A New Integer Linear Programming Formulation to the Inverse QSAR/QSPR for Acyclic Chemical Compounds Using Skeleton Trees*

Fan Zhang[1], Jianshen Zhu[1], Rachaya Chiewvanichakorn[1], Aleksandar Shurbevski[1], Hiroshi Nagamochi[1], Tatsuya Akutsu[2]

[1] Department of Applied Mathematics and Physics, Kyoto University, Kyoto, Japan
{fanzhang, zhujs, ch.rachaya, shurbevski, nag}@amp.i.kyoto-u.ac.jp
[2] Bioinformatics Center, Institute for Chemical Research, Kyoto University, Japan
takutsu@kuicr.kyoto-u.ac.jp

Abstract. Computer-aided drug design is one of important application areas of intelligent systems. Recently a novel method has been proposed for inverse QSAR/QSPR using both artificial neural networks (ANN) and mixed integer linear programming (MILP), where inverse QSAR/QSPR is a major approach for drug design. This method consists of two phases: In the first phase, a feature function f is defined so that each chemical compound G is converted into a vector $f(G)$ of several descriptors of G, and a prediction function ψ is constructed with an ANN so that $\psi(f(G))$ takes a value nearly equal to a given chemical property π for many chemical compounds G in a data set. In the second phase, given a target value y^* of the chemical property π, a chemical structure G^* is inferred in the following way. An MILP $\mathcal{M}$ is formulated so that $\mathcal{M}$ admits a feasible solution (x^*, y^*) if and only if there exist vectors x^*, y^* and a chemical compound G^* such that $\psi(x^*) = y^*$ and $f(G^*) = x^*$. The method has been implemented for inferring acyclic chemical compounds. In this paper, we propose a new MILP for inferring acyclic chemical compounds by introducing a novel concept, skeleton tree, and conducted computational experiments. The results suggest that the proposed method outperforms the existing method when the diameter of graphs is up to around 6 to 8. For an instance for inferring acyclic chemical compounds with 38 non-hydrogen atoms from C, O and S and diameter 6, our method was 5×10^4 times faster.

1 Introduction

Recently, artificial intelligence techniques have been applied to various areas including pharmaceutical and medical sciences. Drug design is one of major tentative topics in such applications. Indeed, many computational methods have been developed for computer-aided drug design. In particular, extensive studies

* The final authenticated version is available online at https://doi.org/10.1007/978-3-030-55789-8_38

have been done on inverse QSAR/QSPR (quantitative structure-activity and structure-property relationships) [12,18]. In this approach, desired chemical activities and/or properties are specified along with some constraints and then chemical compounds satisfying these conditions are inferred, where chemical compounds are usually represented as undirected graphs. Inverse QSAR/QSPR is often formulated as an optimization problem to find a chemical graph maximizing (or minimizing) an objective function under various constraints. In this formalization, objective functions reflect certain chemical activities or properties, and are often derived from a set of training data consisting of known molecules and their activities/properties using statistical machine learning methods.

Since it is difficult to directly handle chemical graphs in traditional statistical learning methods, chemical compounds are often represented as a vector of real or integer numbers, which is called a set of descriptors or a set of features. Using these descriptors, various heuristic and statistical methods have been developed for finding optimal or nearly optimal graph structures under given objective functions [8,12,16]. In such an approach, inference or enumeration of graph structures from a given feature vector is often required as a subtask. In order to solve this subtask, various methods have been developed [5,9,11,14] and some studies have been done on its computational complexity [1,13].

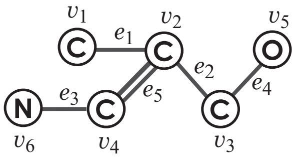
According to the recent rapid progress of Artificial Neural Network (ANN) and deep learning technologies, novel approaches have been proposed for inverse QSAR/QSPR, which include applications of variational autoencoders [6], recurrent neural networks [17,19], and grammar variational autoencoders [10]. In these approaches, neural networks are trained using known compound/activity data and then novel chemical graphs are generated by solving a kind of inverse problems on neural networks. Although these inverse problems are usually solved using various statistical methods, the optimality of the solution is not necessarily guaranteed. In order to guarantee the optimality mathematically, novel mixed integer linear programming (MILP)-based methods have been proposed [2] for ANNs with ReLU functions and sigmoid functions, in which activation functions on neurons are efficiently encoded as piece-wise linear functions so as to represent ReLU functions exactly and sigmoid functions approximately.

Chiewvanichakorn et al. [4] and Azam et al. [3] recently combined the MILP-based formulation of the inverse problem on ANNs [2] with efficient enumeration of tree-like graphs [5]. The combined framework for inverse QSAR/QSPR mainly consists of two phases. The first phase solves (I) Prediction Problem, where each chemical compound G is transformed into a feature vector $f(G)$ and an ANN $\mathcal{N}$ is trained from existing chemical compounds and their values $a(G)$ on a chemical property π to obtain a prediction function $\psi_{\mathcal{N}}$ so that $a(G)$ is predicted as $\psi_{\mathcal{N}}(f(G))$. The second phase solves (II) Inverse Problem, where (II-a) given a target value y^* of the chemical property π, a feature vector x^* is inferred from the trained ANN $\mathcal{N}$ so that $\psi_{\mathcal{N}}(x^*)$ is close to y^* and (II-b) then a set of chemical structures G^* such that $f(G^*) = x^*$ is enumerated. In (II-a) of the above-mentioned methods [3,4], an MILP is formulated. In particular, Azam et al. [3] formulated an MILP for acyclic chemical compounds so that the following is guaranteed: either (i) every feature vector x^* inferred from a trained ANN $\mathcal{N}$ in (II-a) admits a corresponding chemical structure G^* or (ii) no chemical structure exists for a given target value when no feature vector is inferred from the ANN $\mathcal{N}$. In this paper, we propose a new MILP for inferring acyclic chemical compounds with bounded degree, and conducted computational experiments on several chemical properties. The results suggest that the proposed method runs much faster than the previous method [3] does when the number of chemical elements and diameter are relatively small.

2 Preliminary

Let $\mathbb{R}$ and $\mathbb{Z}$ denote the sets of reals and non-negative integers, respectively. For two integers a and b, let $[a, b]$ denote the set of integers i with $a \leq i \leq b$.

Graphs Let $H = (V, E)$ be a graph with a set V of vertices and a set E of edges. For a vertex $v \in V$, the set of neighbors of v in H is denoted by $N_H(v)$, and the *degree* $\deg_H(v)$ of v is defined to be $|N_H(v)|$. The length of a path is defined to be the number of edges in the path. The *distance* $\mathrm{dist}_H(u, v)$ between two vertices $u, v \in V$ is defined to be the minimum length of a path connecting u and v in H. The *diameter* $\mathrm{dia}(H)$ of H is defined to be the maximum distance between two vertices in H. The *sum-distance* $\mathrm{smdt}(H)$ of H is defined to be the sum of distances over all vertex pairs.



$$f(G)=(n(H)=6,\ n_1(H)=3,\ n_2(H)=3,\ n_3(H)=1,\ n_4(H)=0,\ \overline{\mathrm{dia}}(H)=0.667,\ \overline{\mathrm{smdt}}(H)=0.1435,\ n_{\mathtt{C}}(G)=4,\ n_{\mathtt{O}}(G)=1,\ n_{\mathtt{N}}(G)=1,\ \overline{\mathrm{ms}}(G)=13,\ b_2(G)=1,\ b_3(G)=0,\ n_{(\mathtt{C},\mathtt{C},1)}(G)=2,\ n_{(\mathtt{C},\mathtt{C},2)}(G)=1,\ n_{(\mathtt{C},\mathtt{O},1)}(G)=1,\ n_{(\mathtt{C},\mathtt{N},1)}(G)=1)$$

Fig. 1. A chemical graph $G = (H, \alpha, \beta)$ and its feature vector $f(G)$.

Chemical Graphs We represent the graph structure of a chemical compound as a graph with labels on vertices and multiplicity on edges in a hydrogen-suppressed model. Let Λ be a set of labels each of which represents a chemical element such as `C` (carbon), `O` (oxygen), `N` (nitrogen) and so on, where we assume that Λ does not contain `H` (hydrogen). Let $\mathrm{mass}(\mathtt{a})$ and $\mathrm{val}(\mathtt{a})$ denote the mass and valence of a chemical element $\mathtt{a} \in \Lambda$, respectively. In our model, we use integers $\mathrm{mass}^*(\mathtt{a}) = \lfloor 10 \cdot \mathrm{mass}(\mathtt{a}) \rfloor$, $\mathtt{a} \in \Lambda$. We introduce a total order $<$ over the elements in Λ according to their mass values; i.e., we write $\mathtt{a} < \mathtt{b}$ for chemical elements $\mathtt{a}, \mathtt{b} \in \Lambda$ with $\mathrm{mass}(\mathtt{a}) < \mathrm{mass}(\mathtt{b})$. Choose a set $\Gamma_<$ of tuples $\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Lambda \times \Lambda \times [1, 3]$ such that $\mathtt{a} < \mathtt{b}$. For a tuple $\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Lambda \times \Lambda \times [1, 3]$, let $\overline{\gamma}$ denote the tuple $(\mathtt{b}, \mathtt{a}, k)$. Set $\Gamma_> = \{\overline{\gamma} \mid \gamma \in \Gamma_<\}$, $\Gamma_= = \{(\mathtt{a}, \mathtt{a}, k) \mid \mathtt{a} \in$

$\Lambda, k \in [1,3]\}$ and $\Gamma = \Gamma_< \cup \Gamma_=$. A pair of two atoms $\mathtt{a}$ and $\mathtt{b}$ joined with a bond of multiplicity k is denoted by a tuple $\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Gamma$.

A *chemical graph* in a hydrogen-suppressed model is defined to be a tuple $G = (H, \alpha, \beta)$ of a graph $H = (V, E)$, a function $\alpha : V \to \Lambda$ and a function $\beta : E \to [1,3]$ such that (i) H is connected; and (ii) $\sum_{e=uv \in E} \beta(e) \leq \mathrm{val}(\alpha(u))$ for each vertex $u \in V$. Nearly 55% of the acyclic chemical graphs with at most 200 non-hydrogen atoms registered in the chemical database PubChem have maximum degree at most 3 in the hydrogen-suppressed model. Figure 1 illustrates an example of a chemical graph $G = (H, \alpha, \beta)$.

Descriptors In our method, we use only graph-theoretical descriptors for defining a feature vector, which facilitates our designing an algorithm for constructing graphs. Given a chemical graph $G = (H = (V, E), \alpha, \beta)$, we define a *feature vector* $f(G)$ that consists of the following eight kinds of descriptors:
$n(H)$: the number of vertices in H;
$n_d(H)$ ($d \in [1,4]$): the number of vertices of degree d in H;
$\overline{\mathrm{dia}}(H)$: the diameter of H divided by $|V|$;
$\overline{\mathrm{smdt}}(H)$: the sum of distances of H divided by $|V|^3$;
$n_{\mathtt{a}}(G)$ ($\mathtt{a} \in \Lambda$): the number of vertices with label $\mathtt{a} \in \Lambda$;
$\overline{\mathrm{ms}}(G)$: the average of mass* of atoms in G;
$b_i(G)$ ($i = 2, 3$): the number of double and triple bonds;
$n_\gamma(G)$ ($\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Gamma$): the number of label pairs $\{\mathtt{a}, \mathtt{b}\}$ with multiplicity k.
Figure 1 illustrates an example of a feature vector $f(G)$.

3 A Method for Inferring Chemical Graphs

We review how MILPs are used in the method for the inverse QSAR/QSPR [3], which is illustrated in Fig. 2. For a specified chemical property π such as boiling point, we denote by $a(G)$ the observed value of the property π for a given chemical compound G, which is represented by a chemical graph $G = (H, \alpha, \beta)$. As the first phase, we solve (I) Prediction Problem for the inverse QSAR/QSPR with the following three steps.

1. Prepare a data set $D = \{(G_i, a(G_i)) \mid i = 1, 2, \ldots, m\}$ of pairs of a chemical graph G_i and the value $a(G_i)$ for a specified chemical property π. Set reals $\underline{a}, \overline{a} \in \mathbb{R}$ so that $\underline{a} \leq a(G_i) \leq \overline{a}$, $i = 1, 2, \ldots, m$.
2. Set a graph class $\mathcal{G}$ to be a set of chemical graphs such that $\mathcal{G} \supseteq \{G_i \mid i = 1, 2, \ldots, m\}$. Introduce a feature function $f : \mathcal{G} \to \mathbb{R}^k$ for a positive integer k. We call $f(G)$ the *feature vector* of $G \in \mathcal{G}$, and call each entry of a vector $f(G)$ a *descriptor* of G.
3. Construct a prediction function $\psi_\mathcal{N}$ with an ANN $\mathcal{N}$ that, given a vector in $x \in \mathbb{R}^k$, returns a real $\psi_\mathcal{N}(x)$ with $\underline{a} \leq \psi_\mathcal{N}(x) \leq \overline{a}$ so that $\psi_\mathcal{N}(f(G))$ takes a value nearly equal to $a(G)$ for many chemical graphs in D.

See Fig. 2 for an illustration of Steps 1 to 3. As the second phase, we solve (II) Inverse Problem for the inverse QSAR/QSPR by treating the following inference problems.

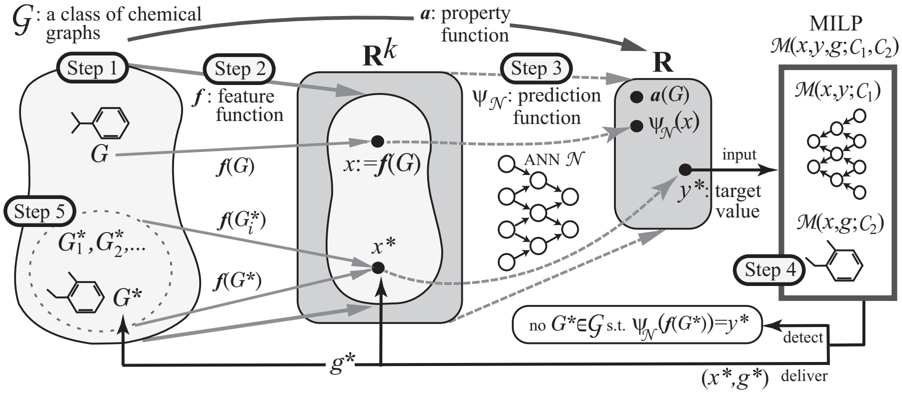


Fig. 2. An illustration of a property function a, a feature function f, a prediction function $\psi_{\mathcal{N}}$ and an MILP that either delivers a vector (x^*, g^*) that forms a chemical graph $G^* \in \mathcal{G}$ such that $\psi_{\mathcal{N}}(f(G^*)) = y^*$ (or $a(G^*) = y^*$) or detects that no such chemical graph G^* exists in $\mathcal{G}$.

(II-a) Inference of Vectors
Input: A real $y^* \in [\underline{a}, \overline{a}]$.
Output: Vectors $x^* \in \mathbb{R}^k$ and $g^* \in \mathbb{R}^h$ such that $\psi_{\mathcal{N}}(x^*) = y^*$ and g^* forms a chemical graph $G^* \in \mathcal{G}$ with $f(G^*) = x^*$.

(II-b) Inference of Graphs
Input: A vector $x^* \in \mathbb{R}^k$.
Output: All graphs $G^* \in \mathcal{G}$ such that $f(G^*) = x^*$.

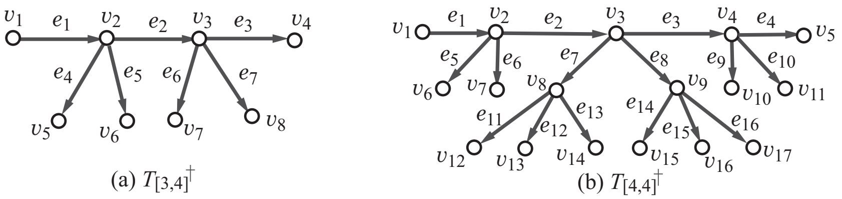
To treat Problem (II-a), we rely on the next result.

Theorem 1 ([2]). *Let $\mathcal{N}$ be an ANN with a piecewise-linear activation function for an input vector $x \in \mathbb{R}^k$, n_A denote the number of nodes in the architecture and n_B denote the total number of break-points over all activation functions. Then there is an MILP $\mathcal{M}(x, y; \mathcal{C}_1)$ that consists of variable vectors $x \in \mathbb{R}^k$, $y \in \mathbb{R}$, and an auxiliary variable vector $z \in \mathbb{R}^p$ for some integer $p = O(n_A + n_B)$ and a set $\mathcal{C}_1$ of $O(n_A + n_B)$ constraints on these variables such that $\psi_{\mathcal{N}}(x^*) = y^*$ if and only if there is a vector (x^*, y^*) feasible to $\mathcal{M}(x, y; \mathcal{C}_1)$.*

We also introduce a variable vector $g \in \mathbb{R}^h$ for some integer h and a set $\mathcal{C}_2$ of constraints on x and g such that (x^*, g^*) is feasible to the MILP $\mathcal{M}(x, g; \mathcal{C}_2)$ if and only if g^* forms a chemical graph $G^* \in \mathcal{G}$ with $f(G^*) = x^*$ (see [3] for details). In MILPs, we can easily impose additional linear constraints or fix some variables to specified constants.

We design an algorithm to Problem (II-b) based on the branch-and-bound method (see [5] for enumerating acyclic chemical compounds).

The second phase consists of the next two steps.



(a) $T_{[3,4]}^{\dagger}$

(b) $T_{[4,4]}^{\dagger}$

Fig. 3. (a) $T^{\dagger}_{[3,4]}$, where $n_{\max}(3,4) = 8$; (b) $T^{\dagger}_{[4,4]}$, where $n_{\max}(4,4) = 17$.

4. Formulate Problem (II-a) as the above MILP $\mathcal{M}(x, y, g; \mathcal{C}_1, \mathcal{C}_2)$ based on $\mathcal{G}$ and $\mathcal{N}$. Find a set F^* of vectors $x^* \in \mathbb{R}^k$ such that $(1 - \varepsilon)y^* \leq \psi_{\mathcal{N}}(x^*) \leq (1 + \varepsilon)y^*$ for a tolerance ε set to be a small positive real.
5. To solve Problem (II-b), enumerate all graphs $G^* \in \mathcal{G}$ such that $f(G^*) = x^*$ for each vector $x^* \in F^*$.

Figure 2 illustrates Steps 4 and 5.

As for an MILP $\mathcal{M}(x, g; \mathcal{C}_2)$, the previous formulation due to Azam et al. [3] is based on an idea that a required acyclic chemical graph G^* with n vertices will be constructed as a subset of $n - 1$ vertex pairs as edges over an $n \times n$ adjacency matrix (or a complete graph K_n with n vertices). In the next section, we introduce a new formulation for an MILP $\mathcal{M}(x, g; \mathcal{C}_2)$ so that G^* will be constructed as an induced subgraph of a larger acyclic graph, which we call "a skeleton tree."

4 MILPs for Representing Acyclic Chemical Graphs

In this section, we propose a new formulation to the MILP $\mathcal{M}(x, g; \mathcal{C}_2)$ in Step 4 of the method introduced in the previous section. We consider acyclic chemical graphs such that the maximum degree is at most 3 or 4.

For an integer D, let $\mathcal{T}_{[D,3]}$ (resp., $\mathcal{T}_{[D,4]}$) denote the set of trees H such that $\mathrm{dia}(H) = D$ and the maximum degree is at most 3 (resp., equal to 4), and define the *skeleton tree* $T^{\dagger}_{[D,d]}$, $d = 3, 4$ to be a tree in $\mathcal{T}_{[D,d]}$ with the maximum number of vertices, where $n_{\max}(D, d)$ denotes the number of vertices in $T^{\dagger}_{[D,d]}$. We assume that the vertices and edges in the skeleton tree $T^{\dagger}_{[D,d]} = (V^{\dagger} = \{v_1, v_2, \ldots, v_{n_{\max}(D,d)}\}, E^{\dagger} = \{e_1, e_2, \ldots, e_{n_{\max}(D,d)-1}\})$ are indexed with the following ordering σ: (i) $T_{[D,d]}$ is rooted at vertex v_1, and $i < j$ holds for any vertex v_i and a child v_j of v_i; (ii) Each edge e_j joins two vertices v_{j+1} and v_k with $k \leq j$, where we denote by $\mathrm{tail}(j)$ the index k of the parent v_k of vertex v_{j+1}; and (iii) For each $i = 1, 2, \ldots, D$, it holds that $v_i v_{i+1} \in E$; i.e., $(e_1, e_2, \ldots, e_D)$ is one of the longest paths in $T^{\dagger}_{[D,d]}$. Let $N_{\sigma}(i)$ for each $i = 1, 2, \ldots, n_{\max}(D, d)$ denote the set of indices j of edges e_j incident to vertex v_i, and $\mathrm{dist}_{\sigma}(i, j)$ denote the distance $\mathrm{dist}_T(v_i, v_j)$ in the tree $T = T^{\dagger}_{[D,d]}$. Figure 3 illustrates such an ordering σ for the skeleton trees $T^{\dagger}_{[3,4]}$ and $T^{\dagger}_{[4,4]}$.

Given integers $n^* \geq 3$, $\mathrm{dia}^* \geq 2$ and $d_{\max} \in \{3, 4\}$, consider an acyclic chemical graph $G = (H = (V, E), \alpha, \beta)$ such that $|V| = n^*$, $\mathrm{dia}(H) = \mathrm{dia}^*$ and the maximum degree in H is at most 3 for $d_{\max} = 3$ (or equal to 4 for $d_{\max} = 4$). We formulate the MILP $\mathcal{M}(x, g; \mathcal{C}_2)$ so that the underlying graph H is an induced subgraph of the skeleton tree $T^{\dagger}_{[\mathrm{dia}^*,d_{\max}]}$ such that $\{v_1, v_2, \ldots, v_{D+1}\} \subseteq V$.

To reduce the number of graph-isomorphic solutions in this MILP, we introduce a precedence constraint as follows. Let P_{prc} be a set of ordered index pairs (i, j) with $D + 2 \leq i < j \leq n_{\max}$. We call P_{prc} *proper* if the next conditions hold:

(a) For each pair of a vertex v_j and its parent v_i with $D + 2 \leq i < j \leq n_{\max}$ in $T^{\dagger}_{[\mathrm{dia}^*,d_{\max}]}$, there is a sequence $(i_1, i_2), (i_2, i_3), \ldots, (i_{k-1}, i_k)$ of index pairs in P_{prc} such that $i_1 = i$ and $i_k = j$; and

(b) Each subtree $H = (V, E)$ of $T^{\dagger}_{[\mathrm{dia}^*,d_{\max}]}$ with $\{v_1, v_2, \ldots, v_{D+1}\} \subseteq V$ is isomorphic to a subtree $H' = (V', E')$ with $\{v_1, v_2, \ldots, v_{D+1}\} \subseteq V'$ such that for each pair $(i, j) \in P_{\mathrm{prc}}$, if $v_j \in V'$ then $v_i \in V'$.

Note that a proper set P_{prc} is not necessarily unique. For example, we use $P_{\mathrm{prc}} = \{(5, 6), (7, 8), (6, 8)\}$ for the tree $T^{\dagger}_{[3,4]}$ in Fig. 3(a).

For a technical reason, we introduce a dummy chemical element ϵ, and denote by Γ_0 the set of dummy tuples (ϵ, ϵ, k), $(\epsilon, \mathtt{a}, k)$ and $(\mathtt{a}, \epsilon, k)$ ($\mathtt{a} \in \Lambda$, $k \in [0, 3]$). To represent elements $\mathtt{a} \in \Lambda \cup \{\epsilon\} \cup \Gamma_< \cup \Gamma_= \cup \Gamma_>$ in an MILP, we encode these elements $\mathtt{a}$ into some integers denoted by $[\mathtt{a}]$, where we assume that $[\epsilon] = 0$. For simplicity, we also denote n^* by n and $n_{\max}(\mathrm{dia}^*, d_{\max})$ by $n_{\max}$. Our new formulation is given as follows.

MILP $\mathcal{M}(x, g; \mathcal{C}_2)$

variables for descriptors in x:

$\mathrm{n}(d) \in [0, n]$ $(d \in [1, 4])$; $\mathrm{smdt} \in [0, n^3]$; $\mathrm{n}(\mathtt{a}) \in [0, n]$ $(\mathtt{a} \in \Lambda)$;
$\mathrm{Mass} \in \mathbb{Z}$; $\mathrm{b}(k) \in [0, n - 1]$ $(k \in [1, 3])$; $\mathrm{n}(\gamma) \in [0, n - 1]$ $(\gamma \in \Gamma_< \cup \Gamma_=)$

variables for constructing H in g:

$v(i) \in \{0, 1\}$ $(i \in [1, n_{\max}])$; $\delta_{\deg}(i, d) \in \{0, 1\}$ $(i \in [1, n_{\max}],\ d \in [0, 4])$;
$\deg(i) \in [0, 4]$ $(i \in [1, n_{\max}])$; $\mathrm{dist}(i, j) \in [0, \mathrm{dia}^*]$ $(1 \leq i < j \leq n_{\max})$;
$\widetilde{\alpha}(i) \in \{[\mathtt{a}] \mid \mathtt{a} \in \Lambda \cup \{\epsilon\}\}$ $(i \in [1, n_{\max}])$; $\widetilde{\beta}(j) \in [0, 3]$ $(j \in [1, n_{\max}-1])$;
$\delta_\alpha(i, \mathtt{a}) \in \{0, 1\}$ $(i \in [1, n_{\max}],\ \mathtt{a} \in \Lambda \cup \{\epsilon\})$;
$\delta_\beta(j, k) \in \{0, 1\}$ $(j \in [1, n_{\max}-1],\ k \in [0, 3])$;
$\delta_\tau(j, \gamma) \in \{0, 1\}$ $(j \in [1, n_{\max}-1], \gamma \in \Gamma \cup \Gamma_0)$

constraints in $\mathcal{C}_2$:

$$\sum_{i \in [1, n_{\max}]} v(i) = n; \qquad v(i) = 1 \ (i \in [1, \mathrm{dia}^* + 1]);$$

$$v(i) \geq v(j) \ ((i, j) \in P_{\mathrm{prc}}); \quad \sum_{i \in [1, n_{\max}]} \delta_{\deg}(i, d) = \mathrm{n}(d) \ (d \in [0, 4]);$$

$$\begin{aligned}
&\sum_{i\in[1,n_{\max}-1]} \delta_\beta(i,k) = \mathrm{b}(k)\ (k\in[1,3]);\quad \sum_{1\le i<j\le n_{\max}} \mathrm{dist}(i,j) = \mathrm{smdt};\\
&\sum_{i\in[1,n_{\max}]} \delta_\alpha(i,\mathtt{a}) = \mathrm{n}(\mathtt{a})\ (\mathtt{a}\in\Lambda);\quad \sum_{\mathtt{a}\in\Lambda} \mathrm{mass}^*(\mathtt{a})\cdot \mathrm{n}(\mathtt{a}) = \mathrm{Mass};\\
&\sum_{i\in[1,n_{\max}-1]} (\delta_\tau(i,\gamma)+\delta_\tau(i,\overline{\gamma}))=\mathrm{n}(\gamma)\ (\gamma\in\Gamma_<);\quad \sum_{i\in[1,n_{\max}-1]} \delta_\tau(i,\gamma)=\mathrm{n}(\gamma)\ (\gamma\in\Gamma_=);
\end{aligned}$$

For each $i = 1, 2, \dots, n_{\max}$,

$$\begin{aligned}
&\sum_{j\in N_\sigma(i)} v(j+1) = \deg(i),\quad \sum_{d\in[0,4]} \delta_{\deg}(i,d) = 1,\quad \sum_{\mathtt{a}\in\Lambda} \delta_\alpha(i,\mathtt{a}) = v(i),\\
&\sum_{i\in[1,n_{\max}]} \delta_\alpha(i,\mathtt{a}) = \mathrm{n}(\mathtt{a}),\quad \sum_{j\in N_\sigma(i)} \widetilde{\beta}(j) \le \sum_{\mathtt{a}\in\Lambda} \mathrm{val}(\mathtt{a})\cdot\delta_\alpha(i,\mathtt{a});
\end{aligned}$$

For each pair (i,j) with $1 \le i < j \le n_{\max}$,

$$\begin{aligned}
&\mathrm{dist}(i,j) \le \mathrm{dia}^*\cdot v(i),\quad \mathrm{dist}(i,j) \ge \mathrm{dist}_\sigma(i,j) - \mathrm{dia}^*\cdot(2-v(i)-v(j)),\\
&\mathrm{dist}(i,j) \le \mathrm{dia}^*\cdot v(j),\quad \mathrm{dist}(i,j) \le \mathrm{dist}_\sigma(i,j) + \mathrm{dia}^*\cdot(2-v(i)-v(j));
\end{aligned}$$

For each $j = 1, 2, \dots, n_{\max}-1$,

$$\begin{aligned}
&v(j+1) \le \widetilde{\beta}(j) \le 3v(j+1),\quad \sum_{k\in[1,3]} \delta_\beta(j,k) = v(j+1),\quad \sum_{k\in[1,3]} k\delta_\beta(j,k) = \widetilde{\beta}(j),\\
&\sum_{\gamma\in\Gamma\cup\Gamma_0} \delta_\tau(j,\gamma) = 1,\quad \sum_{(\mathtt{a},\mathtt{b},k)\in\Gamma\cup\Gamma_0} [\mathtt{a}]\delta_\tau(j,(\mathtt{a},\mathtt{b},k)) = \widetilde{\alpha}(\mathrm{tail}(j)),\\
&\sum_{(\mathtt{a},\mathtt{b},k)\in\Gamma\cup\Gamma_0} [\mathtt{b}]\delta_\tau(j,(\mathtt{a},\mathtt{b},k)) = \widetilde{\alpha}(j+1),\quad \sum_{(\mathtt{a},\mathtt{b},k)\in\Gamma\cup\Gamma_0} k\delta_\tau(j,(\mathtt{a},\mathtt{b},k)) = \widetilde{\beta}(j).
\end{aligned}$$

5 Experimental Results

We implemented our new formulation for the MILP $\mathcal{M}(x, y, g; \mathcal{C}_1, \mathcal{C}_2)$ in Step 4 of the method for the inverse QSAR/QSPR [3], and conducted some experiments to compare the practical performance of the previous and new MILP formulations. We executed the experiments on a PC with Intel Core i5 1.6 GHz CPU and 8 GB of RAM running under the Mac OS operating system version 10.14.4. As a study case, we selected three chemical properties: heat of atomization (Ha), octanol/water partition coefficient (Kow) and heat of combustion (Hc).

Results on Phase 1. In Step 1, we collected a data set D of acyclic chemical graphs for Ha from the article [15] and for Kow and Hc from HSDB from PubChem database. We set Λ to be the set of all chemical elements of the chemical graphs in D; and Γ to be the set of all tuples $\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Lambda \times \Lambda \times [1, 3]$ of the chemical graphs in D. In Step 2, we set a graph class $\mathcal{G}$ to be the set of all possible acyclic chemical graphs over the sets Λ and Γ in Step 1. In Step 3, we used `scikit-learn` (version 0.21.3) to construct ANNs $\mathcal{N}$ where the tool and activation function are set to be MLPRegressor and Relu, respectively. To evaluate the performance of the resulting prediction function $\psi_{\mathcal{N}}$ with cross-validation,

Table 1. The results on Steps 1, 2 and 3

		Steps 1 and 2			Step 3			
π	$\lvert D\rvert$	Λ	$[\underline{n}, \overline{n}]$	$[\underline{a}, \overline{a}]$	K	Arch.	L-time	Test R^2
Ha	128	C,O,S	[2,11]	[450.3,3009.6]	19	$\langle 10\rangle$	11.24	0.999
Kow	62	C,O,S	[2,16]	[-0.77,8.2]	19	$\langle 10, 10\rangle$	0.373	0.969
Kow	430	C,Cl,O,N,S,Br,F	[1,36]	[-4.2,15.6]	42	$\langle 30\rangle$	1.155	0.907
Hc	204	C,O,N	[1,63]	[245.6,35099.6]	26	$\langle 10, 10\rangle$	23.01	0.965
Hc	222	C,O,N,F,S,Br	[1,63]	[245.6,35099.6]	34	$\langle 30, 30\rangle$	17.376	0.961
Hc	262	C,Cl,O,N,S,Br,F,Si,B,P	[1,63]	[245.6,35099.6]	47	$\langle 30, 30\rangle$	5.194	0.965

we partition a given data set D into five subsets D_i, $i \in [1, 5]$ randomly, where $D \setminus D_i$ is used for a training set and D_i is used for a test set in five trials $i \in [1, 5]$.

Table 1 shows the size and range of data sets that we prepared for each chemical property, and results on Phase 1, where we denote the following:
π: one of the chemical properties Ha, Kow, and Hc;
$|D|$: the size of data set D for a chemical property π;
Λ: the set of all chemical elements of the chemical graphs in D;
$[\underline{n}, \overline{n}]$: the minimum and maximum number of vertices in H over data set D;
$[\underline{a}, \overline{a}]$: the minimum and maximum values of $a(G)$ over data set D;
K: the number of descriptors in $f(G)$ for a chemical property π,
where $K = |\Lambda| + |\Gamma| + 12$ for our feature vector $f(G)$;
Arch.: the size of hidden layers of ANNs, where $\langle 10\rangle$ (resp., $\langle 30, 30\rangle$)
means an architecture $(K, 10, 1)$ with an input layer with K nodes,
a middle layer with 10 nodes (resp., two hidden layers with 30 nodes)
and an output layer with a singe node;
L-time: the average time (sec.) to construct ANNs for each trial;
Test R^2: the coefficient of determination averaged over the five test sets.

Results on Phase 2. Let us call the previous MILP formulation based on adjacency matrix the AM method, and our new MILP formulation based on skeleton tree the ST method. We solved MILP instances in these methods using CPLEX (ILOG CPLEX version 12.9) [7]. We conducted an experiment for each property in {Ha, Kow, Hc} as follows. For several pairs $(d_{\max}, \mathrm{dia}^*)$ of integers $d_{\max} \in \{3, 4\}$ and $\mathrm{dia}^* \in [6, 13]$, choose each integer $n^* \in [14, n_{\max}(\mathrm{dia}^*, d_{\max})]$ and six target values y_i^*, $i \in [1, 6]$. We tried to solve the six MILP instances with the AM and ST methods starting with $n^* = 14$ and increasing n^* up to $n_{\max}(\mathrm{dia}^*, d_{\max})$. We terminated solving instances with each of the two methods whenever the running time of solving one of the six instances reaches a time limit of 300 seconds.

Table 2 and Figure 4 show the computation time of the AM and ST methods in Step 4 for property Ha, where we denote the following:
AM: the average time (sec.) to solve six instances with the method AM;
ST: the average time (sec.) to solve six instances with the method ST;
T.O.: the running time of one of the six instances exceeded 300 seconds.

Table 2. The computation time of the methods AM and ST for Ha

			$d_{\max}=3$						$d_{\max}=4$			
	$\mathrm{dia}^*=8$		$\mathrm{dia}^*=12$		$\mathrm{dia}^*=13$		$\mathrm{dia}^*=6$		$\mathrm{dia}^*=8$		$\mathrm{dia}^*=9$	
n^*	AM	ST	AM	ST	AM	ST	AM	ST	AM	ST	AM	ST
14	0.037	0.244	0.020	0.471	0.012	0.185	0.503	0.146	0.218	3.375	0.212	7.242
17	0.999	0.074	0.272	1.712	0.215	3.380	1.657	0.159	1.212	10.194	1.016	14.921
20	3.886	0.493	1.254	5.576	0.920	6.531	8.628	1.326	3.648	13.317	3.770	27.347
23	9.947	0.523	4.999	9.796	3.440	13.741	12.247	1.228	8.937	17.716	7.059	T.O.
26	50.422	0.539	10.713	17.758	8.613	32.541	73.451	1.522	22.141	24.742	20.998	-
29	T.O.	0.601	24.904	13.587	T.O.	94.952	T.O.	1.206	66.298	40.580	T.O.	-
32	-	0.554	T.O.	25.978	-	121.37	-	1.572	T.O.	33.445	-	-
36	-	0.252	-	46.940	-	77.627	(2×10^4)	1.391	-	35.968	-	-
38	-	0.722	-	19.028	-	139.25	(1×10^5)	2.133	-	42.081	-	-
41	-	0.445	-	28.145	-	90.231	-	1.141	-	30.332	-	-
44	-	0.152	-	60.771	-	T.O.	-	1.258	-	41.772	-	-
47	n.a.	n.a.	-	71.990	-	-	-	0.799	-	34.532	-	-
50	n.a.	n.a.	-	72.139	-	-	-	0.445	-	T.O.	-	-
53	n.a.	n.a.	-	T.O.	-	-	-	0.050	-	-	-	-

We executed the method AM without a time limit for an instance with $n^*=36$ (resp., $n^*=38$ and 40) for Ha, $\mathrm{dia}^*=6$ and $d_{\max}=4$, and the computation time was 21,962 (resp., 124,903 and 148,672) seconds. The method ST took 2.133 seconds for the same size of instances with $n^*=38$. This means that the method ST was 58,557 times faster than the method AM for this size of instances.

For space limitation, we describe a summary of the experimental results on instances for Kow and Hc: The method ST outperforms the method AM for the cases of $(\pi=\mathrm{K_{ow}}, |\Lambda|=3, d_{\max}=3, \mathrm{dia}^*\leq 11)$, $(\pi=\mathrm{K_{ow}}, |\Lambda|=3, d_{\max}=4, \mathrm{dia}^*\leq 7)$, $(\pi=\mathrm{K_{ow}}, |\Lambda|=7, d_{\max}=3, \mathrm{dia}^*\leq 8)$, $(\pi=\mathrm{K_{ow}}, |\Lambda|=7, d_{\max}=4, \mathrm{dia}^*\leq 5)$, $(\pi=\mathrm{Hc}, |\Lambda|=3, d_{\max}=3, \mathrm{dia}^*\leq 9)$, $(\pi=\mathrm{Hc}, |\Lambda|=3, d_{\max}=4, \mathrm{dia}^*\leq 6)$, $(\pi=\mathrm{Hc}, |\Lambda|=6, d_{\max}=3, \mathrm{dia}^*\leq 8)$, $(\pi=\mathrm{Hc}, |\Lambda|=6, d_{\max}=4, \mathrm{dia}^*\leq 7)$, $(\pi=\mathrm{Hc}, |\Lambda|=10, d_{\max}=3, \mathrm{dia}^*\leq 7)$ and $(\pi=\mathrm{Hc}, |\Lambda|=10, d_{\max}=4, \mathrm{dia}^*\leq 5)$.

From the experimental results, we observe that the ST method solves the problem of Step 4 faster than the AM method does when the diameter of graphs is up to around 11 for $d_{\max}=3$ and 8 for $d_{\max}=4$. Here we note that chemical graphs with diameter up to 11 for $d_{\max}=3$ and 8 for $d_{\max}=4$ account for about 35 % and 18 %, respectively, out of all acyclic chemical graphs with at most 200 non-hydrogen atoms registered in the PubChem chemical database, and about 63 % and 40 % out of the acyclic chemical graphs with at most 200 non-hydrogen atoms with $d_{\max}=3$ and $d_{\max}=4$, respectively.

6 Concluding Remarks

In this paper, we presented a new formulation of an MILP for inferring acyclic chemical graphs based on the method due to Azam et al. [3]. In the previous

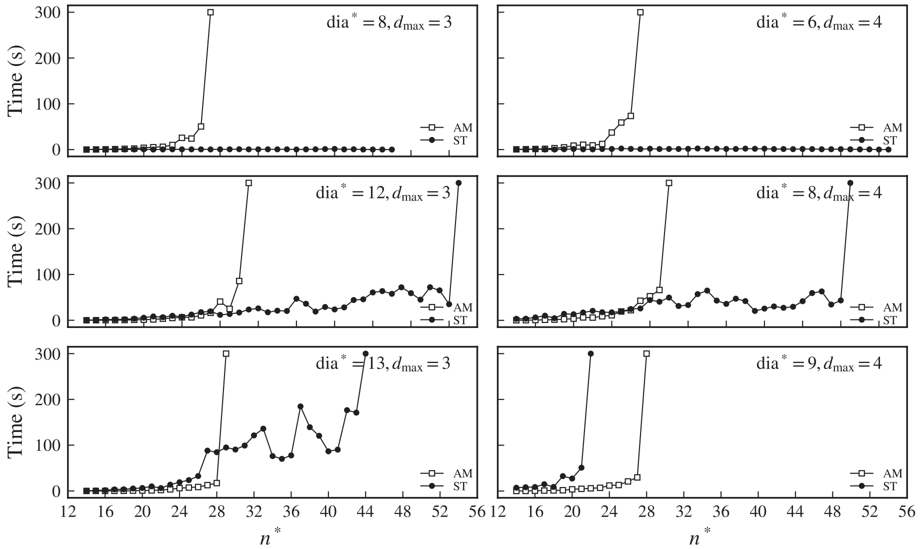


Fig. 4. The average computation time of the methods AM and ST for HA

formulation, a tree H with n vertices was represented as a subset of vertex pairs over an $n \times n$ adjacency matrix. Contrary to this, our new formulation is based on a maximum tree of a specified diameter, called the skeleton tree, from which a required tree H is chosen as an induced subgraph. From the results on some computational experiments, we observe that the proposed method is more efficient than the previous method when the number of chemical elements and diameter are relatively small. Although this paper presented such an MILP for the class $\mathcal{G}$ of acyclic chemical graphs, we can apply a similar formulation to the acyclic part of any chemical graph.

References

1. Akutsu, T., Fukagawa, D., Jansson, J., Sadakane, K.: Inferring a graph from path frequency. Discrete Applied Mathematics **160**(10-11), 1416–1428 (2012)
2. Akutsu, T., Nagamochi, H.: A mixed integer linear programming formulation to artificial neural networks. In: Proceedings of the 2nd International Conference on Information Science and Systems. pp. 215–220. ACM (2019)
3. Azam, N.A., Chiewvanichakorn, R., Zhang, F., Shurbevski, A., Nagamochi, H., Akutsu, T.: A method for the inverse QSAR/QSPR based on artificial neural networks and mixed integer linear programming. In: Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies – Volume 3: BIOINFORMATICS. pp. 101–108 (2020)
4. Chiewvanichakorn, R., Wang, C., Zhang, Z., Shurbevski, A., Nagamochi, H., Akutsu, T.: A method for the inverse QSAR/QSPR based on artificial neural

networks and mixed integer linear programming. In: ICBBB2020, paper K0013 (2020)

5. Fujiwara, H., Wang, J., Zhao, L., Nagamochi, H., Akutsu, T.: Enumerating treelike chemical graphs with given path frequency. Journal of Chemical Information and Modeling **48**(7), 1345–1357 (2008)
6. Gómez-Bombarelli, R., Wei, J.N., Duvenaud, D., Hernández-Lobato, J.M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T.D., Adams, R.P., Aspuru-Guzik, A.: Automatic chemical design using a data-driven continuous representation of molecules. ACS Central Science **4**(2), 268–276 (2018)
7. IBM ILOG: CPLEX Optimization Studio 12.9, `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.9.0/ilog.odms.cplex.help/CPLEX/homepages/usr`
8. Ikebata, H., Hongo, K., Isomura, T., Maezono, R., Yoshida, R.: Bayesian molecular design with a chemical language model. Journal of Computer-aided Molecular Design **31**(4), 379–391 (2017)
9. Kerber, A., Laue, R., Grüner, T., Meringer, M.: MOLGEN 4.0. Match Communications in Mathematical and in Computer Chemistry (37), 205–208 (1998)
10. Kusner, M.J., Paige, B., Hernández-Lobato, J.M.: Grammar variational autoencoder. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 1945–1954 (2017)
11. Li, J., Nagamochi, H., Akutsu, T.: Enumerating substituted benzene isomers of tree-like chemical graphs. IEEE/ACM Transactions on Computational Biology and Bioinformatics **15**(2), 633–646 (2016)
12. Miyao, T., Kaneko, H., Funatsu, K.: Inverse QSPR/QSAR analysis for chemical structure generation (from y to x). Journal of Chemical Information and Modeling **56**(2), 286–299 (2016)
13. Nagamochi, H.: A detachment algorithm for inferring a graph from path frequency. Algorithmica **53**(2), 207–224 (2009)
14. Reymond, J.L.: The chemical space project. Accounts of Chemical Research **48**(3), 722–730 (2015)
15. Roy, K., Saha, A.: Comparative QSPR studies with molecular connectivity, molecular negentropy and TAU indices. Journal of Molecular Modeling **9**(4), 259–270 (2003)
16. Rupakheti, C., Virshup, A., Yang, W., Beratan, D.N.: Strategy to discover diverse optimal molecules in the small molecule universe. Journal of Chemical Information and Modeling **55**(3), 529–537 (2015)
17. Segler, M.H.S., Kogej, T., Tyrchan, C., Waller, M.P.: Generating focused molecule libraries for drug discovery with recurrent neural networks. ACS Central Science **4**(1), 120–131 (2017)
18. Skvortsova, M.I., Baskin, I.I., Slovokhotova, O.L., Palyulin, V.A., Zefirov, N.S.: Inverse problem in QSAR/QSPR studies for the case of topological indices characterizing molecular shape (Kier indices). Journal of Chemical Information and Computer Sciences **33**(4), 630–634 (1993)
19. Yang, X., Zhang, J., Yoshizoe, K., Terayama, K., Tsuda, K.: ChemTS: an efficient python library for de novo molecular generation. Science and Technology of Advanced Materials **18**(1), 972–976 (2017)