. *J. Mol. Graphics Model.* **2004**, *22*, 263–273.

(23) Brown, W. M.; Martin, S.; Rintoul, M. D.; Faulon, J.-L. Designing Novel Polymers with Targeted Properties Using the Signature Molecular Descriptor. *J. Chem. Inf. Model.* **2006**, *46*, 826–835.

(24) Nagamochi, H. A Detachment Algorithm for Inferring a Graph from Path Frequency. *Algorithmica* **2009**, *53*, 207–224.

(25) Ishida, Y.; Zhao, L.; Nagamochi, H.; Akutsu, T. Improved Algorithms for Enumerating Tree Like Chemical Graphs with Given Path Frequency. *Genome Inf. Ser.* **2008**, *21*, 53–64.

(26) Aringhieri, R.; Hansen, P.; Malucelli, F. Chemical Trees Enumeration Algorithms. *4OR. Quart. J. Oper. Res.* **2003**, *1*, 67–83.

(27) Jordan, C. Sur Les Assemblages De Lignes. *J. Reine Angew. Math.* **1869**, *70*, 185–190.

(28) Wang, J.; Zhao, L.; Nagamochi, H.; Akutsu, T. An Efficient Algorithm for Generating Colored Outerplanar Graphs. *Lect. Notes Comput. Sci.* **2007**, *4484*, 573–583.

(29) Shungo, K.; Iwata, S.; Uno, T.; Koshino, H.; Satoh, H. Algorithm for Advanced Canonical Coding of Planar Chemical Structures that Considers Stereochemical and Symmetric Information. *J. Chem. Inf. Model.* **2007**, *47*, 1734–1746.

CI900447Z