Efficient Enumeration of Stereoisomers of Tree Structured Molecules Using Dynamic Programming

Tomoki Imada, Shunsuke Ota,
Hiroshi Nagamochi

Graduate School of Informatics,
Kyoto University
{imada,ota,nag}@amp.i.kyoto-u.ac.jp

Tatsuya Akutsu

Bioinformatics Center,
Institute for Chemical Research,
Kyoto University
takutsu@kuicr.kyoto-u.ac.jp

Abstract

Nonredundant and exhaustive generation of stereoisomers of a chemical compound with a specified constitution is one of the important tools for molecular structure elucidation and molecular design. In this paper, we deal with chemical compounds composed of carbon, hydrogen, oxygen and nitrogen atoms whose graphical structures are tree-like graphs because these compounds are most fundamental, and consider stereoisomers that can be generated by asymmetric carbon atoms and double bonds between two adjacent carbon atoms. Based on dynamic programming, we propose an algorithm of generating all stereoisomers without duplication. We treat a given tree-like graph as a tree rooted at its structural center. Our algorithm first computes recursively the numbers of stereoisomers of the subgraphs induced by the descendants of each vertex, and then constructs each stereoisomer by backtracking the process of computing the numbers of stereoisomers. Our algorithm correctly counts the number of stereoisomers in $O(n)$ time and space, and correctly enumerates all the stereoisomers in $O(n)$ space and in $O(n)$ time per stereoisomer, where n is the number of atoms in a given structure. The source code of the program implementing the proposed algorithm is freely available for academic use upon request.

1 Introduction

One of the most fundamental and important problems in chemoinformatics is nonredundant and exhaustive enumeration of isomers/stereoisomers because it plays core roles in structure elucidation and molecular design [5]. Since Cayley studied enumeration of alkanes in the 19th century [2], extensive studies have been done, which include Pólya's seminal work on counting the number of isomers using group theory [8, 9]. Two chemical compounds with the same isomer may have different three-dimensional configurations due to asymmetry around carbon atoms and many other structural asymmetries. Stereoisomers often exhibit different chemical properties, and synthesis of a specific stereoisomer remains a challenging issue in chemistry. Hence, enumeration of stereoisomers is important as well as enumeration of isomers.

In this paper, we consider stereoisomers caused only by asymmetry around carbon atoms. Such stereoisomers might be further divided into more detailed classes according to their three-dimensional conformations and stabilities. However, if the combinatorial structures based on asymmetry around carbon atoms are different, then the stereoisomers are considered different in any definition. Then stereoisomers caused only by asymmetry around carbon atoms are fundamental and practically important. As to enumeration of such stereoisomers, several methods have been proposed [1, 3, 12], which mostly follow the work by Nourse [7]. Given a chemical compound with m stereocenters, these methods first create a list of all 2^m combinations of the two choices of asymmetries around each carbon atom, and remove each set S of combinations that represent the same stereoisomer leaving one of them as their representative. Although such a set S of combinations can be constructed in $O(|S|m)$ time by a method on permutation groups called the *configuration groups*, the time and space complexity of the entire algorithm is $\Omega(2^m)$, which is always exponential even if the number of stereoisomers is any small. Furthermore, mathematical proofs for the correctness of some of these methods are not fully provided, where the correctness means that an algorithm does not miss any of the stereoisomers and does not output (or count) any of identical structures multiple times. Therefore, in order to provide examples for checking the validity of existing programs, Rücker et al. manually counted the number of stereoisomers of several chemical compounds [11].

In this paper, we focus on *tree structured molecules* (i.e., acyclic molecules) and develop algorithms for enumerating stereoisomers with guaranteed computational complexity. Differently from the existing approaches based on configuration groups, we use *dynamic programming*. For this, we treat a given tree structured molecule as a tree rooted at its structural center, and derive recursive formulas for the numbers of stereoisomers of rooted subtrees. However, it is nontrivial to represent stereoisomers with a mathematically consistent form, without which such recursive formulas cannot be derived. The main contribution of this paper is to give a mathematical representation for stereoisomers by introducing a new notion, "orientation of carbon circuits," and to design a dynamic programming algorithm that counts the total number K of stereoisomers of a given tree based on the derived recursive formulas and a traceback algorithm that constructs the k-th stereoisomer of the tree for each $k = 1, 2, \ldots, K$, by identifying the stereoisomer of each subtree corresponding to the k-th stereoisomer. Assuming that each of the four arithmetic operations can be done in constant time, our algorithm correctly counts the number K of stereoisomers in $O(n)$ time and space, and correctly enumerates all K stereoisomers without duplication in $O(n)$ space and in $O(n)$ time per stereoisomer, where n is the number of atoms in a given tree. The time complexity for counting is optimal. The time complexity for enumerating all stereoisomers is $O(nK)$, and this is also optimal provided that each stereoisomer needs to be output explicitly in $O(n)$ time. The computational key property to achieve the latter result is an efficient bijection algorithm, which is required as a subroutine of our enumeration algorithm. More specifically we show that, given integers $p \in \{1, 2, 3, 4\}$ and $n \geq p$, there is an $O(1)$ time algorithm that delivers the k-th set from the $\binom{n}{p}$ sets of p distinct integers $\{k_1 \in \{1, 2, \ldots, n\}, k_2 \in \{1, 2, \ldots, n\}, \ldots, k_p \in \{1, 2, \ldots, n\}\}$ for a specified integer $k \in \{1, 2, \ldots, \binom{n}{p}\}$. We conducted computational experiments to evaluate the practical computation time of the proposed algorithm. The results confirm that our proposed algorithm is very fast in practice for both counting and enumeration.

2 Preliminary and problem formulation

2.1 Problem definition

In this paper, we deal with the problem defined as follows.

Input A tree-like chemical graph whose vertex set V consists of carbon, hydrogen, oxygen and nitrogen atoms. A *vertex-number* $n : V \to \{1, 2, \ldots, |V|\}$, by which each vertex is numbered from 1 to $|V|$.

Output All the *stereoisomers* that can be generated by asymmetry around carbon atoms (the exact definition of stereoisomers in this paper is given in Section 2.4).

We denote a given chemical graph by $G = (V, E)$ with a vertex set V and an edge set E. The vertex set V is partitioned into $V_{\mathrm{C}} = \{v \mid v \text{ is a carbon atom}\}$, $V_{\mathrm{H}} = \{v \mid v \text{ is a hydrogen atom}\}$, $V_{\mathrm{O}} = \{v \mid v \text{ is an oxygen atom}\}$ and $V_{\mathrm{N}} = \{v \mid v \text{ is a nitrogen atom}\}$. We denote $|V| = n$. Multiple edges are treated as one single edges and the edge set E is partitioned into $E_1 = \{e \mid e \text{ is a single bond}\}$, $E_2 = \{e \mid e \text{ is a double bond}\}$ and $E_3 = \{e \mid e \text{ is a triple bond}\}$.

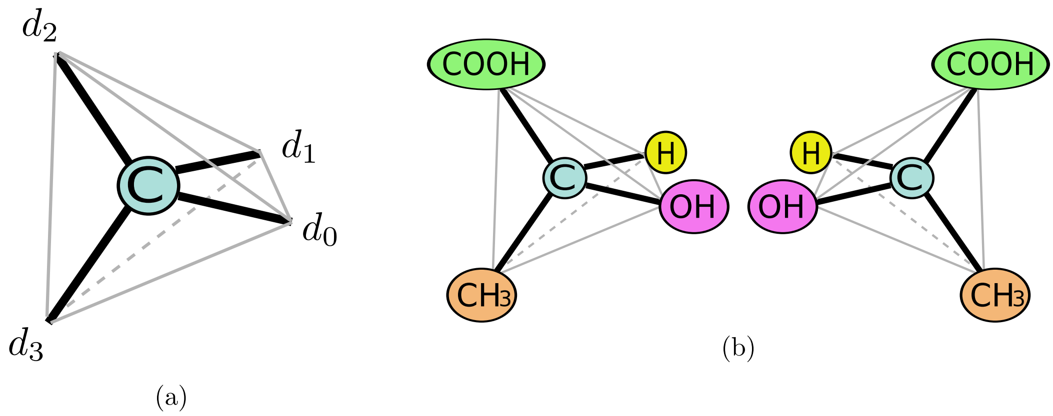


Figure 1: (a) The four directions d_0, d_1, d_2 and d_3 around a carbon atom in the three-dimensional space. (b) Configurations around the asymmetric carbon atom in lactic acid. There are two different configurations around the asymmetric carbon atoms (the carbon atom at the center of the tetrahedron).

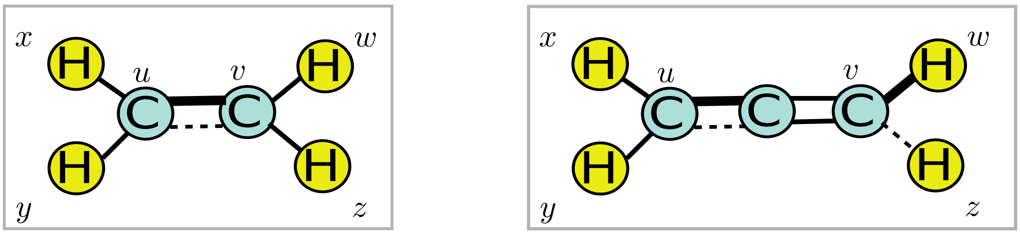
Fig. 1(a) illustrates that the three-dimensional structure around a carbon atom forms a regular tetrahedron, where d_0, d_1, d_2 and d_3 represent the directions along the four edges incident to the carbon atom. We define the *configuration* around a carbon atom v as a correspondence between the edges incident to v and d_i ($i = 0, 1, 2, 3$), where we do not distinguish two correspondences which result in the same stereoisomorphic (stereochemically isomorphic) compounds. For example, we consider that there is only one configuration around v if it is adjacent to an atom by a triple bound. Informally, we consider that there are two different configuration around v only when one of the following cases occurs:
(i) v is adjacent to four different substructures;
(ii) v is adjacent to a substructure T_1 by a double bond and two different substructures T_2 and T_3 by single bonds, and T_1 is not symmetric along the double bound; and
(iii) v is adjacent to two substructures T_1 and T_2 by double bonds, and each T_i, $i = 1, 2$ is not symmetric along the double bound.
The exact relationship between configurations and stereoisomers will be given in Section 2.5. For example, there are two different configurations around the asymmetric carbon atom in lactic acid (see Fig. 1(b)).

We here show our assumption on the three-dimensional structure of a chain of double bonds between two carbon atoms u and v such that u is adjacent to two atoms x and y by single bonds and v is adjacent to two atoms w and z by single bonds, as shown in Fig. 2. For the number k of double bonds between u and v, we assume that
• x, y, w and z are on the same plane when k is odd; and
• x, y, w and z are not on the same plane when k is even.
For example, Fig. 2(a) and (b) illustrate the chain of double bonds of ethylene ($k = 1$) and allene ($k = 2$), respectively.

2.2 Isomorphism of tree-like graphs

Our algorithm first detects the *centroid* of a given tree-like graph G. For any tree, the next theorem specifies a structurally unique vertex or edge.

Theorem 1 (Jordan's theorem [6]). *For any tree of $n \geq 1$ vertices, exactly one of the next two statements holds.*
1. There exists a unique vertex v^* such that each of the subtrees obtained by removing v^* contains at most $\lceil (n-1)/2 \rceil$ vertices.



(a) Ethylene (b) Allene

Figure 2: Configurations around a chain of double bonds between two carbon atoms u and v. The rectangle shows the plane that contains the left two hydrogen atoms x and y. Thick lines indicate edges on the front side of the plane and dashed lines indicate edges on the back side of the plane.

2. There exists a unique edge e^* such that each of the two subtrees obtained by removing e^* contains $n/2$ vertices. □

Such a vertex v^* and an edge e^* are called the *unicentroid* and *bicentroid* of the tree, respectively. We call the unicentroid and bicentroid the *centroid* of the tree. The *root* of the tree is defined by the vertex/vertices in its centroid. For every vertex $v \in V$ except for the root, we define the *parent* of v as the vertex adjacent to v which is nearer to the root than v. For each vertex $v \in V$, $Ch(v)$ denotes the set of the children of vertex v, and the *rooted tree* T_v is defined to be the tree induced by v and all descendants of v.

The set of vertices and the set of edges of a graph G are also denoted by $V(G)$ and $E(G)$, respectively. Two chemical graphs G_1 and G_2 are called *isomorphic* if there is a bijection $\psi : V(G_1) \to V(G_2)$ such that $(u, v) \in E(G_1)$ if and only if $(\psi(u), \psi(v)) \in E(G_2)$, where the types of atoms of u (resp., v) and $\psi(u)$ (resp., $\psi(v)$) are identical, and the types of bonds of (u, v) and $(\psi(u), \psi(v))$ are identical. Such a bijection is called an *isomorphism* of G_1 and G_2. For two rooted subtrees T_u and T_v, we say that T_u and T_v are *rooted-isomorphic* if there is an isomorphism ψ between T_u and T_v such that $\psi(u) = v$. If T_u and T_v are rooted-isomorphic, then we write this as $T_u \underset{r}{\approx} T_v$.

For each subtree T_v, we write $\sigma(v, T_v)$ to refer to the *signature* of the subtree T_v, that is a non-negative integer satisfying a property that

$$\sigma(v, T_v) = \sigma(u, T_u) \Leftrightarrow T_v \underset{r}{\approx} T_u.$$

It is known that there is a choice of signature such that signatures of all rooted subtrees of a given non-colored rooted tree can be computed in linear time [4]. In this paper, we consider a tree-like chemical graph composed of only four types of atoms. Then, by converting a given rooted chemical tree G into a non-colored rooted tree, we can compute signatures of all rooted subtrees of G in linear time. Note that signature $\sigma(v, T_v)$ is independent of the given number of vertices. In the rest of this paper, we write $\sigma(v, T_v)$ as $\sigma(v)$ if T_v is clear from the context.

2.3 Sketch of our algorithm

Before giving the definition of stereoisomers, we show a sketch of our counting algorithm. Here, we consider an example given in Fig. 3. Our counting algorithm computes the number of stereoisomers from bottom to the root along tree G. At vertex v_1, the number of combinations of stereoisomers of children of v_1 such that v_1 is (resp., is not) an asymmetric carbon atom is computed as $h(v_1)$ (resp., $g(v_1)$), and the number of stereoisomers of T_{v_1} is computed as $f(v_1)$. Obviously, we have $g(v_1) = 0$, $h(v_1) = 1$ and $f(v_1) = g(v_1)+2h(v_1) = 2$ because there exist two different configurations around v_1 when v_1 is an asymmetric carbon atom. We represent these two configurations by two labels " + " and " − ."

Similarly, we have $g(v_2) = 0$, $h(v_2) = 1$ and $f(v_2) = g(v_2) + 2h(v_2) = 2$. After that, at vertex v_3, we compute $g(v_3)$, $h(v_3)$ and $f(v_3)$. Let $T^+_{v_1}$ and $T^-_{v_1}$ (resp., $T^+_{v_2}$ and $T^-_{v_2}$) be two possible

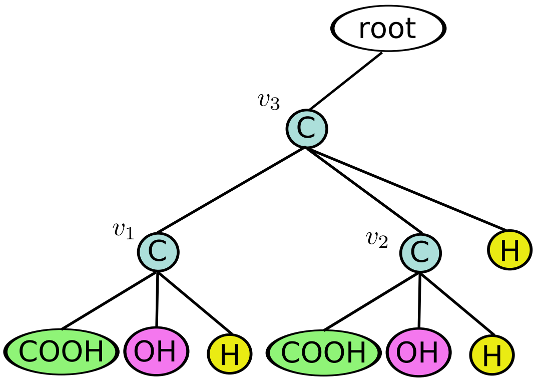


Figure 3: An example of rooted subgraphs for showing a sketch of our algorithm

configurations of T_{v_1} (resp., T_{v_2}), where $T^+_{v_1}$ and $T^+_{v_2}$ (resp., $T^-_{v_1}$ and $T^-_{v_2}$) are stereoisomorphic. Then, $g(v_3)$ corresponds to two combinations $(T^+_{v_1}, T^+_{v_2})$ and $(T^-_{v_1}, T^-_{v_2})$, and $h(v_3)$ corresponds to one combination $(T^+_{v_1}, T^-_{v_2})$. Since T_{v_1} and T_{v_2} are rooted-isomorphic, it is enough to consider one combination $(T^+_{v_1}, T^-_{v_2})$ though we need to consider two configurations $(T^+_{v_1}, T^-_{v_2})$ and $(T^-_{v_1}, T^+_{v_2})$ for this combination. Then we have $g(v_3) = f(v_1) = 2$, $h(v_3) = \binom{f(v_1)}{2} = 1$ and $f(v_3) = g(v_3) + 2h(v_3) = 4$.

In Sections 2.4 and 2.5, we give a formal definition of labels (such as "+" and "−"), isomorphism considering difference of configurations, functions g, h and f, and configurations corresponding to labels.

2.4 Definition of stereoisomer

This subsection gives a formal definition of *stereoisomers* considered in this paper.

2.4.1 Definition of representations and stereoisomorphism

To define stereoisomers of G, we first introduce a label $l(v)$ for each carbon atom $v \in V_{\rm C}$, where $l(v)$ takes one of $+$, $-$, *cis*, *trans* and nil (nil means that v has a unique configuration around v). As will be shown, labels *cis* and *trans* do not always correspond to chemical terms cis and trans. The definition of each label is given in Section 2.4.2. We define the total order among these labels by

$$\text{“}+\text{”} > \text{“}-\text{”} > \text{“}\textit{cis}\text{”} > \text{“}\textit{trans}\text{”} > \text{“nil.”}$$

For every vertex $v \in V_{\rm O} \cup V_{\rm N} \cup V_{\rm H}$, define $l(v) = \text{nil}$.

We next introduce a *representation* I of G as a set of pairs of vertex-number $n(v)$ and label $l(v)$ over all vertices $v \in V$. That is,

$$I = \{(n(v), l(v)) \mid v \in V\}.$$

Let $\mathcal{R}(G)$ denote the set of all representations I of G, where $|\mathcal{R}(G)| = 5^{|V_{\rm C}|}$ holds. Similarly, for each vertex $v \in V$, we define a *representation* I_v of the rooted subtree T_v as

$$I_v = \{(n(u), l(u)) \mid u \in V(T_v)\}.$$

Let $\mathcal{R}(T_v)$ denote the set of all representations I_v of T_v. As will be shown in Section 2.4.2, only representations which satisfy a certain condition, called "proper representations," define stereoisomers.

For each vertex $v \in V$ such that $V(T_v) = \{v, v_1, \ldots, v_p\}$ holds, the *signature* $\sigma_s(I_v)$ of a representation $I_v \in \mathcal{R}(T_v)$ is given as the sequence

$$\sigma_s(I_v) = [(\sigma(v), l(v)), (\sigma(v_1), l(v_1)), \ldots, (\sigma(v_p), l(v_p))],$$

where the order that $v_1, v_2, \ldots, v_p$ appear in the sequence is determined by the next recursive formula.
(i) For a leaf $v \in V$, it holds that $p = |V(T_v) \setminus \{v\}| = 0$. We define $\sigma_s(I_v) = [(\sigma(v), l(v))]$.
(ii) For a representation I_v of the subtree T_v rooted at a non-leaf vertex $v \in V$ with $Ch(v) = \{x_1, x_2, \ldots, x_k\}$, denote $I_v = \{(n(v), l(v))\} \cup I_{x_1} \cup I_{x_2} \cup \cdots \cup I_{x_k}$, $I_{x_i} \in \mathcal{R}(T_{x_i})$ $(i = 1, 2, \ldots .k)$. We assume without loss of generality that $\sigma_s(I_{x_1}), \sigma_s(I_{x_2}), \ldots \sigma_s(I_{x_k})$ are sorted in a lexicographically non-decreasing order and that it holds $\sigma_s(I_{x_i}) = [(\sigma(x_{i1}), l(x_{i1})), (\sigma(x_{i2}), l(x_{i2})), \ldots, (\sigma(x_{in_i}), l(x_{in_i}))]$, $n_i = |V(T_{x_i})|$ $(i = 1, 2, \ldots, k)$. Then we define $[v_1, v_2, \ldots, v_p] = [x_{11}, x_{12}, \ldots, x_{1n_1}, x_{21}, x_{22}, \ldots, x_{2n_2}, \ldots, x_{kn_k}]$.

Note that the $\sigma_s(I_v)$ is independent of the given numbering of vertices.

Definition 2. *For two subtrees T_u and T_v, representations $I_u \in \mathcal{R}(T_u)$ and $I_v \in \mathcal{R}(T_v)$ are* rooted-stereoisomorphic *if and only if $\sigma_s(I_u) = \sigma_s(I_v)$ holds. If I_u and I_v are rooted-stereoisomorphic, we write this as $I_u \underset{I}{\approx} I_v$.*

The *signature* $\sigma_s(I)$ of a representation $I \in \mathcal{R}(G)$ is defined as follows.
(i) If G has the unicentroid v, then we define $\sigma_s(I) = \sigma_s(I_v)$.
(ii) If G has the bicentroid $\{v_1, v_2\}$, where $\sigma_s(I_{v_1}) \geq \sigma_s(I_{v_2})$ and

$$\sigma_s(I_{v_i}) = [(\sigma(v_{i1}), l(v_{i1}), (\sigma(v_{i2}), l(v_{i2})), \ldots, (\sigma(v_{in_i}), l(v_{in_i}))],\ n_i = |V(T_{x_i})|\ (i = 1, 2),$$

then we define

$$\sigma_s(I) = [(\sigma(v_{11}), l(v_{11})), \ldots, (\sigma(v_{1n_1}), l(v_{1n_1})), (\sigma(v_{21}), l(v_{21})), \ldots, (\sigma(v_{2n_2}), l(v_{2n_2}))].$$

Definition 3. *Two representations $I, I' \in \mathcal{R}(G)$ are* stereoisomorphic *if and only if $\sigma_s(I) = \sigma_s(I')$ holds.*

We remark that a representation $I \in \mathcal{R}(G)$ may not correspond to any possible set of configurations around carbon atoms. Section 2.4.2 defines "proper representations" to denote those which give recursive structures of configurations around carbon atoms. Also two distinct representations I and I' may be stereoisomorphic. Section 2.4.3 shows how to uniquely choose one of them as the "canonical form."

2.4.2 Definition of proper representations

This subsection defines "proper representations." In the rest of this section, we regard only the vertex v_1 with $n(v_1) < n(v_2)$ in the bicentroid $\{v_1, v_2\}$ of G as the centroid of G, and treat the edge corresponding to a double bond between two adjacent carbon atoms as two distinct edges. We consider that these two edges and two carbon atoms form a circuit, which we call a *carbon circuit.*

First we introduce an *orientation* of a carbon circuit. We define an *orientation* of a carbon circuit between two adjacent carbon atoms $u, v \in V_C$ only if one of the following cases holds. Otherwise, no orientation is defined for carbon circuits. We suppose that v is closer to the root than u. Orientation of carbon circuit is the new key notion to lead us to a mathematically consistent representation for stereoisomers.

Case-1. u has two children x and y such that $\sigma_s(I_x) > \sigma_s(I_y)$ (see Fig. 4(a)): For the four directions d_0, d_1, d_2 and d_3 of carbon atom u (see Fig. 1(a)), x and y are assumed to be in directions d_2 and d_3, respectively. Then we define the orientation of the carbon circuit between u and v as

$$d_0 \to u \to d_1$$

(see Fig. 5(a)).

Case-2. u and its child $u' \in V_C$ are connected by a double bond (see Fig. 4(b)): For the four directions d_0, d_1, d_2 and d_3 of carbon atom u (see Fig. 1(a)), v is assumed to be in directions d_0

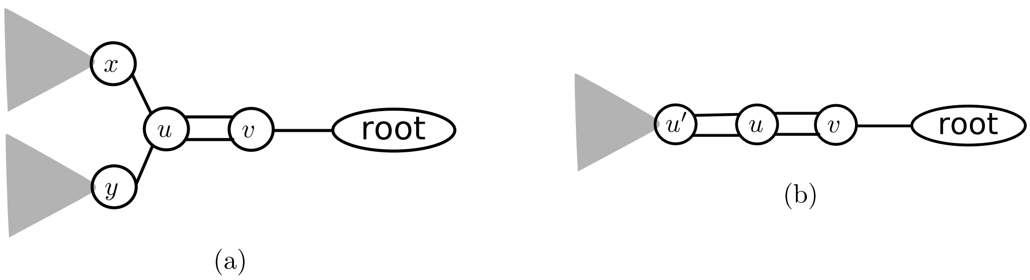


Figure 4: Graph structure around a carbon circuit between u and v.

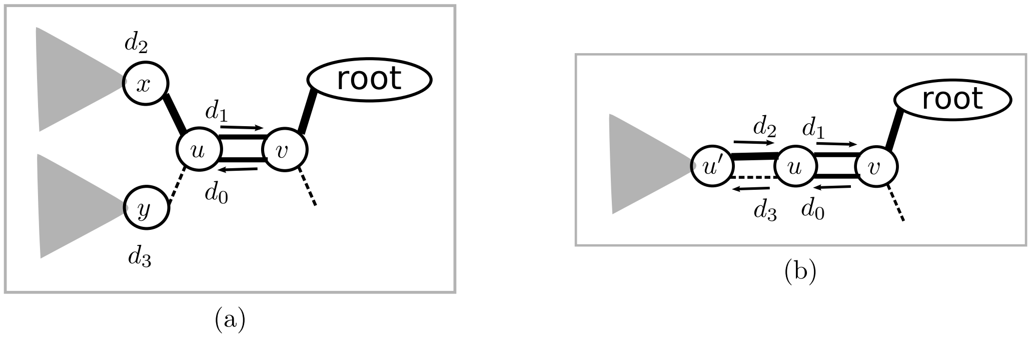


Figure 5: The orientation of a carbon circuit, where d_0, d_1, d_2 and d_3 are the directions from u.

and d_1 and the orientation of the carbon circuit between u and u' is already given as $d_2 \to u \to d_3$. Then we define the orientation of the carbon circuit between u and v is given as

$$d_0 \to u \to d_1$$

(see Fig. 5(b)).

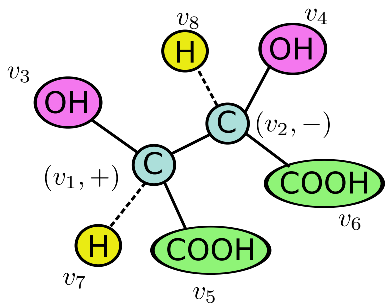
Definition 4. *A representation* $I \in \mathcal{R}(G)$ *(or* $I \in \mathcal{R}(T_v)$*,* $v \in V$*) is called* proper *if the label* $l(v)$ *of each carbon atom* $v \in V_{\rm C}$ *in* I *(or* I_v*) satisfies the following condition.*

Case-1. *v is connected with four atoms: $l(v) \in \{+, -\}$ if $\sigma_s(I_u)$ of every child u of v is different from each other, and $l(v) =$ nil otherwise.*

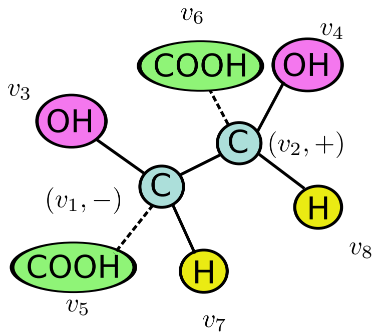
Case-2. *v and one of its children $u \in V_{\rm C}$ are connected by a double bond:*
(i) *the carbon circuit between v and u has no orientation: $l(v) =$ nil.*
(ii) *the carbon circuit between v and u has an orientation, and v is not the centroid of G: $l(v) \in \{cis, trans\}$ if v has other child x than u, and $l(v) =$ nil otherwise.*
(iii) *the carbon circuit between v and u has an orientation, and v is the centroid of G:*
(iii-1) *v and its child $u'(\neq u)$ are connected by a double bond: $l(v) \in \{cis, trans\}$ if the carbon circuit between u and u' has orientation, and $l(v) =$ nil otherwise.*
(iii-2) *v and its children $x, y(\neq u)$ are connected by single bonds: $l(v) \in \{cis, trans\}$ if $\sigma_s(I_x) \neq \sigma_s(I_y)$, and $l(v) =$ nil otherwise.*

Case-3. *The other case: $l(v) =$ nil.*

As will be discussed in Section 2.5, a *proper* representation $I_v \in \mathcal{R}(T_v)$ realizes a set of configurations around carbon atoms in T_v, and is considered as a *rooted-stereoisomer* of T_v. Similarly we consider a *proper* representation $I \in \mathcal{R}(G)$ as a *stereoisomer* of G. However, two proper representations $I_u \in \mathcal{R}(T_u)$ and $I_v \in \mathcal{R}(T_v)$ may be rooted-stereoisomorphic. In the next section, we determine one of all rooted-stereoisomorphic (resp., stereoisomorphic) proper representations as the "canonical form" of the corresponding rooted-stereoisomer (resp., stereoisomer).



(a)



(b)

Figure 6: An example of the compounds that have distinct proper representations which are stereoisomorphic.

2.4.3 Canonical form of proper representations

Here we consider an example given in Fig. 6, where COOH and OH are regarded as single vertices for simplicity and we write the vertex whose vertex-number is i as v_i (i.e., $n(v_i) = i$). For the graph G in Fig. 6, v_1 and v_2 are the bicentroid of G, and there are representations $I_a, I_b \in \mathcal{R}(G)$ with

$$I_a = \{(1, +), (2, -), (3, \text{nil}), (4, \text{nil}), (5, \text{nil}), (6, \text{nil}), (7, \text{nil}), (8, \text{nil})\},$$
$$I_b = \{(1, -), (2, +), (3, \text{nil}), (4, \text{nil}), (5, \text{nil}), (6, \text{nil}), (7, \text{nil}), (8, \text{nil})\}$$

(see Fig. 6(a) and (b), respectively). Fig. 6 shows $T(v_1) \underset{r}{\approx} T(v_2)$, $T(v_3) \underset{r}{\approx} T(v_4)$, $T(v_5) \underset{r}{\approx} T(v_6)$, $T(v_7) \underset{r}{\approx} T(v_8)$ and no two of $T_{v_1}, T_{v_3}, T_{v_5}$ and T_{v_7} are rooted-isomorphic. Then we assume that $s_1 = \sigma(v_1) = \sigma(v_2) > s_2 = \sigma(v_3) = \sigma(v_4) > s_3 = \sigma(v_5) = \sigma(v_6) > s_4 = \sigma(v_7) = \sigma(v_8)$ for integers s_i, $i = 1, 2, 3, 4$. Note that I_a and I_b are distinct as sets. However, I_a and I_b are stereoisomorphic because they have the identical signature

$$\sigma_s(I_a) = \sigma_s(I_b) = [(s_1, +), (s_2, \text{nil}), (s_3, \text{nil}), (s_4, \text{nil}), (s_1, -), (s_2, \text{nil}), (s_3, \text{nil}), (s_4, \text{nil})].$$

Then we define the *canonical form* of proper representations $I \in \mathcal{R}(G)$ as follows.

Definition 5. *Let $L(I)$ be a non-decreasing sequence of the elements $(n(v), l(v))$ in a set I according to the given numbering of the vertices in V.*
(i) *The proper representation $I \in \mathcal{R}(G)$ with the lexicographically maximum $L(I)$ among all proper representations in $\mathcal{R}(G)$ which are stereoisomorphic is defined as the* canonical form *of these representations.*
(ii) *For each vertex $v \in V$, the* canonical form *of representations in $\mathcal{R}(T_v)$ which are rooted-stereoisomorphic is defined by the representation $I_v \in \mathcal{R}(T_v)$ with the lexicographically maximum $L(I_v)$ among them.*

Note that $L(I)$ now reflects the given numbering on the vertex set V (recall that the signature σ_s does not reflect the vertex numbering). For the example in Fig. 6, we have

$$L(I_a) = [(1, +), (2, -), (3, \text{nil}), (4, \text{nil}), (5, \text{nil}), (6, \text{nil}), (7, \text{nil}), (8, \text{nil})].$$
$$L(I_b) = [(1, -), (2, +), (3, \text{nil}), (4, \text{nil}), (5, \text{nil}), (6, \text{nil}), (7, \text{nil}), (8, \text{nil})].$$

and we define I_a to be the canonical form of these stereoisomorphic representations.

Definition 6. *For a tree-like chemical graph $G = (V, E)$, we define the number $f^*(G)$ of stereoisomers of G by the number of all canonical forms of proper representations in $\mathcal{R}(G)$. Similarly, for each vertex $v \in V$, we define the number $f(G, v)$ of stereoisomers of G by the number of all canonical forms of proper representations in $\mathcal{R}(T_v)$.*

Definition 7. *For a tree-like chemical graph $G = (V, E)$, let $\mathcal{I}(G)$ denote a set of proper representations in $\mathcal{R}(G)$ such that $|\mathcal{I}(G)| = f^*(G)$ and no two representations in $\mathcal{I}(G)$ are stereoisomorphic. Similarly, for each vertex $v \in V$, let $\mathcal{I}(v)$ denote a set of proper representations in $\mathcal{R}(T_v)$ such that $|\mathcal{I}(v)| = f(G, v)$ and no two representations in $\mathcal{I}(v)$ are stereoisomorphic.*

In Section 3, we give an algorithm that outputs each element I of $\mathcal{I}(G)$ without duplication. The choice of $\mathcal{I}(G)$ and $\mathcal{I}(v), v \in V$ is determined by an order of choosing backtracking processes in our algorithm (see Section 3.2.1). The algorithm is based on the following relationship between canonical forms of subtrees $T_v, v \in V$.

We call a vertex $v \in V_{\rm C}$ with $l(v) \in \{+, -\}$ an *asymmetric carbon atom.* If $l(v) \in \{cis, trans\}$ then we say that a *cis-trans isomer* arises around v. By definition, a cis-trans isomer cannot arise around an asymmetric carbon atom v.

To compute $f(G, v)$, we define the following.

$g(G, v)$: the number of combinations of stereoisomers of T_x over all children x of v such that (i) v is not an asymmetric carbon atom (i.e., v receives label $l(v) \notin \{+, -\}$ due to the combination); and (ii) no cis-trans isomer arises around any vertex u with $u = v$ or an ancestor u connected to v by a chain of double bonds between carbon atoms (i.e., none of such a vertex u receives label $l(u) \notin \{cis, trans\}$ due to the combination),

$h(G, v)$: the number of combinations of stereoisomers of T_x over all children x of v such that (i) v is an asymmetric carbon atom; or (ii) a cis-trans isomer arises around any vertex u with $u = v$ or an ancestor u connected to v by a chain of double bonds between carbon atoms.

In the rest of this paper, we write $f(G, v), g(G, v)$ and $h(G, v)$ as $f(v), g(v)$ and $h(v)$, respectively.

First we consider the case when v becomes an asymmetric carbon.

Lemma 8. *Let $v \in V_{\rm C}$ be a carbon atom which is not the centroid.*
(i) *v is an asymmetric carbon atom for a combination of stereoisomers of its children if and only if v has exactly three children x, y and w connected with v by single bonds (see Fig. 7(a)) and $I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x$ holds for the rooted-stereoisomers $I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)$ and $I_w \in \mathcal{I}(w)$.*
(ii) *If v has exactly three children x, y and w, then for two sets*

$$\mathcal{I}_h(v) = \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}$$

and

$$\mathcal{I}_g(v) = \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\} \setminus \mathcal{I}_h(v),$$

$\mathcal{I}(v)$ is given by

$$\mathcal{I}(v) = \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), +)\}, I \cup \{(n(v), -)\} \mid I \in \mathcal{I}_h(v)\},$$

and we have $g(v) = |\mathcal{I}_g(v)|$, $h(v) = |\mathcal{I}_h(v)|$ and $f(v) = |\mathcal{I}(v)| = g(v) + 2h(v)$.

Proof of Lemma 8 is given in S1.1.

Next we consider the case when a cis-trans isomer arises.

Lemma 9. *Let $v \in V_{\rm C}$ be a carbon atom which is not the centroid, and let $v' \in V_{\rm C} - \{v\}$ be a descendent of v connected to v by a chain of double bonds between carbon atoms.*
(i) *A cis-trans isomer arises around v for a combination of stereoisomers of children of v' if and only if v has a child adjacent to v by a single bond and v' has exactly two children x and y adjacent to v' by single bonds (see Fig. 7(b)) and $I_x \underset{I}{\not\approx} I_y$ holds for the rooted-stereoisomers $I_x \in \mathcal{I}(x)$ and $I_y \in \mathcal{I}(y)$.*

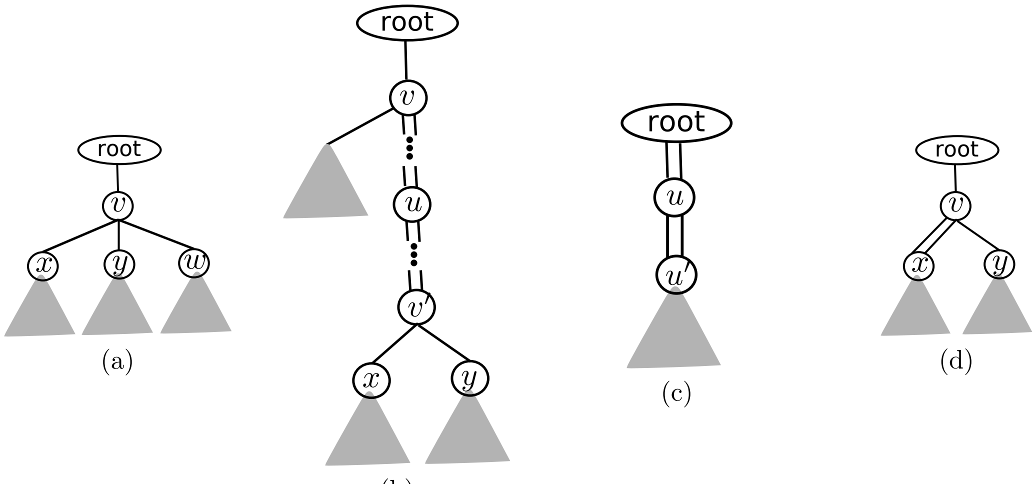


Figure 7: Graph structures around a non-root vertex (in Lemmas 8 and 9).

(ii) *If v' has exactly two children x and y, then for two sets*

$$\mathcal{I}_g(v') = \{I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}$$

and

$$\mathcal{I}_h(v') = \{I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\},$$

$\mathcal{I}(v')$ is given by

$$\mathcal{I}(v') = \{I \cup \{(n(v'), \mathrm{nil})\} \mid I \in \mathcal{I}_g(v') \cup \mathcal{I}_h(v')\},$$

and we have $g(v') = |\mathcal{I}_g(v')|$, $h(v') = |\mathcal{I}_h(v')|$ and $f(v') = |\mathcal{I}(v')| = g(v') + h(v')$.

(iii) *Let u be a carbon atom $u \in V_{\mathrm{C}} - \{v, v'\}$ in the v-v' chain of double bonds between carbon atoms, and u' be the child of u (see Fig. 7(b) and (c)). For two sets*

$$\mathcal{I}_g(u) = I_g(u') \text{ and } \mathcal{I}_h(u) = I_h(u'),$$

$\mathcal{I}(u)$ is given

$$\mathcal{I}(u) = \{I \cup \{(n(v'), \mathrm{nil})\} \mid I \in \mathcal{I}_g(u) \cup \mathcal{I}_h(u)\},$$

and we have $g(u) = |\mathcal{I}_g(u)|$, $h(u) = |\mathcal{I}_h(u)|$ and $f(v) = |\mathcal{I}(u)| = g(u) + h(u)$.

(iv) *If v has a child x adjacent to v by a double bond and a child y adjacent to v by a single bond (see Fig. 7(d)), then for two sets*

$$\mathcal{I}_g(v) = \{I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}(y)\}$$

and

$$\mathcal{I}_h(v) = \{I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}(y)\},$$

$\mathcal{I}(v)$ is given by

$$\mathcal{I}(v) = \{I \cup \{(n(v), \mathrm{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), \mathit{cis})\}, I \cup \{(n(v), \mathit{trans})\} \mid I \in \mathcal{I}_h(v)\},$$

and we have $g(v) = |\mathcal{I}_g(v)|$, $h(v) = |\mathcal{I}_h(v)|$ and $f(v) = |\mathcal{I}(v)| = g(v) + 2h(v)$.

Proof of Lemma 9 is given in S1.2.

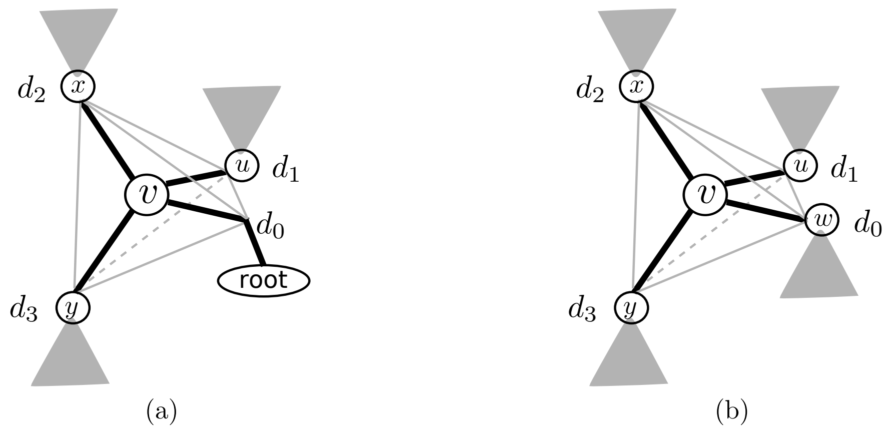


Figure 8: Configurations around a carbon atom $v \in V_C$ which is adjacent to four atoms. (a) The case where v has the parent. It holds $\sigma_s(I_u) > \sigma_s(I_x) > \sigma_s(I_y)$ if and only if $l(v) = +$. It holds $\sigma_s(I_u) > \sigma_s(I_y) > \sigma_s(I_x)$ if and only if $l(v) = -$. (b) The case where v has no parent (i.e., v is the unicentroid). It holds $\sigma_s(I_w) > \sigma_s(I_u) > \sigma_s(I_x) > \sigma_s(I_y)$ if and only if $l(v) = +$. It holds $\sigma_s(I_w) > \sigma_s(I_u) > \sigma_s(I_y) > \sigma_s(I_x)$ if and only if $l(v) = -$.

2.5 Configuration around each carbon atom corresponding to label

This subsection describes how the configuration around each carbon atom v is determined based on its label $l(v)$. By the definition of labels, the configuration around v is unique if a carbon atom v receives label $l(v) = \text{nil}$. In what follows, we consider a carbon atom v with $l(v) \neq \text{nil}$. There are two such cases.

Case-1. v is adjacent to four atoms, and $l(v) \in \{+, -\}$: Such a case occurs only when signature σ_s of every child of v is different each other. If v is one of the bicentroid of G, then we treat the other vertex in the bicentroid as the parent of v.

(i) v has the parent: For the four directions d_i, $i = 0, 1, 2, 3$ from v, as in Fig. 1(a), we assume without loss of generality that the parent of v appears in direction d_0 and the child u of v with the maximum σ_s appears in direction d_1. Then each of the two configurations around v is determined by placing the rest of adjacent vertices x and y in directions d_2 and d_3 so that either

$$\sigma_s(I_x) > \sigma_s(I_y) \Leftrightarrow l(v) = +$$

or

$$\sigma_s(I_x) < \sigma_s(I_y) \Leftrightarrow l(v) = -$$

holds (see Fig. 8(a)).

(ii) v has no parent (i.e. v is the unicentroid of G): For the four directions d_i, $i = 0, 1, 2, 3$ from v, as in Fig. 1(a), we assume without loss of generality that the child w of v with the maximum σ_s appears in direction d_0 and the child u of v with the second maximum σ_s appears in direction d_1. Then each of the two configurations around v is determined by placing the rest of adjacent vertices x and y in directions d_2 and d_3 so that either

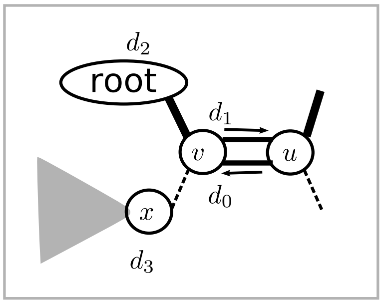
$$\sigma_s(I_x) > \sigma_s(I_y) \Leftrightarrow l(v) = +$$

or

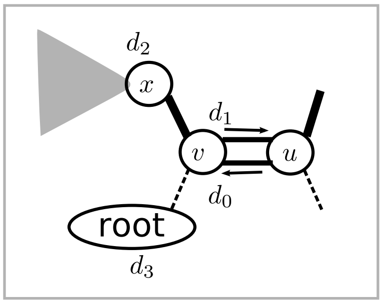
$$\sigma_s(I_x) < \sigma_s(I_y) \Leftrightarrow l(v) = -$$

holds (see Fig. 8(b)).

Case-2. v is adjacent to one of its children $u \in V_C$ by a double bond and $l(v) \in \{\mathit{cis}, \mathit{trans}\}$: Such a case occurs only when the carbon circuit between v and u has an orientation. For the four

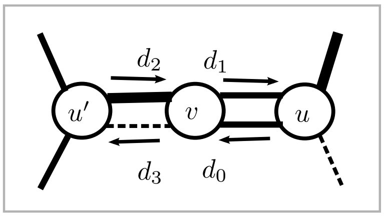


(a) *cis*

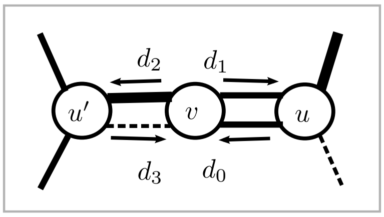


(b) *trans*

Figure 9: Configurations around a carbon circuit between $v \in V_{\mathrm{C}}$ and a child $u \in V_{\mathrm{C}}$ of v, where v is not the centroid of G. (a) $l(v) = cis$, (b) $l(v) = trans$.



(a) *cis*



(b) *trans*

Figure 10: Configurations around carbon circuits between $v \in V_{\mathrm{C}}$ and children $u, u' \in V_{\mathrm{C}}$ of v, where v is the centroid of G. (a) $l(v) = cis$, (b) $l(v) = trans$.

directions d_i, $i = 0, 1, 2, 3$ from v, as in Fig. 1(a), we assume without loss of generality that the orientation of the carbon circuit between v and u is given by $d_0 \to v \to d_1$.

(i) v is not the centroid of G: Since $l(v) \in \{cis, trans\}$, v has exactly two children. Let x be the other child than u. Then each of the two configurations around v is determined by placing x so that either

$$x \text{ appears in direction } d_3 \Leftrightarrow l(v) = cis$$

or

$$x \text{ appears in direction } d_2 \Leftrightarrow l(v) = trans$$

holds (see Fig. 9).

(ii) v is the centroid of G:

(1) v is adjacent to a child $u'(\neq u)$ by a double bond: The carbon circuit between u and u' has an orientation because $l(v) \in \{cis, trans\}$. Then each of the two configurations around v is determined by placing u' so that either

$$l(v) = cis \Leftrightarrow \text{ the orientation of the carbon circuit between } u \text{ and } u' \text{ is } d_2 \to v \to d_3$$

or

$$l(v) = trans \Leftrightarrow \text{ the orientation of the carbon circuit between } u \text{ and } u' \text{ is } d_2 \leftarrow v \leftarrow d_3$$

holds (see Fig. 10).

(2) v is adjacent to its children x, y $(\neq u)$ by single bonds: It holds $\sigma_s(I_x) \neq \sigma_s(I_y)$ because $I(v) \in \{cis, trans\}$. Assume without loss of generality that x and y appear in directions d_2 and d_3, respectively. Then each of the two configurations around v is determined by placing x and y so that either

$$\sigma_s(I_x) > \sigma_s(I_y) \Leftrightarrow l(v) = cis$$

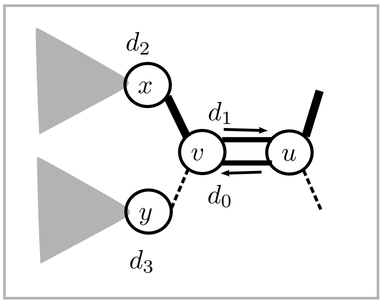


Figure 11: A Configuration around a carbon circuit between $v \in V_{\rm C}$ and a child $u \in V_{\rm C}$ of v, where v is the centroid of G. $l(v) = cis$ if and only if $\sigma_s(I_x) > \sigma_s(I_y)$. $l(v) = trans$ if and only if $\sigma_s(I_x) < \sigma_s(I_y)$.

or

$$\sigma_s(I_x) < \sigma_s(I_y) \Leftrightarrow l(v) = trans$$

holds (see Fig. 11).

Given a proper representation $I \in \mathcal{I}(G)$, we can determine the set of configurations around all carbon atoms represented by I, which are determined from bottom to the root along the rooted tree G. Conversely, given a set of configurations of all carbon atoms of a stereoisomer of G, we can construct the proper representation I corresponding to the structure from bottom to the root along the rooted tree G.

3 Algorithm

In this section we propose an algorithm for enumerating all stereoisomers of a tree-like chemical graph G. The first phase, called *Counting phase*, compute $f^*(G)$ by dynamic programming. Using the information calculated by Counting phase, the second phase, called *Output phase*, constructs each stereoisomer one by one. Sections 3.1 and 3.2 explain Counting phase and Output phase respectively.

3.1 Counting phase

Counting phase computes $f(v), g(v)$ and $h(v)$ for every vertex $v \in V$ from bottom to the root along tree G. When we reach the centroid, we are ready to compute $f^*(G)$. All the recursive formulas for $f(v), g(v), h(v)$ and $f^*(G)$ are given in S2. An entire description of the algorithm is given as follows.

Algorithm Counting phase
Input: A tree-like chemical graph $G = (V, E)$ whose vertex set consists of carbon, hydrogen, oxygen and nitrogen atoms along with vertex-numbers.
Output: The number of stereoisomers $f^*(G)$ and $f(v), g(v), h(v)$ for every vertex $v \in V$ which is not the unicentroid.

Find the centroid of G;
Let the centroid be the root of the tree;
Compute signatures of all rooted subtrees $T_v, v \in V$;
Initialize the scanning queue $Q \leftarrow \phi$;
for each leaf $v \in V$ **do**
  $g(v) := 1$; $h(v) := 0$; $f(v) := 1$;
  Let v be "scanned";
  /* Let u be the parent of v. */

if all the children of u are "scanned" and u is not the unicentroid **then**
ENQUE(Q, u)
end if
end for;
while $Q \neq \phi$ **do**
$v =$ DEQUE(Q);
Compute $f(v), g(v)$ and $h(v)$ as described in Section B.1;
Let v be "scanned";
/* Let u be the parent of v. */
if all the children of u are "scanned" and u is not the unicentroid **then**
ENQUE(Q, u)
end if
end while;
if G has the unicentroid **then**
Compute $f^*(G)$ as described in Section B.2 Case-1.
else /* G has the bicentroid. */
Compute $f^*(G)$ as described in Section B.2 Case-2.
end if.

In general, the number of stereoisomers increases exponentially as the number of atoms increases. In the following, we assume that each of addition, subtraction, multiplication, and division over integers can be executed in a unit time. We get the following theorem.

Theorem 10. *For a tree-like chemical graph $G = (V, E)$ with $|V| = n$, Counting phase computes the number of stereoisomers $f^*(G)$ in $O(n)$ time and space.*

Proof of theorem 10 is given in S1.3.

3.2 Output phase

Output phase constructs proper representations for stereoisomers by using $f^*(G), f(v), g(v)$ and $h(v)$ for all non-unicentroid vertices v. For $i = 1, 2, \ldots, f^*(G)$, we output the proper representation for the i-th stereoisomer of G by backtracking the computation process of Counting phase. When we compute the k-th rooted-stereoisomer of T_v, we detect the corresponding $l(v)$ and calculate k_u for every child u of v, and we trace the computation process recursively to the leaves of G. When this backtrack process completes, we get one proper representation generated by the settled labels $l(v)$ for all $v \in V$.

Here we consider an example given in Fig. 3. When Output phase processes v_3, we have received an instruction from the parent of v_3 "we choose the k_{v_3}-th rooted-stereoisomer of T_{v_3}." It holds $1 \leq k_{v_3} \leq 4$ because Counting phase computed $f(v_3) = 4$. We order rooted-stereoisomers of T_{v_3} as follows.

• If $k_{v_3} = 1$ holds, then we have $l(v_3) =$ nil, and the rooted-stereoisomers of T_{v_3} is composed of the first stereoisomer of T_{v_1} and the first stereoisomer of T_{v_2} (i.e., $k_{v_1} = k_{v_2} = 1$ holds).
• If $k_{v_3} = 2$ holds, then we have $l(v_3) =$ nil, and the rooted-stereoisomers of T_{v_3} is composed of the second stereoisomer of T_{v_1} and the second stereoisomer of T_{v_2} (i.e., $k_{v_1} = k_{v_2} = 2$ holds).
• If $k_{v_3} = 3$ holds, then we have $l(v_3) = +$, and the rooted-stereoisomers of T_{v_3} is composed of the first stereoisomer of T_{v_1} and the second stereoisomer of T_{v_2} (i.e., $k_{v_1} = 1$ and $k_{v_2} = 2$ hold).
• If $k_{v_3} = 4$ holds, then we have $l(v_3) = -$, and the rooted-stereoisomers of T_{v_3} is composed of the first stereoisomer of T_{v_1} and the second stereoisomer of T_{v_2} (i.e., $k_{v_1} = 1$ and $k_{v_2} = 2$ hold).

We compute k_{v_1}, k_{v_2} and $l(v_3)$ from given k_{v_3}, using information of $g(v_3)$, $h(v_3)$ and $f(v_3)$ computed in Counting phase.

The rest of this section is organized as follows. After Section 3.2.1 defines bijections between a set of tuples and combinations of the elements in tuples, Section 3.2.2 gives an entire description of the algorithm and analyzes the time complexity of Output phase.

3.2.1 Bijections for fast generation

Recall that we do not generate any table of (rooted) stereoisomers during Counting phase. However, Output phase needs to find for a given k the k-th combination of numbers k_u of all children of u. To design an $O(1)$ time algorithm for finding a desired combination of such numbers k_u, this subsection defines bijections between a set of tuples and combinations of the elements in tuples.

Definition 11. *For positive integers $M_1, M_2, \ldots, M_p$, define the set $D(M_1, M_2, \ldots, M_p)$ of tuples by*

$$D(M_1, M_2, \ldots, M_p) := \{[k_1, k_2, \ldots, k_p] \mid k_i \in \{1, 2, \ldots, M_i\},\ i = 1, 2, \ldots, p\}.$$

Let $D(; M_1, M_2, \ldots, M_p)$ denote a bijection between the set $\{1, 2, \ldots, M_1 M_2 \cdots M_p\}$ of integers and $D(M_1, M_2, \ldots, M_p)$. Let $D(k; M_1, M_2, \ldots, M_p)$ denote the tuple $[k_1, k_2, \ldots, k_p] \in D(M_1, M_2, \ldots, M_p)$ corresponding to $k \in \{1, 2, \ldots, M_1 M_2 \cdots M_p\}$.

Note that choice of such a bijection $D(; M_1, M_2, \ldots, M_p)$ is not unique. It is not difficult to see that there exists a bijection $D(; M_1, M_2, \ldots, M_p)$ such that we can compute $D(k; M_1, M_2, \ldots, M_p)$ in $O(p)$ time and space for any integer $k \in \{1, 2, \ldots, M_1 M_2 \cdots M_p\}$ (see S3 for the detail).

Definition 12. *For positive integers n and p, define the set $C_{n,p}$ of tuples by*

$$C_{n,p} := \{[k_1, k_2, \ldots, k_p] \mid k_i \in \{1, 2, \ldots, n\},\ i = 1, 2, \ldots, p,\ k_j \neq k_{j'},\ 1 \leq j < j' \leq p\}.$$

Let $C_{n,p}()$ denote a bijection between the set $\{1, 2, \ldots, \binom{n}{p}\}$ of integers and $C_{n,p}$. Let $C_{n,p}(k)$ denote the tuple $[k_1, k_2, \ldots, k_p] \in C_{n,p}$ corresponding to $k \in \{1, 2, \ldots, \binom{n}{p}\}$.

Again choice of such a bijection $C_{n,p}()$ is not unique. For $p \leq 4$, we have shown that there exists a bijection $C_{n,p}()$ such that we can compute $C_{n,p}(k)$ in $O(1)$ time and space for any integer $k \in \{1, 2, \ldots, \binom{n}{p}\}$ (see S3 for the detail).

3.2.2 Description of Output phase

This subsection gives an entire description of Output phase and analyzes its time complexity. Computation precesses for all the cases are given in S4.

Algorithm Output phase
Input: A tree-like chemical graph $G = (V, E)$ whose vertex set consists of carbon, hydrogen, oxygen and nitrogen atoms along with vertex-numbers, the root of G, $f(v), g(v)$ and $h(v)$ for all non-unicentroid vertices v, signatures of all rooted-subtrees $T_v, v \in V$, and $f^*(G)$.
Output: All the elements of $I \in \mathcal{I}(G)$ without duplication.

```
for each k = 1, 2, ..., f*(G) do
  for each v ∈ V do
    l(v) := nil
  end for;
  if G has the unicentroid v then
    /* Let v_1, ..., v_i be the children of v */
    Compute l(v_j) and k_j (j = 1, ..., i) as described in Section D.1;
    for each v_j (j = 1, ..., i) do
      Reverse(v_j, T_{v_j}, k_j)
    end for
  else
    /* Let {v_1, v_2} be the bicentroid of G, where n(v_1) < n(v_2) holds*/
    Compute l(v_1), l(v_2), k_1 and k_2 as described in Section D.1;
    for each v_j (j = 1, 2) do
      Reverse(v_j, T_{v_j}, k_j)
    end for
```

end if;
Output $I = \{(i, l(v_i) \mid i \in \{1, 2, \ldots, n\}\}$ as the k-th stereoisomer
end for.

Procedure Reverse(v,T_v,k)
Input: $v \in V$, a rooted-subtree T_v and positive integer k.
Output: $l(u)$ for all the vertices $u \in T_v$.
if v is not a leaf **then**
/* Let $v_1, \ldots, v_i$ be children of v */
Compute $l(v)$ and k_j $(i = 1, \ldots, i)$ as described in Section D.2;
for each v_j $(j = 1, \ldots, i)$ **do**
Reverse(v_j,T_{v_j},k_j)
end for
else
Return
end if.

Similarly to the time complexity of Counting phase, in the following, we assume that each of addition, subtraction, multiplication, and division over integers can be executed in a unit time. We get the following theorem.

Theorem 13. *For a tree-like chemical graph $G = (V, E)$ with $|V| = n$, Output phase enumerates all the stereoisomers $I \in \mathcal{I}(G)$ without duplication in $O(n)$ space and in $O(n)$ time per isomer.*

Proof of theorem 13 is given in S1.4.

4 Experimental results

We implemented our algorithms and conducted some experiments to evaluate the practical performance. This section shows the experimental results.

In general, the number of vertices in chemical graphs is not so large. Hence computation times of Counting phase are very short. Computation times of Output phase per one stereoisomer are also very short. Then we experimented in order to see that computation times of Output phase increase linearly to the number of stereoisomers. In addition to that, we experimented in order to see that our algorithms run correctly by comparing with the results of Razinger et al.. They constructed the program for exhaustive, nonredundant stereoisomers generation using the idea of Nourse [7], and tested the program with various compounds [10]. We report experimental results performed on a PC with a Dual-Core AMD Opteron(tm) Processor 1212 1.00GHz CPU.

Experiment 1

Using KegDraw obtained from KEGG website (http://www.genome.jp/kegg/download/keg tools.html), we created structural isomers of $C_{25}O_{24}H_{52}$ as instances such that a number of stereoisomers are generated. Fig. 12 shows graph structures of instances. Table 1 and Fig. 13 show the experimental results of our algorithms. From the graph in Fig. 13, we see that the computation time of Output phase increases linearly to the number of stereoisomers.

Experiment 2

We chose some of instances used in [10], which are composed of carbon, hydrogen, oxygen, fluorine, chlorine, and bromine atoms. The current versions of our algorithms can treat only the compounds which are composed of carbon, hydrogen, oxygen, and nitrogen atoms. Then, using KegDraw, we created instances by replacing fluorine, chlorine and bromine atoms in the instances of [10] with substructures $-NH_2$, $-CH_2-NH_2$ and $-CH_2-CH_2-NH_2$, respectively. Graph structures of these instances are shown in Fig. 14. Experimental results are shown in Table. 2. For each compounds, CPU times for Counting phase and Output phase are 0.00, and the number of stereoisomers $f^*(G)$ is the same as that of [10].

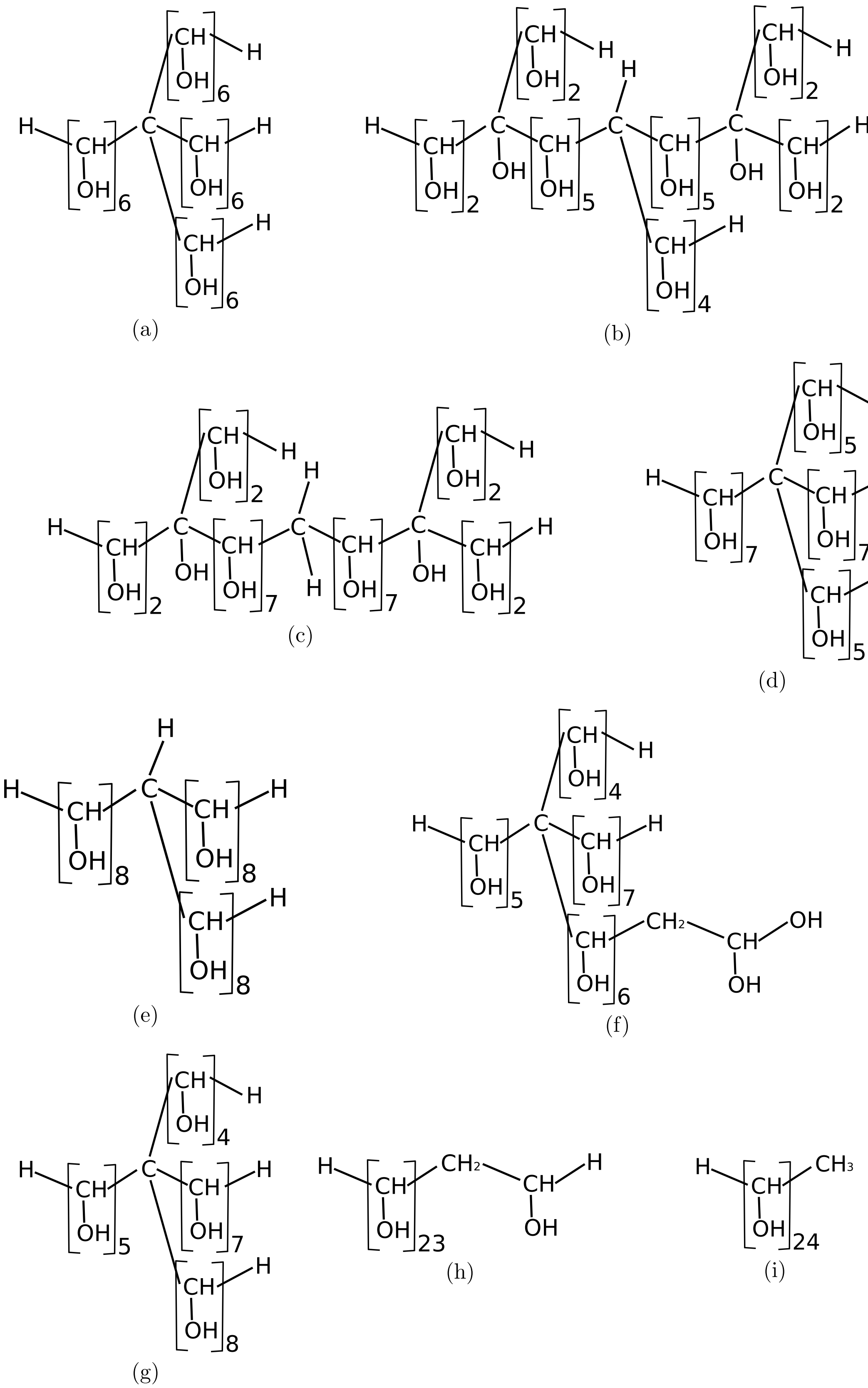


Figure 12: Graph structures of $C_{25}O_{24}H_{52}$

Table 1: Computation time for the chemical graphs shown in Fig. 12.

Input:		time (s)	
$C_{25}O_{24}H_{52}$	$f^*(G)$	Counting phase	Output phase
(a)	88,320	0.00	0.60
(b)	131,072	0.00	0.65
(c)	131,328	0.00	0.69
(d)	524,800	0.00	2.64
(e)	699,136	0.00	3.74
(f)	1,048,576	0.00	4.88
(g)	2,097,152	0.00	9.74
(h)	4,194,304	0.00	19.71
(i)	8,388,608	0.00	39.39

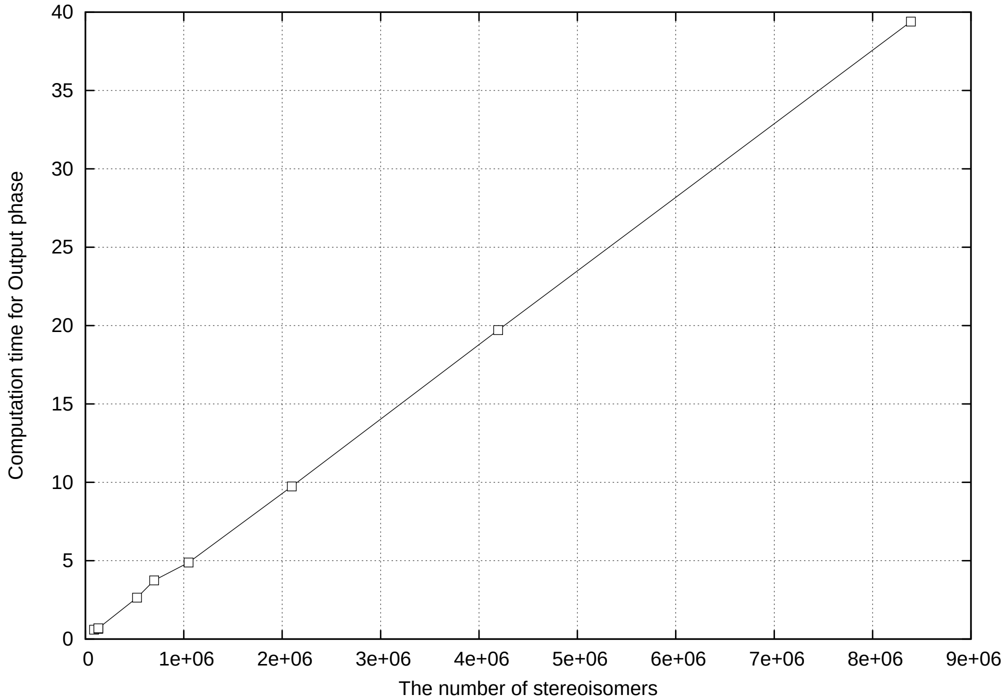


Figure 13: Experimental results on Output phase for $C_{25}O_{24}H_{52}$

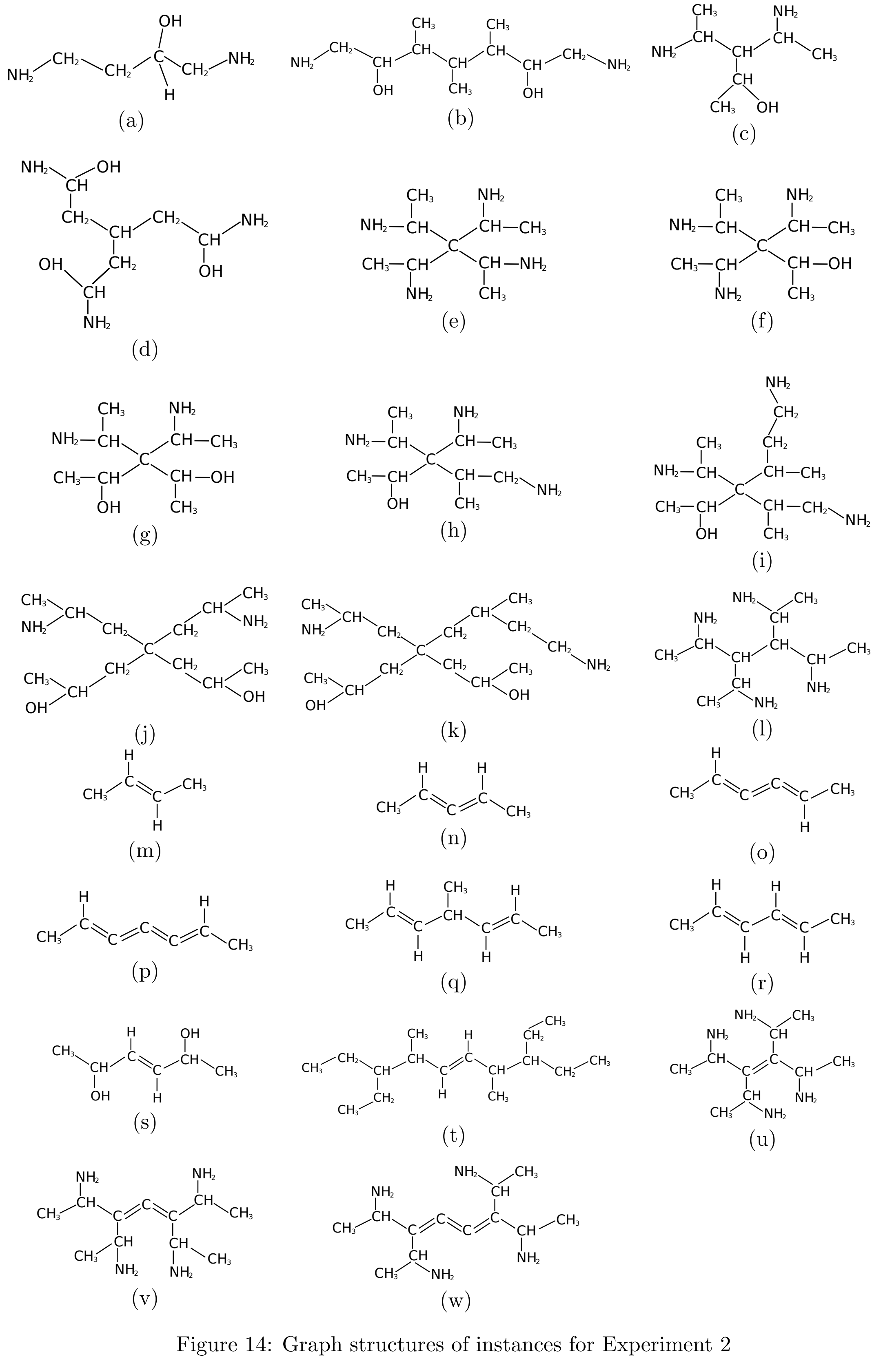


Figure 14: Graph structures of instances for Experiment 2

Table 2: Experimental results for the chemical graphs shown in Fig. 14. $f^*(G)$ is the number of stereoisomers which our algorithms output. N is the number of stereoisomers shown in [10].

Input: Graphs shown in Fig. 14	The number of stereoisomers			The number of stereoisomers	
	$f^*(G)$	N		$f^*(G)$	N
(a)	2	2	(m)	2	2
(b)	16	16	(n)	2	2
(c)	8	8	(o)	2	2
(d)	4	4	(p)	2	2
(e)	5	5	(q)	4	4
(f)	8	8	(r)	3	3
(g)	10	10	(s)	6	6
(h)	16	16	(t)	6	6
(i)	32	32	(u)	7	7
(j)	10	10	(v)	7	7
(k)	32	32	(w)	7	7
(l)	10	10			

5 Conclusion

In this paper, we designed an algorithm for generating stereoisomers of tree-like chemical graphs based on dynamic program. For this, we defined representations of stereoisomers, by attaching a suitable label to each vertex. For a graph with n vertices, our algorithm correctly counts the number of stereoisomers in $O(n)$ time and space and correctly outputs all possible stereoisomers in $O(n)$ space and in $O(n)$ time per stereoisomer. To our knowledge, it is the first algorithm for counting and enumerating stereoisomers with guaranteed computational complexity. Furthermore, the algorithm is optimal provided that each stereoisomer needs to be output explicitly in $O(n)$ time. We also conducted computational experiments to evaluate the practical performance of the algorithm. The results showed that it is very fast also in practice.

We considered in this paper stereoisomers caused only by asymmetry around carbon atoms. However, the proposed techniques might be extended for other types of stereoisomers for which stereochemical configurations depend only on local substructures. Molecules considered in this paper were also limited to those with tree-like structures. Therefore, it is left as future work to extend our algorithms to a wider class of graphs, such as outerplanar graphs, as well as to extend to other types of stereoisomers. Another future work includes visualization and classification of output representations of stereoisomers.

References

[1] Abe, H.; Hayasaka, H.; Miyashita, Y.; Sasaki, S. Generation of Stereoisomeric Structures Using Topological Information Alone. *J. Chem. Inf. Comput. Sci.* **1984**, *24*, 216-219.

[2] Cayley, A. On the Mathematical Theory of Isomers. *Phil. Mag. Ser.* **1874**, *47*, 444-446.

[3] Contreras, M. L.; Alvarez, J.; Guajardo, D.; Rozas, R. Algorithm for Exhaustive and Nonredundant Organic Stereoisomer Generation. *J. Chem. Inf. Model.* **2006**, *46*, 2288-2298.

[4] Dinitz, Y.; Itai, A; Rodeh, M. On an algorithm of Zemlyachenko for subtree isomorphism. *Information Processing Letters* **1999**, *70*, 141-146.

[5] Faulon, J-L.; Visco Jr., D. P.; Roe D. Enumerating Molecules. *Reviews in Computational Chemistry* **2005**, *21*, 209-286.

[6] Jordan, C. Sur Les Assemblages De Lignes, *J. Reine Angew. Math.*, Vol. 70, 185-190, **1869**.

[7] Nourse, J. G. The Configuration Symmetry Group and Its Application to Stereoisomer Generation, Specification, and Enumeration. *J. Am. Chem. Soc.* **1979**, *101*, 1210-1216.

[8] Pólya, G. Kombinatorische Anzahlbestimmungen für Gruppen, Graphen, un chemische Verbindungen. *Acta Math.* **1937**, *68*, 145-253.

[9] Pólya, G.; Read, R. C., *Combinatorial Enumeration of Groups, Graphs, and Chemical Compounds.* Springer-Verlag, New York, **1987**.

[10] Razinger, M.; Balasubramanian, K.; Perdih, M.; Munk, M. E. Stereoisomer Generation in Computer-Enhanced Structure Elucidation. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 812-825.

[11] Rücker, C.; Gugisch, R.; Kerber, A. Manual Construction and Mathematics- and Computer-Aided Counting of Stereoisomers. The Example of Oligoinositols. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1654-1665.

[12] Wieland, T.; Kerber, A.; Laue, R. Principles of the Generation of Constitutional and Configurational Isomers. *J. Chem. Inf. Comput. Sci.* **1996**, *36*, 413-419.

Supplementary information
Efficient Enumeration of Stereoisomers of Tree Structured Molecules Using Dynamic Programming

Tomoki Imada, Shunsuke Ota,
Hiroshi Nagamochi

Graduate School of Informatics,
Kyoto University
{imada,ota,nag}@amp.i.kyoto-u.ac.jp

Tatsuya Akutsu

Bioinformatics Center,
Institute for Chemical Research,
Kyoto University
takutsu@kuicr.kyoto-u.ac.jp

S1 Proof of Lemmas and Theorems

S1.1 Proof of Lemma 8

Proof: (i) From definition of proper representations, $l(v) \in \{+, -\}$ if and only if v is adjacent to four atoms and the signatures $\sigma_s(I_u)$ of all children u of v are different each other. Since v is not the centroid, v has exactly three children x, y and w adjacent to v by single bonds and $I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x$ holds for the rooted-stereoisomers $I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)$ and $I_w \in \mathcal{I}(w)$.

(ii) If a non-root carbon atom v has exactly three children x, y and w, then v is adjacent to x, y, w and its parent by single bonds. Then $h(v)$ (resp., $g(v)$) is the number of combinations of stereoisomers of $T_{x'}$ over all children x' of v such that v is (resp., is not) an asymmetric carbon atom. By (i), v is an asymmetric carbon atom if and only if $I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x$ holds. Then Lemma 8(ii) holds obviously. □

S1.2 Proof of Lemma 9

Proof: (i) From definition of proper representations, for $v \in V_C$ which is not the centroid, $l(v) \in \{cis, trans\}$ if and only if v and one of its children w is connected by a double bond and the other by a single bond, and the carbon circuit between v and w has an orientation. From definition of an orientation of a carbon circuit, the carbon circuit between v and w has an orientation if and only if there exists a descendent $v' \in V_C$ of v connected to v by a chain of double bonds between carbon atoms, v' has exactly two children x and y connected with v' by single bonds, and $I_x \underset{I}{\not\approx} I_y$ holds for rooted-stereoisomers $I_x \in \mathcal{I}(x)$ and $I_y \in \mathcal{I}(y)$.

(ii) From lemmas 8(i) and 9 (i), v' is not an asymmetric carbon atom and a cis-trans isomer does not arise around v'. Then $h(v')$ (resp., $g(v')$) is the number of combinations of stereoisomers of $T_{x'}$ over all children x' of v' such that a cis-trans isomer arises around any ancestor connected to v' by a chain of double bonds between carbon atoms (resp., no cis-trans isomer arises around any ancestor connected to v' by a chain of double bonds between carbon atoms). From Lemma 9(i), a cis-trans isomer arises around any ancestor connected to v' by a chain of double bonds between carbon atoms if and only if $T_x \underset{I}{\not\approx} T_y$ holds. Then Lemma 9(ii) holds obviously.

(iii) From lemmas 8(i) and 9(i), u is not an asymmetric carbon atom and a cis-trans isomer do not arise around u. Then $h(u)$ (resp., $g(u)$) is the number of combinations of stereoisomers of $T_{x'}$ over all children x' of u such that a cis-trans isomer arises around any ancestor connected to u by a chain of double bonds between carbon atoms (resp., no cis-trans isomer arises around any ancestor connected to u by a chain of double bonds between carbon atoms). From Lemma 9(i) and (ii), a cis-trans isomer arises around any ancestor connected to u by a chain of double bonds between carbon atoms if and only if $I_{u'} \in \mathcal{I}_h(u')$ holds. Then Lemma (iii) holds obviously.

(iv) v is adjacent to its parent by a single bond. From Lemma 8(i), v is not an asymmetric carbon atom. Then $h(v)$ (resp., $g(v)$) is the number of combinations of stereoisomers of $T_{x'}$ over all children x' of v such that a cis-trans isomer arises around v (resp., a cis-trans isomer do not arise around v). From Lemma 9(i), (ii) and (iii), a cis-trans isomer arises around v if and only if $I_x \in \mathcal{I}_h(x)$ holds. Then Lemma 9(iv) holds obviously. □

S1.3 Proof of Theorem 10

Proof: We can find the centroid of G in $O(n)$ time and space by Jordan's Theorem [6], and we can compute signatures of all rooted-subtrees $T_v, v \in V$ in $O(n)$ time and space [4]. In Counting phase, every vertex $v \in V$ is visited exactly once and $f(v), g(v)$ and $h(v)$ can be calculated in $O(1)$ time and space as described in Section S2.1, and at the root of G, $f^*(G)$ can be calculated in $O(1)$ time and space as described in Section S2.2. Hence Counting phase runs in $O(n)$ time and space.

Sections S2.1 and S2.2 take all the cases into consideration, and hence Counting phase computes the number of stereoisomers $f^*(G)$ correctly. □

S1.4 Proof of Theorem 13

Proof: For outputting one stereoisomer, every vertex $v \in V$ is visited exactly once and $l(v)$ and k_u for every child u of v can be calculated in $O(1)$ time and space as described in Sections S4.1 and S4.2. Hence Output phase takes $O(n)$ space and $O(n)$ time per stereoisomer.

Sections S4.1 and S4.2 takes all the cases into consideration, and hence Output phase outputs all the stereoisomers $I \in \mathcal{I}(G)$ without duplication. □

S2 Recursive formulas for Counting phase

This section is organized as follows. Section S2.1 shows how to compute $f(v), g(v)$ and $h(v)$. Section S2.2 shows how to compute $f^*(G)$.

S2.1 How to compute $f(v), g(v)$ and $h(v)$

We compute $f(v), g(v)$ and $h(v)$ using Lemmas 8 and 9. We consider the following five cases.
Case-1. $v \in V$ is a leaf: $l(v)$ must be nil and we have

$$g(v) = 1, \quad h(v) = 0, \quad f(v) = 1.$$

Case-2. $v \in V_{\rm C}$ and v has three children. Let x, y and w be three children of v (see Fig. S1 (a)): We consider the following three subcases.
(i) No two of T_x, T_y and T_w are rooted-isomorphic each other: Hence $I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x$ holds. Then

$$\begin{aligned}
\mathcal{I}_g(v) &= \phi, \\
\mathcal{I}_h(v) &= \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\}, \\
\mathcal{I}(v) &= \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), +)\}, I \cup \{(n(v), -)\} \mid I \in \mathcal{I}_h(v)\},
\end{aligned}$$

and we have

$$g(v) = 0, \quad h(v) = f(x)f(y)f(w), \quad f(v) = g(v) + 2h(v).$$

(ii) $T_x \underset{r}{\approx} T_y$ and $T_x \underset{r}{\not\approx} T_w$ hold: Hence $I_x \underset{I}{\not\approx} I_w$ and $I_y \underset{I}{\not\approx} I_w$ hold. Then

$$\begin{aligned}
\mathcal{I}_g(v) &= \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}, \\
\mathcal{I}_h(v) &= \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y\}, \\
\mathcal{I}(v) &= \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), +)\}, I \cup \{(n(v), -)\} \mid I \in \mathcal{I}_h(v)\},
\end{aligned}$$

and we have

$$g(v) = f(x)f(w), \quad h(v) = \binom{f(x)}{2} f(w), \quad f(v) = g(v) + 2h(v).$$

(iii) $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ holds: Then

$$\begin{aligned}
\mathcal{I}_g(v) &= \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}, \\
\mathcal{I}_h(v) &= \{I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}, \\
\mathcal{I}(v) &= \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), +)\}, I \cup \{(n(v), -)\} \mid I \in \mathcal{I}_h(v)\},
\end{aligned}$$

and we have

$$g(v) = f(x)^2, \quad h(v) = \binom{f(x)}{3}, \quad f(v) = g(v) + 2h(v).$$

Case-3. $v \in V_{\mathrm{C}}$, and v is joined to two subtrees by single bonds and is joined to one subtree by a double bond: We consider the following two subcases.
(i) v is joined to its parent by a double bond (see Fig. S1 (b)): Then

$$\begin{aligned}
\mathcal{I}_g(v) &= \{I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}, \\
\mathcal{I}_h(v) &= \{I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}, \\
\mathcal{I}(v) &= \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v) \cup \mathcal{I}_h(v)\},
\end{aligned}$$

and we consider the following two subcases.
(1) If $T_x \underset{r}{\not\approx} T_y$ holds, then

$$g(v) = 0, \quad h(v) = f(x)f(y), \quad f(v) = g(v) + h(v).$$

(2) If $T_x \underset{r}{\approx} T_y$ holds, then

$$g(v) = f(x), \quad h(v) = \binom{f(x)}{2}, \quad f(v) = g(v) + h(v).$$

(ii) v is joined to a child x of v by a double bond (see Fig. S1 (c)): Then

$$\begin{aligned}
\mathcal{I}_g(v) &= \{I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}(y)\}, \\
\mathcal{I}_h(v) &= \{I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}(y)\}, \\
\mathcal{I}(v) &= \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v)\} \cup \{I \cup \{(n(v), \textit{cis})\}, I \cup \{(n(v), \textit{trans})\} \mid I \in \mathcal{I}_h(v)\},
\end{aligned}$$

and we have

$$g(v) = g(x)f(y), \quad h(v) = h(x)f(y), \quad f(v) = g(v) + 2h(v).$$

Case-4. $v \in V_C$ and v is joined to its parent by a double bond and its child y by a double bond (see Fig. S1 (d)): Then

$$\mathcal{I}_g(v) = \mathcal{I}_g(y), \quad \mathcal{I}_h(v) = \mathcal{I}_h(y), \quad \mathcal{I}(v) = \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}_g(v) \cup \mathcal{I}_h(v)\},$$

and we have

$$g(v) = g(y), \quad h(v) = h(y), \quad f(v) = f(y).$$

Case-5. The case other than Cases-1,2,3 and 4: In this case $h(v) = 0$ holds. Then $f(v) = g(v)$, and we consider the following two subcases.
(i) $v \in V$ has exactly one child x: It holds

$$\mathcal{I}(v) = \{I \cup \{(n(v), \text{nil})\} \mid I \in \mathcal{I}(x)\}$$

and we have

$$f(v) = g(v) = f(x).$$

(ii) $v \in V - V_C$ has exactly two children x and y: Then

$$\begin{aligned}\mathcal{I}(v) =& \{I_x \cup I_y \cup \{(n(v), \text{nil})\} \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\} \\ & \cup \{I_x \cup I_y \cup \{(n(v), \text{nil})\} \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}\end{aligned}$$

and we consider the following two subcases.
(1) If $T_x \underset{r}{\not\approx} T_y$ holds, then

$$g(v) = f(x)f(y).$$

(2) If $T_x \underset{r}{\approx} T_y$ holds, then

$$g(v) = f(x) + \binom{f(x)}{2}.$$

S2.2 How to compute $f^*(G)$

We consider the following two subcases.
Case-1. The root of G is the unicentroid $v \in V$: We consider the following three subcases.

(i) $v \in V_C$ holds: We consider the following four subcases.

(1) v has exactly four children x, y, w and z (see Fig. S2 (a)): In this case v can be an asymmetric carbon atom. We consider the following five subcases.
i. If no two of T_x, T_y, T_w and T_z are rooted-isomorphic each other, then

$$\begin{aligned}\mathcal{I}(G) = \{&\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z, \{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z)\}\end{aligned}$$

and we have

$$f^*(G) = 2f(x)f(y)f(w)f(z).$$

ii. If $T_x \underset{r}{\approx} T_y$ holds and no two of T_x, T_y and T_w are rooted-isomorphic each other, then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v), \text{nil}\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y\} \\ & \cup \{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z, \{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y\}\end{aligned}$$

and we have

$$f^*(G) = f(x)f(w)f(z) + 2\binom{f(x)}{2}f(w)f(z).$$

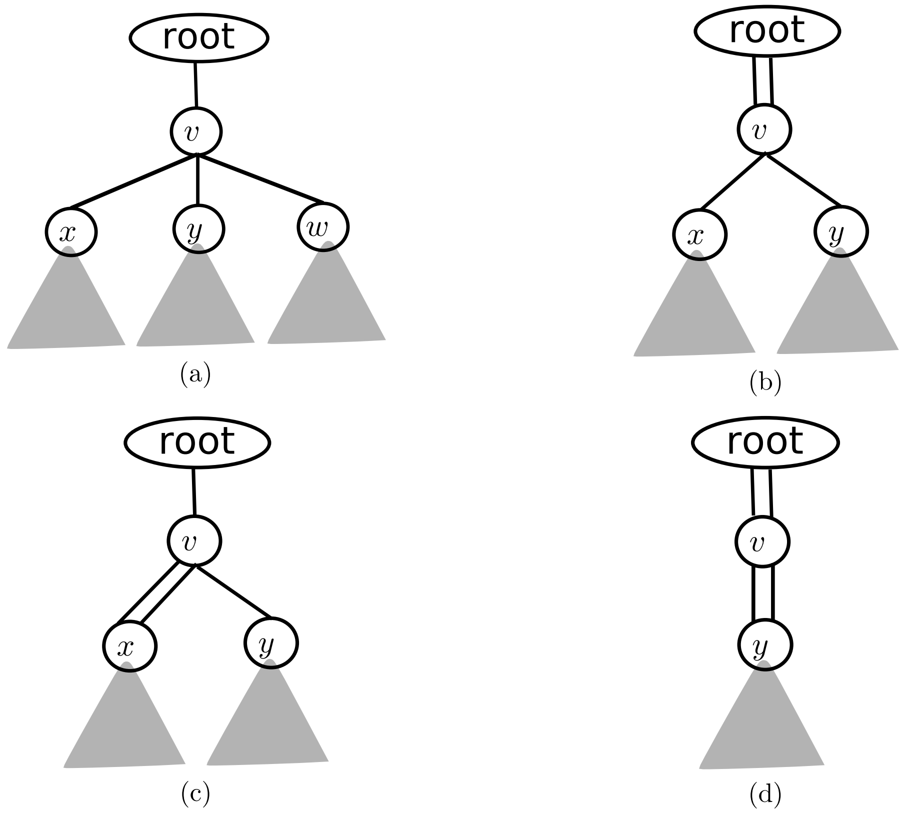


Figure S1: Graph structures around a non-root vertex v.

iii. If $T_x \underset{r}{\approx} T_y, T_w \underset{r}{\approx} T_z$ and $T_x \underset{r}{\not\approx} T_w$ hold, then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y, I_w \underset{I}{\approx} I_z\} \\ & \cup \{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y, I_w \underset{I}{\not\approx} I_z\} \\ & \cup \{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y, I_w \underset{I}{\approx} I_z\} \\ & \cup \{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z, \{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y, I_w \underset{I}{\not\approx} I_z\}\end{aligned}$$

and we have

$$f^*(G) = \left\{ f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w) \right\} + 2\binom{f(x)}{2}\binom{f(w)}{2}.$$

iv. If $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ and $T_x \underset{r}{\not\approx} T_z$ hold, then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y\} \\ & \cup \{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z, \{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}\end{aligned}$$

and we have

$$f^*(G) = f(x)^2 f(z) + 2\binom{f(x)}{3} f(z).$$

v. If $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w \underset{r}{\approx} T_z$ holds, then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\approx} I_w\} \\ & \cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\approx} I_z\} \\ & \cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_z \underset{I}{\not\approx} I_x\} \\ & \cup \{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z, \{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid \\ & \quad I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), \\ & \quad \text{No two of } I_x, I_y, I_w \text{ and } I_z \text{ represent the same stereoisomer}\}\end{aligned}$$

and we have

$$f^*(G) = \left\{ f(x)^2 + \binom{f(x)}{2} + f(x)\binom{f(x)-1}{2} \right\} + 2\binom{f(x)}{4}.$$

(2) v is joined to a child u by a double bond and children x and y by single bonds (see Fig. S2 (b)): In this case a cis-trans isomer can arise around v.

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\} \\ & \cup \{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_g(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\} \\ & \cup \{\{(n(v), \mathit{cis})\} \cup I_u \cup I_x \cup I_y, \{(n(v), \mathit{trans})\} \cup I_u \cup I_x \cup I_y \mid \\ & \quad I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}\end{aligned}$$

and we consider the following two subcases.
i. If $T_x \underset{r}{\not\approx} T_y$ holds, then we have

$$f^*(G) = g(u)f(x)f(y) + 2h(u)f(x)f(y).$$

ii. If $T_x \underset{r}{\approx} T_y$ holds, then we have

$$f^*(G) = \left\{ g(u)f(x) + h(u)f(x) + g(u)\binom{f(x)}{2} \right\} + 2h(u)\binom{f(x)}{2}.$$

(3) v is joined to a child x by a triple bond and children y by a single bond (see Fig. S2 (c)): In this case $I_x \underset{I}{\not\approx} I_y$ holds. Then

$$\mathcal{I}(G) = \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}$$

and we have

$$f^*(G) = f(x)f(y).$$

(4) v is joined to children x and y by double bonds (see Fig. S2 (d)): In this case a cis-trans isomer can arise around v. We consider the following two subcases.

i. $T_x \not\approx_r T_y$ holds: Then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y)\}\\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_h(y)\}\\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_g(y)\}\\ &\cup \{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y, \{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y)\}\end{aligned}$$

and we have

$$f^*(G) = g(x)g(y) + g(x)h(y) + h(x)g(y) + 2h(x)h(y).$$

ii. $T_x \approx_r T_y$ holds: Then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y), I_x \approx_I I_y\}\\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y), I_x \not\approx_I I_y\}\\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_h(y)\}\\ &\cup \{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y, \{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \approx_I I_y\}\\ &\cup \{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y, \{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \not\approx_I I_y\}\end{aligned}$$

and we have

$$f^*(G) = g(x) + \binom{g(x)}{2} + g(x)h(x) + 2\left\{h(x) + \binom{h(x)}{2}\right\}.$$

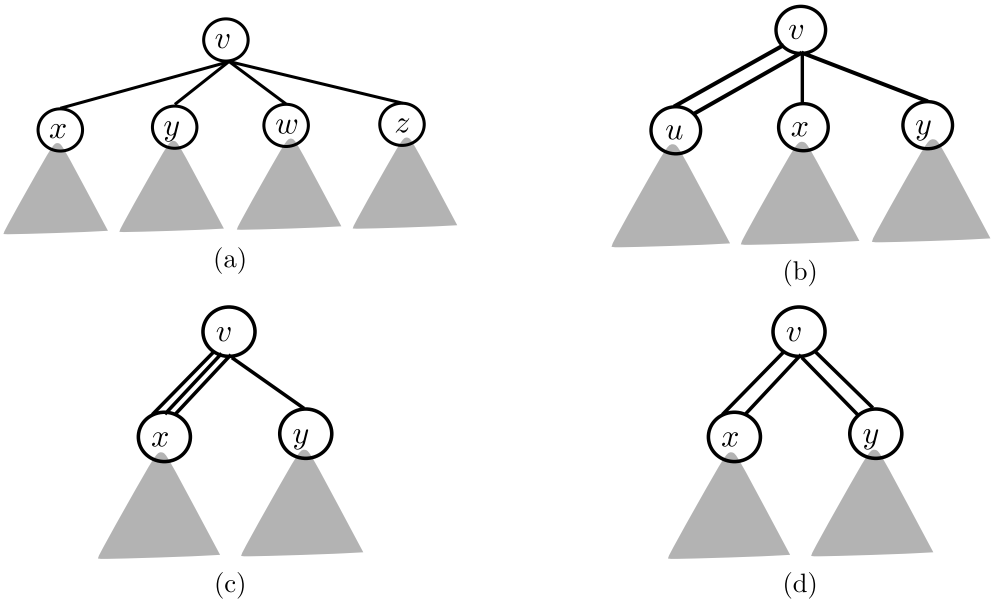


Figure S2: Graph structures around the unicentroid $v \in V_{\mathrm{C}}$.

(ii) $v \in V_{\mathrm{N}}$ holds: We consider the following two subcases.

(1) v has exactly three children x, y and w: We consider the following three subcases.
i. If no two of T_x, T_y and T_w are rooted-isomorphic each other, then

$$\mathcal{I}(G) = \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\}$$

and we have

$$f^*(G) = f(x)f(y)f(w).$$

ii. If $T_x \underset{r}{\approx} T_y$ and $T_x \underset{r}{\not\approx} T_w$ hold, then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\} \\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y\}\end{aligned}$$

and we have

$$f^*(G) = f(x)f(w) + \binom{f(x)}{2} f(w).$$

iii. If $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ holds, then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\} \\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}\end{aligned}$$

and we have

$$f^*(G) = f(x)^2 + \binom{f(x)}{3}.$$

(2) v is joined to a child x by a double bond and a child y by a single bond: In this case $I_x \underset{I}{\not\approx} I_y$ holds. Then

$$\mathcal{I}(G) = \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}$$

and we have

$$f^*(G) = f(x)f(y).$$

(iii) $v \in V_{\text{O}}$ holds: Then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\} \\ &\cup \{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}\end{aligned}$$

and we consider the following two subcases.
(1) If $T_x \underset{r}{\not\approx} T_y$ holds, then we have

$$f^*(G) = f(x)f(y).$$

(2) If $T_x \underset{r}{\approx} T_y$ holds, then we have

$$f^*(G) = f(x) + \binom{f(x)}{2}.$$

Case-2. The root of G is the bicentroid $v_1, v_2 \in V$: We suppose that $n(v_1) < n(v_2)$ holds. In this case, we first compute $f(v_1), g(v_1)$ and $h(v_1)$ regarding v_2 as the single root and $f(v_2), g(v_2)$ and $h(v_2)$ regarding v_1 as the single root. Then we consider the following two subcases.
(i) $v_1, v_2 \in V_{\text{C}}$ holds, and v_1 and v_2 are joined by a double bond (see Fig. S3): A cis-trans isomer occurs around v_1 if and only if $I_{v_1} \in \mathcal{I}_h(v_1)$ and $I_{v_2} \in \mathcal{I}_h(v_2)$. We consider the following two subcases.
(1) If $T_{v_1} \underset{r}{\not\approx} T_{v_2}$ holds, then

$$\begin{aligned}\mathcal{I}(G) =&\{\{(n(v_1), \text{nil})\} \cup \{(n(v_2), \text{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_g(v_1), I_{v_2} \in \mathcal{I}_g(v_2)\} \\ &\cup \{\{(n(v_1), \text{nil})\} \cup \{(n(v_2), \text{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_g(v_1), I_{v_2} \in \mathcal{I}_h(v_2)\} \\ &\cup \{\{(n(v_1), \text{nil})\} \cup \{(n(v_2), \text{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_h(v_1), I_{v_2} \in \mathcal{I}_g(v_2)\} \\ &\cup \{\{(n(v_1), \textit{cis})\} \cup \{(n(v_2), \text{nil})\} \cup I_{v_1} \cup I_{v_2}, \{(n(v_1), \textit{trans})\} \cup \{(n(v_2), \text{nil})\} \cup I_{v_1} \cup I_{v_2} \mid \\ &\quad I_{v_1} \in \mathcal{I}_h(v_1), I_{v_2} \in \mathcal{I}_h(v_2)\}\end{aligned}$$

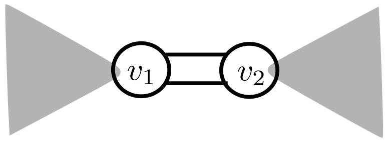


Figure S3: Graph structures around the bicentroid $v_1, v_2 \in V_C$.

and we have

$$f^*(G) = g(v_1)g(v_2) + g(v_1)h(v_2) + h(v_1)g(v_2) + 2h(v_1)h(v_2).$$

(2) If $T_{v_1} \underset{r}{\approx} T_{v_2}$ holds, then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v_1), \mathrm{nil})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_g(v_1), I_{v_2} \in \mathcal{I}_g(v_2), I_{v_1} \underset{I}{\approx} I_{v_2}\} \\ &\cup \{\{(n(v_1), \mathrm{nil})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_g(v_1), I_{v_2} \in \mathcal{I}_g(v_2), I_{v_1} \underset{I}{\not\approx} I_{v_2}\} \\ &\cup \{\{(n(v_1), \mathrm{nil})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}_g(v_1), I_{v_2} \in \mathcal{I}_h(v_2)\} \\ &\cup \{\{(n(v_1), \mathit{cis})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2}, \{(n(v_1), \mathit{trans})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid \\ &\quad I_{v_1} \in \mathcal{I}_h(v_1), I_{v_2} \in \mathcal{I}_h(v_2), I_{v_1} \underset{I}{\approx} I_{v_2}\} \\ &\cup \{\{(n(v_1), \mathit{cis})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2}, \{(n(v_1), \mathit{trans})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid \\ &\quad I_{v_1} \in \mathcal{I}_h(v_1), I_{v_2} \in \mathcal{I}_h(v_2), I_{v_1} \underset{I}{\not\approx} I_{v_2}\}\end{aligned}$$

and we have

$$f^*(G) = g(v_1) + \binom{g(v_1)}{2} + g(v_1)h(v_1) + 2\left\{h(v_1) + \binom{h(v_1)}{2}\right\}.$$

(ii) The case other than case (i): Then

$$\begin{aligned}\mathcal{I}(G) =& \{\{(n(v_1), \mathrm{nil})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}(v_1), I_{v_2} \in \mathcal{I}(v_2), I_{v_1} \underset{I}{\approx} I_{v_2}\} \\ & \{\{(n(v_1), \mathrm{nil})\} \cup \{(n(v_2), \mathrm{nil})\} \cup I_{v_1} \cup I_{v_2} \mid I_{v_1} \in \mathcal{I}(v_1), I_{v_2} \in \mathcal{I}(v_2), I_{v_1} \underset{I}{\not\approx} I_{v_2}\}\end{aligned}$$

and we consider the following two subcases.
(1) If $T_{v_1} \underset{r}{\not\approx} T_{v_2}$ holds, then we have

$$f^*(G) = f(v_1)f(v_2).$$

(2) If $T_{v_1} \underset{r}{\approx} T_{v_2}$ holds, then we have

$$f^*(G) = f(v_1) + \binom{f(v_1)}{2}.$$

S3 Design of bijections

First, we consider how to design bijections in Definition 11.

The case of $p = 1$ is trivial. We set $D(k; M_1) := k$.

When $p = 2$, we number all pairs of two integers as in Table 1. Using the table, we compute $a \geq 0$ and $b \in \{1, 2, \ldots, M_2\}$ such that $k = aM_2 + b$, and set $D(k; M_1, M_2) := [a + 1, b]$.

By extending the case of $p = 2$, we get the following theorem.

Table 1: Lexicographical numbering.

k	1	2	$\cdots$	M_2						$M_1 M_2$
k_1	1	1	$\cdots$	1	2	2	$\cdots$	2	$\cdots$	M_1
k_2	1	2	$\cdots$	M_2	1	2	$\cdots$	M_2	$\cdots$	M_2

Theorem S1. *For any positive integers $p \geq 1$ and M_i $(i = 1, 2, \ldots, p)$, there exists a bijection $D(; M_1, M_2, \ldots, M_p)$ such that we can compute $D(k; M_1, M_2, \ldots, M_p)$ in $O(p)$ time and space for any integer $k \in \{1, 2, \ldots, M_1 M_2 \cdots M_p\}$.*

Proof. We design an algorithm to compute $D(k; M_1, M_2, \ldots, M_p)$ as follows. Clearly, it works in $O(p)$ time and space.

$k' := k$;
for $i = p$ to 1 **do**
  Compute $a \geq 0$ and $b \in \{1, 2, \ldots, M_i\}$ such that $k = aM_i + b$;
  $k_i := b$; $k' := a + 1$
end for;
Set $D_p(k) := [k_1, k_2, \ldots, k_p]$;

□

Next, we consider how to design bijections in Definition 12.

The case of $p = 1$ is trivial. We set $C_{n,1}(k) := k$.

When $p = 2$, we get the following theorem.

Theorem S2. *For any positive integer $n \geq 2$, there exists a bijection $C_{n,2}()$ such that we can compute $C_{n,2}(k)$ in $O(1)$ time and space for any integer $k \in \{1, 2, \ldots, \binom{n}{2}\}$.*

Proof. We design an algorithm to compute $C_{n,2}(k)$ in $O(1)$ time and space.

Case-1. n is odd: Assume that n integers are on a circle ordered clockwise. Let each pair of distinct integers $\{m_1, m_2\}$ with $m_1, m_2 \in \{1, 2, \ldots, n\}$ specify an edge of K_n. By rotating an edge $\{1, i + 1\}$, we can get n pairs of integers whose differences are equal to i.

Let $E(i)$ denote the set of pairs of two integers, where the difference between elements of $e \in E(i)$ equals to i, as follows.

$$E(i) = \{[m_1, m_2] \mid m_1 \in \{1, 2, \ldots, n\}, m_2 = \max\{m_1 + i, m_1 + i - n\}\},\ i = 1, 2, \ldots, (n-1)/2.$$

Then

$$|E(i)| = n,\ i = 1, 2, \ldots, (n-1)/2.$$

We compute $a \geq 0$ and $b \in \{1, 2, \ldots, n\}$ such that $k = an + b$, and set $C_{n,2}(k)$ to be the b-th element of $E(a + 1)$. Thus we set

$$C_{n,2}(k) := \begin{cases} [b, b + a + 1] & \text{if } b + a + 1 \leq n, \\ [b, b + a + 1 - n] & \text{if } b + a + 1 > n. \end{cases}$$

Case-2. n is even: We treat the pairs $[i, n]$ $(i \in \{1, 2, \ldots, n-1\})$ separately from the other pairs $[i, j]$ $(i, j \in \{1, 2, \ldots, n-1\})$. Then we set

$$C_{n,2}(k) := \begin{cases} [k, n] & \text{if } 1 \leq k < n, \\ C_{n-1,2}(k - (n-1)) & \text{if } k \geq n. \end{cases}$$

□

For the case of $p = 3$ and 4, we define the following bijection.

Definition S3. *For positive integers α, m and an integer β such that $\alpha i + \beta \geq 1$ $(1 \leq i \leq m)$, define the set $C'_{\alpha,\beta,m}$ of tuples by*

$$C'_{\alpha,\beta,m} := \{[i,j] \mid i \in \{1,2,\ldots,m\}, j \in \{1,2,\ldots,\alpha i+\beta\}\}.$$

Let $C'_{\alpha,\beta,m}()$ denote a bijection between the set $\{1,2,\ldots,\sum_{i=1}^{m} \alpha i+\beta\}$ of integers and $C'_{\alpha,\beta,m}$. Let $C'_{\alpha,\beta,m}(k)$ denote the tuple $[i,j] \in C'_{\alpha,\beta,m}$ corresponding to $k \in \{1,2,\ldots,\sum_{i=1}^{m} \alpha i+\beta\}$.

Lemma S4. *For any positive integers α, m and any integer β such that $\alpha i + \beta \geq 1$ $(1 \leq i \leq m)$, there exists a bijection $C'_{\alpha,\beta,m}()$ such that we can compute $C'_{\alpha,\beta,m}(k)$ in $O(1)$ time and space for any integer $k \in \{1,2,\ldots,\sum_{i=1}^{m} \alpha i+\beta\}$.*

Proof. We design an algorithm to compute $C'_{\alpha,\beta,m}(k)$ in $O(1)$ time and space.

We define the following two sets.

$$\mathcal{S} := \{[i,j] \mid 1 \leq i \leq m, 1 \leq j \leq \alpha i+\beta\},$$

$$T(i) := \{[i,j] \mid 1 \leq j \leq \alpha i+\beta\} \ (1 \leq i \leq m).$$

Then $\mathcal{S}$ is partitioned into $T(i), 1 \leq i \leq m$. The size of the set $T(i) \cup T(m-i+1), 1 \leq i \leq \lfloor m/2 \rfloor$ is

$$\begin{aligned}|T(i) \cup T(m-i+1)| &= \alpha i+\beta+\alpha(m-i+1)+\beta \\ &= (m+1)\alpha+2\beta.\end{aligned}$$

We define $\gamma := (m+1)\alpha+2\beta$ and compute $a \geq 1$ and $b \in \{1,2,\ldots,\gamma\}$ such that $k = (a-1)\gamma+b$, Let $C'_{\alpha,\beta,m}(k)$ be the b-th element of $T(a) \cup T(m-a+1)$. Then we set

$$C'_{\alpha,\beta,m}(k) := \begin{cases} [a,b] & \text{if } b \leq \alpha a+\beta, \\ [m-a+1, b-(\alpha a+\beta)] & \text{if } b > \alpha a+\beta. \end{cases}$$

□

When $p = 3$, we get the following theorem.

Theorem S5. *For any positive integer $n \geq 3$, there exists a bijection $C_{n,3}()$ such that we can compute $C_{n,3}(k)$ in $O(1)$ time and space for any integer $k \in \{1,2,\ldots,\binom{n}{3}\}$.*

Proof. Let $m = \lfloor n/3 \rfloor$ and $r = n-3m \in \{0,1,2\}$.

Case-1. $r \in \{1,2\}$: Assume that n integers are on a circle of length n, ordered clockwise. Then each triplet of three integers specifies a set of triangles $[a,b,c]$, where $[a,b,c]$ is a triplet of the lengths of clockwise ordered edges of triangles in the set. If we choose a and b, then c is specified as $c = n-a-b$. By rotating one triangle whose vertices are $\{1, a+1, a+b+1\}$, we can get n triplets of three integers, and each triplet corresponds to one triangle in the set of triangles $[a,b,c]$. From assumption we consider the following two patterns.

(i) $a < b < c$ or $a < c < b$,

(ii) $a < b = c$ or $a = c < b$.

To generate the patterns above, we generate pairs $[a,b]$ such that $a < b$ and $c = n-a-b \geq a$. For each $a = 1,2,\ldots,\lfloor n/3 \rfloor$, we set

$$b = a+j \text{ for } j = 1,2,\ldots,n-3a.$$

Now we take all the patterns into consideration. Clearly, we have $\lfloor n/3 \rfloor = m = (n-r)/3$ and

$$\sum_{1 \leq a \leq \lfloor n/3 \rfloor} (n-3a) = (n-r)(n+r-3)/6.$$

Then we have

$$n \sum_{1 \le a \le \lfloor n/3 \rfloor} (n - 3a) = n(n-1)(n-2)/6 = \binom{n}{3}$$

because $r \in \{1, 2\}$.

Let $E(a, b)$ be the set of triplets of three integers generated by rotating a triangle in the set of triangles $[a, b, c = n - a - b]$. Thus

$$E(a,b) = \{[m_1, m_2, m_3] \mid \\ m_1 \in \{1, 2, \ldots, n\}, m_2 = \max\{m_1 + a, m_1 + a - n\}, m_3 = \max\{m_2 + b, m_2 + b - n\}\}$$

and $|E(a, b)| = n$ hold. We compute $k' \ge 1$ and $k'' \in \{1, 2, \ldots, n\}$ such that $k = (k' - 1)n + k''$, and let $C_{n,3}(k)$ be the k''-th element of the k'-th set $E(a, b)$.

To decide k'-th set $E(a, b)$, we compute one element of the set of triplets of three integers

$$\{[a, b = a + j, c = n - a - b] \mid j = 1, 2, \ldots, n - 3a\}, \ a = 1, 2, \ldots, m$$

from given k'.

In order to convert a triplet $[a, b, c]$ into a pair of two integers $[i, j]$, we set $a := m - i + 1$. Then the k'-th triplets of three integers $[a, b = a + j, c = n - a - b]$ is decided by the k'-th pair $[i, j]$ $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, 3i + r - 3)$. From Lemma S4, there exists an $O(1)$ time and space algorithm that computes the k'-th pair $[i, j]$. Then we can decide the k'-th $E(a, b)$ in $O(1)$ time and space.

Case-2. $r = 0$: We treat the triplets $[i, j, n]$ $(i, j \in \{1, 2, \ldots, n - 1\})$ separately from the other triplets $[i, j, k]$ $(i, j, k \in \{1, 2, \ldots, n - 1\})$. Then we set

$$C_{n,3}(k) := \begin{cases} C_{n-1,2}(k) \cup \{n\} & \text{if } 1 \le k \le \binom{n-1}{2}, \\ C_{n-1,3}(k - \binom{n-1}{2}) & \text{if } k > n. \end{cases}$$

□

When $p = 4$, we get the following theorem.

Theorem S6. *For any positive integer $n \ge 4$, there exists a bijection $C_{n,4}()$ such that we can compute $C_{n,4}(k)$ in $O(1)$ time and space for any integer $k \in \{1, 2, \ldots, \binom{n}{4}\}$.*

Proof. Let $m = \lfloor n/4 \rfloor$ and $r = n - 4m \in \{0, 1, 2, 3\}$.

Case-1. $r = 1$: Assume that n integers are on a circle of length n, ordered clockwise. Then each set of four integers specifies a set of tetragons $[a, b, c, d]$, where $[a, b, c, d]$ is a series of the lengths of clockwise ordered edges of tetragons in the set. By rotating one tetragon whose vertices are $\{1, a+1, a+b+1, a+b+c+1\}$, we can get n series of four integers, and each set of four integers corresponds to one tetragon in the set of tetragons $[a, b, c, d]$. We suppose that a is the shortest edge length among a, b, c and d. If there are two or three shortest edges, then we choose the one whose next edge is not the shortest as a. Then we have $a = 1, 2, \ldots, (n-1)/4$, $b > a$, $c \ge a$ and $d \ge a$. We consider patterns of tetragons according to the following two cases.

(i) $c = a$ holds: We define the set $A(a)$ of series of four integers for each $a = 1, 2, \ldots, m$ as follows.

$$A(a) := \{[a, b, c, d] \mid b = n - a - c - d, c = a, d \in \{a, a+1, \ldots, \lfloor (n - 2a)/2 \rfloor\}\}.$$

Then we have

$$|A(a)| = \lfloor (n - 2a)/2 \rfloor - a + 1 = (n - 4a + 1)/2 = 2m + 1 - 2a, \text{ for } a = 1, 2, \ldots, m,$$

and

$$\sum_{1 \le a \le m} |A(a)| = \sum_{1 \le a \le m} (2m + 1 - 2a) = m^2.$$

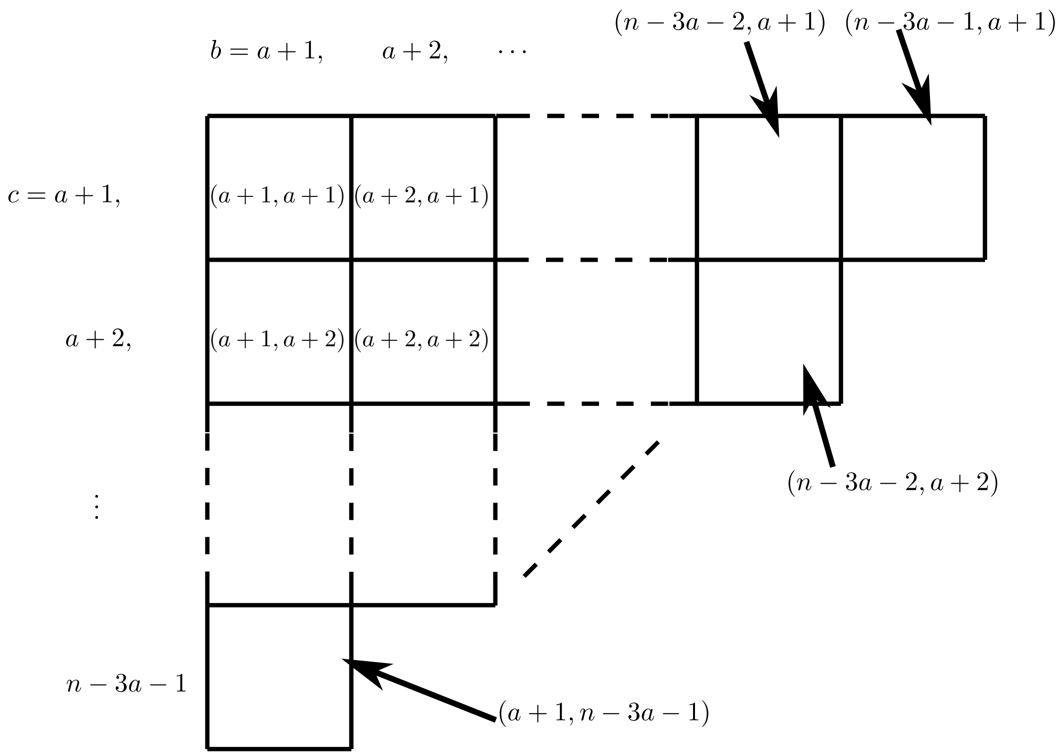


Figure S4: Size of $B(a)$. Each pair in each block denotes (b, c). For $c = a+1$, $b \in \{a+1, a+2, \ldots, a+(n-3a-c)(= n-3a-1)\}$, and for $c = a+2$, $b \in \{a+1, a+2, \ldots, n-3a-2\}$, and $\ldots$, and for $c = a+(n-4a-1)(= n-3a-1)$, $b \in \{a+1\}$.

(ii) $c > a$ holds: We define the set $B(a, c)$ of series of four integers for each $a = 1, 2, \ldots, m-1$ and $c = a+1, a+2, \ldots, a+(n-4a-1)$ as follows.

$$B(a, c) := \{[a, b, c, d] \mid b \in \{a+1, a+2, \ldots, a+(n-3a-c)\}, d = n-a-b-c\}.$$

Then $|B(a, c)| = n-3a-c$ holds. We define the set $B(a) = \cup_{a+1 \leq c \leq n-3a-1} B(a, c)$ for each $a = 1, 2, \ldots, m-1$. Then all the elements of $B(a)$ can be arranged in the upper half of a $(n-4a-1) \times (n-4a-1)$ square, including the diagonal elements (see Fig. S4). Then we have

$$|B(a)| = (n-4a-1)(n-4a)/2 = 2(m-a)(4m+1-4a), \text{ for } a = 1, 2, \ldots m-1.$$

and

$$\sum_{1 \leq a \leq m-1} |B(a)| = \sum_{1 \leq a \leq m-1} 2(m-a)(4m+1-4a) = m(m-1)(8m-1)/3.$$

Now we take all the patterns into consideration. Clearly, $m = (n-1)/4$ holds and we have

$$\begin{aligned} n\left(\sum_{1 \leq a \leq m} |A(a)| + \sum_{1 \leq a \leq m-1} |B(a)|\right) &= n\left\{m^2 + m(m-1)(8m-1)/3\right\} \\ &= nm(8m^2 - 6m + 1)/3 \\ &= n(n-1)(n-2)(n-3)/24 = \binom{n}{4}. \end{aligned}$$

First we compute $k' \geq 1$ and $k'' \in \{1, 2, \ldots, n\}$ such that $k = (k'-1)n + k''$, and let $C_{n,4}(k)$ be the k''-th element generated by rotation of the k'-th tetragon $[a, b, c, d]$. To compute $[a, b, c, d]$ from k', we consider the following two cases.

(i) $k' \leq m^2 = \sum_{1\leq a\leq m} |A(a)|$ holds: From definition, we have $A(a) = \{[a, b = n - a - c - d, c = a, d = a - 1 + j] \mid 1 \leq j \leq 2m + 1 - 2a\},\ 1 \leq a \leq m$. In order to convert a series $[a, b, c, d]$ into a pair of two integers $[i, j]$, we set $a := m - i + 1$. Then $A(a)$ is rewritten as

$$A(a) = \{[a = m + 1 - i, b = n - a - c - d, c = a, d = a - 1 + j] \mid 1 \leq j \leq 2i - 1\}, 1 \leq i \leq m.$$

Then the k'-th series of four integers $[a, b, c, d]$ is decided by the k'-th pair $[i, j]$ $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, 2i - 1)$. From Lemma S4, there exists an $O(1)$ time algorithm that computes the k'-th pair $[i, j]$.

(ii) Otherwise: We set $k' := k' - m^2$. From definition, we have $B(a, c) = \{[a, b = a + l, c = a + h, d = n - a - b - c] \mid 1 \leq l \leq 4m + 1 - 3a - c\},\ 1 \leq h \leq 4m - 4a, 1 \leq a \leq m - 1$. In order to convert a series $[a, b, c, d]$ into a triplet of three integers $[i, j, l]$, we set $i := m - a$ and $j := 4i - h + 1$. Then $B(a, c)$ is rewritten as

$$B(a, c) = \{[a = m-i, b = a+l, c = a+4i-j+1, d = n-a-b-c] \mid 1 \leq l \leq j\}, 1 \leq j \leq 4i, 1 \leq i \leq m-1.$$

Then the k'-th series of four integers $[a, b, c, d]$ is decided by the k'-th triplet $[i, j, l]$ $(i = 1, 2, \ldots, m-1,\ j = 1, 2, \ldots, 4i,\ l = 1, 2, \ldots, j)$. In the following, we consider how to compute the k'-th triplet $[i, j, l]$.

We define the set of triplets of three integers

$$\mathcal{S}(i) := \{[i, j, l] \mid 1 \leq j \leq 4i, 1 \leq l \leq j\}, 1 \leq i \leq m - 1.$$

and then $\mathcal{S}(i)$ is partitioned into the following sets, called blocks.

$$C(i, J, L) := \{[i, j, l] \mid [i, j, l] \in \mathcal{S}(i), j = 4(J - 1) + j', l = 4(L - 1) + l'\ (j', l' \in [1, 4])\}$$
$$(1 \leq J \leq i, 1 \leq L < J),$$

$$D(i, J) := \{[i, j, l] \mid [i, j, l] \in \mathcal{S}(i), j = 4(J - 1) + j', l = 4(J - 1) + l'\ (j', l' \in [1, 4])\}\ (1 \leq J \leq i).$$

For example, $\mathcal{S}(4)$ is partitioned into blocks $C(4, J, L)$ $(1 \leq J \leq 4, 1 \leq L < 3)$ and blocks $D(4, J)$ $(1 \leq J \leq 4)$ (see Fig. S5). For any $i \in \{1, 2, \ldots, m - 1\}$, $J \in \{1, 2, \ldots, i\}$ and $L \in \{1, 2, \ldots, J - 1\}$, we have

$$|C(i, J, L)| = 16, |D(i, J)| = 10.$$

Then we have

$$\sum_{1\leq i\leq m-1} \sum_{1\leq J<i} \sum_{1\leq L<J} |C(i, J, L)| = 16 \sum_{1\leq i\leq m-1} \sum_{1\leq J<i} \sum_{1\leq L<J} 1 = 8m(m - 1)(m - 2)/3.$$

Hence we consider the following two subcases.

(1) $k' \leq 8m(m - 1)(m - 2)/3$ holds: We compute $k_1 \geq 1$ and $k_2 \in \{1, 2, \ldots, 16\}$ such that $k' = 16(k_1 - 1) + k_2$, and let $[i, j, l]$ be the k_2-th element of the k_1-th block $C(i, J, L)$. Block $C(i = I+1, J, L)$ which contains $[i, j, l]$ is decided by $[I, J, L]$ $(1 \leq I \leq m-2, 1 \leq J < I, 1 \leq L < J)$. From Theorem S5, there exists an $O(1)$ time and space algorithm that computes the k_1-th triplet $[I, J, L]$. From Theorem S1, there exists an $O(1)$ time and space algorithm that computes the k_2-th pair $[j', l']$. Then we can compute the k'-th triplet $[i, j, l]$ in $O(1)$ time and space.

(2) Otherwise: We compute $k_1 \geq 1$ and $k_2 \in \{1, 2, \ldots, 10\}$ such that $k' - 8m(m - 1)(m - 2)/3 = 10(k_1 - 1) + k_2$, and let $[i, j, l]$ be the k_2-th element of the k_1-th block $D(i, J)$. Block $D(i, J)$ which contains $[i, j, l]$ is decided by $[i, J]$ $(1 \leq i \leq m - 1, 1 \leq J \leq i)$. From Lemma S4, there exists an $O(1)$ time and space algorithm that computes the k_1-th pair $[i, J]$. From Lemma S4, there exists an $O(1)$ time and space algorithm that computes the k_2-th pair $[j', l']$. Then we can compute the k'-th triplet $[i, j, l]$ in $O(1)$ time and space.

Case-2. $r \in \{0, 2, 3\}$: We treat the series $[i, j, k, n]$ $(i, j, k \in \{1, 2, \ldots, n - 1\})$ separately from the other series $[i, j, k, l]$ $(i, j, k, l \in \{1, 2, \ldots, n - 1\})$. Then we set

$$C_{n,4}(k) := \begin{cases} C_{n-1,3}(k) \cup \{n\} & \text{if } 1 \leq k \leq \binom{n-1}{3}, \\ C_{n-1,4}(k - \binom{n-1}{3}) & \text{if } k > \binom{n-1}{3}. \end{cases}$$

The number of recursive calls $C_{n,p}$ is at most four. □

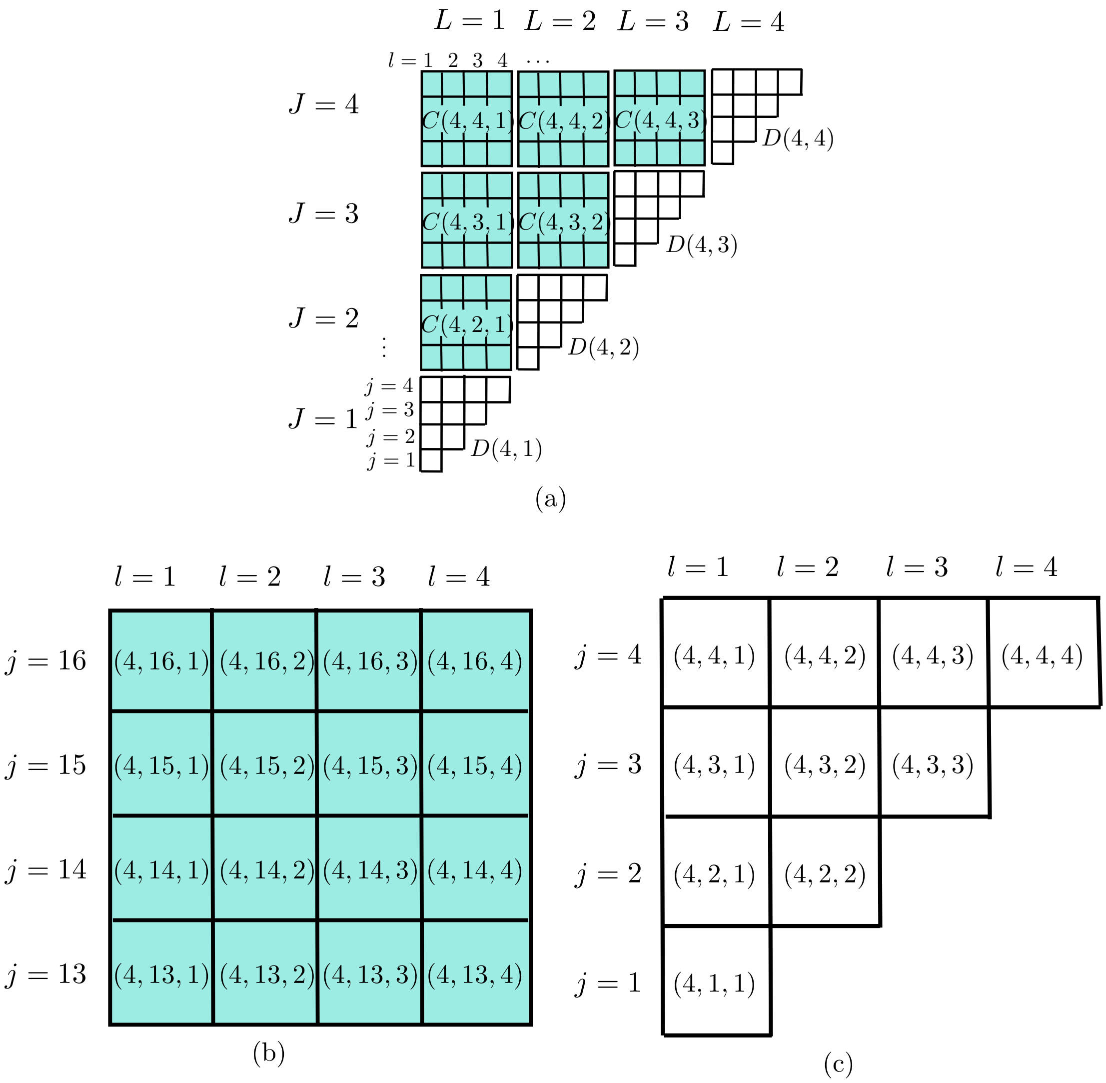


Figure S5: (a) Partitioning of $\mathcal{S}(4)$ into $C(4, J, L)$ $(1 \leq J \leq 4, 1 \leq L < 3)$ and $D(4, J)$ $(1 \leq J \leq 4)$. (b) $C(4, 4, 1)$. Each triplet in each block denotes $[i, j, l]$. (c) $D(4, 1)$. Each triplet in each block denotes $[i, j, l]$.

S4 Computation processes for Output phase

This section is organized as follows. Sections S4.1 and S4.2 show the computation process of Output phase at the root and at a non-root vertex, respectively.

S4.1 Computation process at the root

When Output phase starts for generating the k-th stereoisomer of G, first it initializes $l(v) := \mathrm{nil}$ for all $v \in V$. If the root of G is the unicentroid $v \in V$, then it computes $l(v)$ and k_u for each child u of v from a given k. If the root of G is the bicentroid $\{v_1, v_2\}$, then it computes $l(v_1), l(v_2), k_{v_1}$ and k_{v_2}. We consider the following two subcases.
Case-1. The root of G is the unicentroid $v \in V$: We consider the following three subcases.

(i) $v \in V_{\mathrm{C}}$ holds: We consider the following four subcases.

(1) v has exactly four children x, y, w and z (see Fig. S2 (a)): In this case, v can be an asymmetric carbon atom. We consider the following five subcases.
i. No two of T_x, T_y, T_w and T_z are rooted-isomorphic each other: It holds

$$f^*(G) = 2f(x)f(y)f(w)f(z).$$

We consider the following two subcases.
• $k \leq f(x)f(y)f(w)f(z)$ holds: We choose the k-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z)\},$$

such that I_x is the k_x-th element of $\mathcal{I}(x)$, I_y is the k_y-th element of $\mathcal{I}(y)$, I_w is the k_w-th element of $\mathcal{I}(w)$, and I_z is the k_z-th element of $\mathcal{I}(z)$. Then we set $l(v) := +$ and $[k_x, k_y, k_w, k_z] := D(k; f(x), f(y), f(w), f(z))$.
• Otherwise: We set $\hat{k} = k - f(x)f(y)f(w)f(z)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z)\}.$$

such that I_x is the k_x-th element of $\mathcal{I}(x)$, I_y is the k_y-th element of $\mathcal{I}(y)$, I_w is the k_w-th element of $\mathcal{I}(w)$, and I_z is the k_z-th element of $\mathcal{I}(z)$. Then we set $l(v) := -$ and $[k_x, k_y, k_w, k_z] := D(\hat{k}; f(x), f(y), f(w), f(z))$.

ii. $T_x \underset{r}{\approx} T_y$ holds and no two of T_x, T_y and T_w are rooted-isomorphic each other: It holds

$$f^*(G) = f(x)f(w)f(z) + 2\binom{f(x)}{2}f(w)f(z).$$

We consider the following three subcases.
• $k \leq f(x)f(w)f(z)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k_x, k_w, k_z] = D(k; f(x), f(w), f(z))$ and $k_y := k_x$.
• $f(x)f(w)f(z) < k \leq f(x)f(w)f(z) + \binom{f(x)}{2}f(w)f(z)$ holds: We set $\hat{k} = k - f(x)f(w)f(z)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := +$, $[k', k_w, k_z] := D(\hat{k}; \binom{f(x)}{2}, f(w), f(z))$ and $[k_x, k_y] := C_{f(x),2}(k')$.
• Otherwise: We set $\hat{k} = k - f(x)f(w)f(z) - \binom{f(x)}{2}f(w)f(z)$ and choose $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := -$ and set $[k_x, k_y, k_w, k_z]$ similarly to the case where $f(x)f(w)f(z) < k \le f(x)f(w)f(z) + \binom{f(x)}{2}f(w)f(z)$ holds.

iii. $T_x \underset{r}{\approx} T_y, T_w \underset{r}{\approx} T_z$ and $T_x \underset{r}{\not\approx} T_w$ hold: It holds

$$f^*(G) = \left\{f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w)\right\} + 2\binom{f(x)}{2}\binom{f(w)}{2}.$$

We consider the following five subcases.

• $k \le f(x)f(w)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y, I_w \underset{I}{\approx} I_z\}.$$

Then we set $[k_x, k_w] := D(k; f(x), f(w))$, $k_y := k_x$ and $k_z := k_w$.

• $f(x)f(w) < k \le f(x)f(w) + f(x)\binom{f(w)}{2}$ holds: We set $\hat{k} = k - f(x)f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y, I_w \underset{I}{\not\approx} I_z\}.$$

Then we set $[k_x, k'] := D(\hat{k}; f(x), \binom{f(w)}{2})$, $k_y := k_x$ and $[k_w, k_z] := C_{f(w),2}(k')$.

• $f(x)f(w) + f(x)\binom{f(w)}{2} < k \le f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w)$ holds: We set $\hat{k} = k - f(x)f(w) - f(x)\binom{f(w)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y, I_w \underset{I}{\approx} I_z\}.$$

Then we set $[k', k_w] := D(\hat{k}; \binom{f(x)}{2}, f(w))$, $[k_x, k_y] := C_{f(x),2}(k')$ and $k_z := k_w$.

• $f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w) < k \le f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w) + \binom{f(x)}{2}\binom{f(w)}{2}$ holds: We set $\hat{k} = k - f(x)f(w) - f(x)\binom{f(w)}{2} - \binom{f(x)}{2}f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y, I_w \underset{I}{\not\approx} I_z\}.$$

Then we set $l(v) := +$, $[k', k''] := D(\hat{k}; \binom{f(x)}{2}, \binom{f(w)}{2})$, $[k_x, k_y] := C_{f(x),2}(k')$ and $[k_w, k_z] := C_{f(w),2}(k'')$.

• Otherwise: We set $\hat{k} = k - f(x)f(w) - f(x)\binom{f(w)}{2} - \binom{f(x)}{2}f(w) - \binom{f(x)}{2}\binom{f(w)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y, I_w \underset{I}{\not\approx} I_z\}.$$

Then we set $l(v) = -$ and set $[k_x, k_y, k_w, k_z]$ similarly to the case where $f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w) < k \le f(x)f(w) + f(x)\binom{f(w)}{2} + \binom{f(x)}{2}f(w) + \binom{f(x)}{2}\binom{f(w)}{2}$ holds.

iv. $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ and $T_x \underset{r}{\not\approx} T_z$ hold: It holds

$$f^*(G) = f(x)^2 f(z) + 2\binom{f(x)}{3}f(z).$$

We consider the following three subcases.

• $k \le f(x)^2 f(z)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k_x, k_w, k_z] := D(k; f(x), f(x), f(z))$ and $k_y := k_x$.

• $f(x)^2 f(z) < k \leq f(x)^2 f(z) + \binom{f(x)}{3} f(z)$ holds: We set $\hat{k} = k - f(x)^2$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}.$$

Then we set $l(v) := +$ and $[k', k_z] = D(\hat{k}; \binom{f(x)}{3}, f(z))$ and $[k_x, k_y, k_w] := C_{f(x),3}(k')$.
• Otherwise: We set $\hat{k} = k - f(x)^2$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}.$$

Then we set $l(v) := -$ and set $[k_x, k_y, k_w, k_z]$ similarly to the case where $f(x)^2 f(z) < k \leq f(x)^2 f(z) + \binom{f(x)}{3} f(z)$ holds.

v. $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w \underset{r}{\approx} T_z$ holds: It holds

$$f^*(G) = \left\{ f(x)^2 + \binom{f(x)}{2} + f(x)\binom{f(x)-1}{2} \right\} + 2\binom{f(x)}{4}.$$

We consider the following five subcases.
• $k \leq f(x)^2$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\approx} I_w\}.$$

Then we set $[k_x, k_z] := D(k; f(x), f(z))$, $k_y := k_x$ and $k_w := k_x$.
• $f(x)^2 < k \leq f(x)^2 + \binom{f(x)}{2}$ holds: We set $\hat{k} = k - f(x)^2$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\approx} I_z\}.$$

Then we set $[k_x, k_w] := C_{f(x),2}(\hat{k})$, $k_y := k_x$ and $k_z := k_w$.
• $f(x)^2 + \binom{f(x)}{2} < k \leq f(x)^2 + \binom{f(x)}{2} + f(x)\binom{f(x)-1}{2}$ holds: We set $\hat{k} = k - f(x)^2 - \binom{f(x)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z), I_x \underset{I}{\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_z \underset{I}{\not\approx} I_x\}.$$

Then we compute p and q such that $k = 3p + q$ $(p \geq 0, q \in \{1, 2, 3\})$, and set $[k_1, k_2, k_3] := C_{f(x),2}(p + 1)$ and

$$[k_x, k_y, k_w, k_z] := \begin{cases} [k_1, k_1, k_2, k_3] & \text{if } q = 1, \\ [k_2, k_2, k_3, k_1] & \text{if } q = 2, \\ [k_3, k_3, k_1, k_2] & \text{if } q = 3. \end{cases}$$

• $f(x)^2 + \binom{f(x)}{2} + f(x)\binom{f(x)-1}{2} < k \leq f(x)^2 + \binom{f(x)}{2} + f(x)\binom{f(x)-1}{2} + \binom{f(x)}{4}$ holds: We set $\hat{k} = k - f(x)^2 - \binom{f(x)}{2} - f(x)\binom{f(x)-1}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z),$$
$$\text{No two of } I_x, I_y, I_w \text{ and } I_z \text{ are rooted-stereoisomorphic}\}.$$

Then we set $l(v) := +$ and $[k_x, k_y, k_w, k_z] := C_{f(x),4}(\hat{k})$.
• Otherwise: We set $\hat{k} = k - f(x)^2 - \binom{f(x)}{2} - f(x)\binom{f(x)-1}{2} - \binom{f(x)}{4}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \cup I_z \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_z \in \mathcal{I}(z),$$
$$\text{No two of } I_x, I_y, I_w \text{ and } I_z \text{ are rooted-stereoisomorphic}\}.$$

Then we set $l(v) := -$ and $[k_x, k_y, k_w, k_z] := C_{f(x),4}(\hat{k})$.

(2) v is joined to a child u by a double bond and children x and y by single bonds (see Fig. S2 (b)): We consider the following two subcases.
i. $T_x \underset{r}{\not\approx} T_y$ holds: It holds

$$f^*(G) = g(u)f(x)f(y) + 2h(u)f(x)f(y).$$

We consider the following three subcases.
• $k \leq g(u)f(x)f(y)$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_g(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_u, k_x, k_y] := D(k; g(u), f(x), f(y))$.
• $g(u)f(x)f(y) < k \leq g(u)f(x)f(y) + h(u)f(x)f(y)$ holds: We set $\hat{k} = k - g(u)f(x)f(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \textit{cis})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $l(v) := \textit{cis}$, $[k', k_x, k_y] := D(\hat{k}; h(u), f(x), f(y))$ and $k_u := g(u) + k'$.
• Otherwise: We set $\hat{k} = k - g(u)f(x)f(y) - h(u)f(x)f(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \textit{trans})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $l(v) := \textit{trans}$ and set $[k_u, k_x, k_y]$ similarly to the case where $g(u)f(x)f(y) < k \leq g(u)f(x)f(y) + h(u)f(x)f(y)$ holds.

ii. $T_x \underset{r}{\approx} T_y$ holds: It holds

$$f^*(G) = \left\{g(u)f(x) + h(u)f(x) + g(u)\binom{f(x)}{2}\right\} + 2h(u)\binom{f(x)}{2}.$$

We consider the following five subcases.
• $k \leq f(x)g(u)$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_g(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k_u, k_x] := D(k; g(u), f(x))$ and $k_y := k_x$.
• $f(x)g(u) < k \leq f(x)g(u) + f(x)h(u)$ holds: We set $\hat{k} = k - f(x)g(u)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k', k_x] := D(\hat{k}; h(u), f(x))$, $k_u := g(u) + k'$ and $k_y := k_x$.
• $f(x)g(u) + f(x)h(u) < k \leq f(x)g(u) + f(x)h(u) + g(u)\binom{f(x)}{2}$ holds: We set $\hat{k} = k - f(x)g(u) - f(x)h(u)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \text{nil})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_g(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $[k_u, k'] := D(\hat{k}; g(u), \binom{f(x)}{2})$ and $[k_x, k_y] := C_{f(x),2}(k')$.
• $f(x)g(u) + f(x)h(u) + g(u)\binom{f(x)}{2} < k \leq f(x)g(u) + f(x)h(u) + g(u)\binom{f(x)}{2} + h(u)\binom{f(x)}{2}$ holds: We set $\hat{k} = k - f(x)g(u) - f(x)h(u) - g(u)\binom{f(x)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \textit{cis})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := \textit{cis}$, $[k', k''] := D_2(\hat{k}; h(u), \binom{f(x)}{2})$, $k_u := g(u) + k'$ and $[k_x, k_y] := C_{f(x),2}(k'')$.

• Otherwise: We set $\hat{k} = k - f(x)g(u) - f(x)h(u) - g(u)\binom{f(x)}{2} - h(u)\binom{f(x)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{trans})\} \cup I_u \cup I_x \cup I_y \mid I_u \in \mathcal{I}_h(u), I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := \mathit{trans}$ and set $[k_u, k_x, k_y]$ similarly to the case where $f(x)g(u) + f(x)h(u) + g(u)\binom{f(x)}{2} < k \le f(x)g(u) + f(x)h(u) + g(u)\binom{f(x)}{2} + h(u)\binom{f(x)}{2}$ holds.

(3) v is joined to a child x by a triple bond and children y by a single bond (see Fig. S2 (c)): It holds

$$f^*(G) = f(x)f(y).$$

We output the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; f(x), f(y))$.

(4) v is joined to a child x and y by double bonds (see Fig. S2 (d)): We consider the following two subcases.
i. $T_x \underset{r}{\not\approx} T_y$ holds: It holds

$$f^*(G) = g(x)g(y) + g(x)h(y) + h(x)g(y) + 2h(x)h(y).$$

We consider the following five subcases.
• $k \le g(x)g(y)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y)\}.$$

Then we set $[k_x, k_y] := D(k; g(x), g(y))$.
• $g(x)g(y) < k \le g(x)g(y) + g(x)h(y)$ holds: We set $\hat{k} = k - g(x)g(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_h(y)\}.$$

Then we set $[k_x, k'] := D(\hat{k}; g(x), h(y))$ and $k_y := g(y) + k'$.
• $g(x)g(y) + g(x)h(y) < k \le g(x)g(y) + g(x)h(y) + h(x)g(y)$ holds: We set $\hat{k} = k - g(x)g(y) - g(x)h(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_g(y)\}.$$

Then we set $[k', k_y] := D(\hat{k}; h(x), g(y))$ and $k_x := g(x) + k'$.
• $g(x)g(y) + g(x)h(y) + h(x)g(y) < k \le g(x)g(y) + g(x)h(y) + h(x)g(y) + h(x)h(y)$ holds: We set $\hat{k} = k - g(x)g(y) - g(x)h(y) - h(x)g(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y)\}.$$

Then we set $l(v) := \mathit{cis}$, $[k', k''] := D(\hat{k}; h(x), h(y)), k_x := g(x) + k'$ and $k_y := g(y) + k''$.
• Otherwise: We set $\hat{k} = k - g(x)g(y) - g(x)h(y) - h(x)g(y) - h(x)h(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y)\}.$$

Then we set $l(v) := \mathit{trans}$ and set $[k_x, k_y]$ similarly to the case where $g(x)g(y) + g(x)h(y) + h(x)g(y) < k \le g(x)g(y) + g(x)h(y) + h(x)g(y) + h(x)h(y)$ holds.

ii. $T_x \underset{r}{\approx} T_y$ holds: It holds

$$f^*(G) = g(x) + \binom{g(x)}{2} + g(x)h(x) + 2\left\{h(x) + \binom{h(x)}{2}\right\}.$$

We consider the following seven subcases.
• $k \leq g(x)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $k_x := k$ and $k_y := k$.
• $g(x) < k \leq g(x) + \binom{g(x)}{2}$ holds: We set $\hat{k} = k - g(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_g(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $[k_x, k_y] := C_{g(x),2}(\hat{k})$.
• $g(x) + \binom{g(x)}{2} < k \leq g(x) + \binom{g(x)}{2} + g(x)h(x)$ holds: We set $\hat{k} = k - g(x) - \binom{g(x)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}_h(y)\}.$$

Then we set $[k_x, k'] := D(\hat{k}; g(x), h(x))$ and $k_y := g(x) + k'$.
• $g(x) + \binom{g(x)}{2} + g(x)h(x) < k \leq g(x) + \binom{g(x)}{2} + g(x)h(x) + h(x)$ holds: We set $\hat{k} = k - g(x) - \binom{g(x)}{2} - g(x)h(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $l(v) := \mathit{cis}$, $k_x := g(x) + \hat{k}$ and $k_y := g(x) + \hat{k}$.
• $g(x) + \binom{g(x)}{2} + g(x)h(x) + h(x) < k \leq g(x) + \binom{g(x)}{2} + g(x)h(x) + 2h(x)$ holds: We set $\hat{k} = k - g(x) - \binom{g(x)}{2} - g(x)h(x) - h(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $l(v) := \mathit{trans}$, $k_x := g(x) + \hat{k}$ and $k_y := g(x) + \hat{k}$.
• $g(x) + \binom{g(x)}{2} + g(x)h(x) + 2h(x) < k \leq g(x) + \binom{g(x)}{2} + g(x)h(x) + 2h(x) + \binom{h(x)}{2}$ holds: We set $\hat{k} = k - g(x) - \binom{g(x)}{2} - g(x)h(x) - 2h(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := \mathit{cis}$, $[k', k''] := C_{h(x),2}(\hat{k})$, $k_x := g(x) + k'$ and $k_y := g(x) + k''$.
• Otherwise: We set $\hat{k} = k - g(x) - \binom{g(x)}{2} - g(x)h(x) - 2h(x) - \binom{h(x)}{2}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}_h(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := \mathit{trans}$ and set $[k_x, k_y]$ similarly to the case where $g(x) + \binom{g(x)}{2} + g(x)h(x) + 2h(x) < k \leq g(x) + \binom{g(x)}{2} + g(x)h(x) + 2h(x) + \binom{h(x)}{2}$ holds.

(ii) $v \in V_{\mathrm{N}}$ holds: We consider the following two subcases.

(1) v has exactly three children x, y and w: We consider the following three subcases.

i. No two of T_x, T_y and T_w are rooted-isomorphic each other: It holds

$$f^*(G) = f(x)f(y)f(w).$$

We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\}.$$

Then we set $[k_x, k_y, k_w] := D(k; f(x), f(y), f(w))$.

ii. $T_x \underset{r}{\approx} T_y$ and $T_x \underset{r}{\not\approx} T_w$ hold: It holds

$$f^*(G) = f(x)f(w) + \binom{f(x)}{2} f(w).$$

We consider the following two subcases.
• $k \le f(x)f(w)$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}.$$

Then we set$[k_x, k_w] := D(k; f(x)f(w))$ and $k_y := k_x$.
• Otherwise: We set $\hat{k} = k - f(x)f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set$[k', k_w] := D(\hat{k}; \binom{f(x)}{2}, f(w))$ and $[k_x, k_y] := C_{f(x),2}(k')$.

iii. $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ holds: It holds

$$f^*(G) = f(x)^2 + \binom{f(x)}{3}.$$

We consider the following two subcases.
• $k \le f(x)^2$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}.$$

Then we set$[k_x, k_w] := D(k; f(x), f(w))$ and $k_y := k_x$.
• Otherwise: We set $\hat{k} = k - f(x)f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}.$$

Then we set$[k_x, k_y, k_w] := C_{f(x),3}(\hat{k})$.

(2) v is joined to a child x by a double bond and a child y by a single bond: It holds

$$f^*(G) = f(x)f(y).$$

We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; f(x), f(y))$.

(iii) $v \in V_{\text{O}}$ holds: We consider the following two subcases.

(1) $T_x \underset{r}{\not\approx} T_y$ holds: It holds

$$f^*(G) = f(x)f(y).$$

We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; f(x), f(y))$.

(2) $T_x \underset{r}{\approx} T_y$ holds: It holds

$$f^*(G) = f(x) + \binom{f(x)}{2}.$$

We consider the following two subcases.
• $k \le f(x)$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $k_x := k$ and $k_y := k$.
• Otherwise: We set $\hat{k} = k - f(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $[k_x, k_y] := C_{f(x),2}(\hat{k})$.

Case-2. The root of G is the bicentroid $v_1, v_2 \in V$: We assume without loss of generality that $n(v_1) < n(v_2)$ holds. We consider the following two subcases.

(i) $v_1, v_2 \in V_{\mathrm{C}}$ holds, and v_1 and v_2 are joined by a double bond (see Fig. S3): We set $l(v_1), k_{v_1}$ and k_{v_2} similarly to the Case-1.(i)(4), by interpreting $\{x, y\}$ as $\{v_1, v_2\}$ and $l(v)$ as $l(v_1)$.

(ii) The case other than case (i): We set k_{v_1} and k_{v_2} similarly to the Case-1.(iii), by interpreting $\{x, y\}$ as $\{k_{v_1}, k_{v_2}\}$.

S4.2 Computation process at a non-root vertex v

When Output phase processes a non-root vertex v, it computes $l(v)$ and k_u for each child u of v from a given k.

We consider the following five cases.

Case-1. $v \in V$ is a leaf: It holds

$$f(v) = 1$$

and we set $l(v) := \text{nil}$.

Case-2. $v \in V_{\mathrm{C}}$ and v has three children. Let x, y and w be the three children of v (see Fig. S1 (a)): We consider the following three subcases.

(i) No two of T_x, T_y and T_w are rooted-isomorphic each other: It holds

$$f(v) = 2f(x)f(y)f(w).$$

We consider the following two subcases.
• $k \le f(x)f(y)f(w)$ holds: We choose the k-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\},$$

such that I_x is the k_x-th element of $\mathcal{I}(x)$, I_y is the k_y-th element of $\mathcal{I}(y)$, and I_w is the k_w-th element of $\mathcal{I}(w)$. Then we set $l(v) := +$ and $[k_x, k_y, k_w] := D(k; f(x), f(y), f(w))$.
• $k > f(x)f(y)f(w)$ holds: We set $\hat{k} = k - f(x)f(y)f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w)\},$$

such that I_x is the k_x-th element of $\mathcal{I}(x)$, I_y is the k_y-th element of $\mathcal{I}(y)$, and I_w is the k_w-th element of $\mathcal{I}(w)$. Then we set $l(v) := -$ and $[k_x, k_y, k_w] := D(\hat{k}; f(x), f(y), f(w))$.

(ii) $T_x \underset{r}{\approx} T_y$ and $T_x \underset{r}{\not\approx} T_w$ hold: It holds

$$f(v) = f(x)f(w) + 2\binom{f(x)}{2}f(w).$$

We consider the following three subcases.

• $k \leq f(x)f(w)$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k_x, k_w] := D(k; f(x), f(w))$ and $k_y := k_x$.
• $f(x)f(w) < k \leq f(x)f(w) + \binom{f(x)}{2}f(w)$ holds: We set $\hat{k} = k - f(x)f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := +$ and $[k', k_w] := D(\hat{k}; \binom{f(x)}{2}, f(w)), [k_x, k_y] := C_{f(x),2}(k')$.
• Otherwise: We set $\hat{k} = k - f(x)f(w) - \binom{f(x)}{2}f(w)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $l(v) := -$ and set $[k_x, k_y, k_w]$ similarly to the case where $f(x)f(w) < k \leq f(x)f(w) + \binom{f(x)}{2}f(w)$ holds.

(iii) $T_x \underset{r}{\approx} T_y \underset{r}{\approx} T_w$ holds: It holds

$$f(v) = f(x)^2 + 2\binom{f(x)}{3}.$$

We consider the following three subcases.
• $k \leq f(x)^2$ holds: We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\approx} I_y\}.$$

Then we set $[k_x, k_w] := D(k; f(x), f(w))$ and $k_y := k_x$.
• $f(x)^2 < k \leq f(x)^2 + \binom{f(x)}{3}$ holds: We set $\hat{k} = k - f(x)^2$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), +)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}.$$

Then we set $l(v) := +$ and $[k_x, k_y, k_w] := C_{f(x),3}(\hat{k})$.
• Otherwise: We set $\hat{k} = k - f(x)^2 - \binom{f(x)}{3}$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), -)\} \cup I_x \cup I_y \cup I_w \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_w \in \mathcal{I}(w), I_x \underset{I}{\not\approx} I_y \underset{I}{\not\approx} I_w \underset{I}{\not\approx} I_x\}.$$

Then we set $l(v) := -$ and $[k_x, k_y, k_w] := C_{f(x),3}(\hat{k})$.

Case-3. $v \in V_{\mathrm{C}}$ and v is joined to two subtrees by single bonds and is joined to one subtree by a double bond: We consider the following two subcases.

(i) v is joined to its parent by a double bond (see Fig. S1 (b)): We consider the following two subcases.

(1) If $T_x \underset{r}{\not\approx} T_y$ holds, then

$$f(v) = f(x)f(y)$$

holds. We choose the k-th element of

$$\{\{(n(v), \text{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; f(x), f(y))$.

(2) If $T_x \underset{r}{\approx} T_y$ holds, then

$$f(v) = f(x) + \binom{f(x)}{2}$$

holds. We consider the following two subcases.
• $k \le f(x)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $k_x := k$ and $k_y := k$.
• Otherwise: We set $\hat{k} = k - f(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $[k_x, k_y] := C_{f(x),2}(\hat{k})$.

(ii) v is joined to a child x of v by a double bond (see Fig. S1 (c)):

$$f(v) = g(x)f(y) + 2h(x)f(y)$$

holds and we consider the following three subcases.
• $k \le g(x)f(y)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_g(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; g(x), f(y))$.
• $g(x)f(y) < k \le g(x)f(y) + h(x)f(y)$ holds: We set $\hat{k} = k - g(x)f(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{cis})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $l(v) := \mathit{cis}$ and $[k_x, k_y] := D(\hat{k}; h(x), f(y))$.
• Otherwise: We set $\hat{k} = k - g(x)f(y) - h(x)f(y)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathit{trans})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}_h(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $l(v) := \mathit{trans}$ and $[k_x, k_y] := D(\hat{k}; h(x), f(y))$.

Case-4. $v \in V_{\mathrm{C}}$ and v is joined to its parent by a double bond and its child y by a double bond (see Fig. S1 (d)): It holds

$$f(v) = f(y).$$

We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_y \mid I \in \mathcal{I}(y)\}.$$

Then and we set $k_y := k$.

Case-5. The case other than Cases-1,2,3 and 4: We consider the following two subcases.

(i) $v \in V$ has exactly one child x: It holds

$$f(v) = f(x).$$

We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \mid I \in \mathcal{I}(x)\}.$$

Then we set $k_x := k$.

(ii) $v \in V - V_{\mathrm{C}}$ has exactly two children x and y: We consider the following two subcases.

(1) $T_x \underset{r}{\not\approx} T_y$ holds; It holds

$$f(v) = f(x)f(y).$$

We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y)\}.$$

Then we set $[k_x, k_y] := D(k; f(x), f(y))$.

(2) $T_x \underset{r}{\approx} T_y$ holds; It holds

$$f(v) = f(x) + \binom{f(x)}{2}.$$

We consider the following two subcases.

• $k \leq f(x)$ holds: We choose the k-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\approx} I_y\}.$$

Then we set $k_x := k$ and $k_y := k$.

• otherwise: We set $\hat{k} = k - f(x)$ and choose the $\hat{k}$-th element of

$$\{\{(n(v), \mathrm{nil})\} \cup I_x \cup I_y \mid I_x \in \mathcal{I}(x), I_y \in \mathcal{I}(y), I_x \underset{I}{\not\approx} I_y\}.$$

Then we set $[k_x, k_y] := C_{f(x),2}(\hat{k})$.