A Mixed Integer Linear Programming Formulation to Artificial Neural Networks

Tatsuya Akutsu[1] and Hiroshi Nagamochi[2]

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

[2]Department of Applied Mathematics and Physics, Kyoto University
nag@amp.i.kyoto-u.ac.jp

Abstract Let a system $\mathcal{S} = (G = (V, E), w, F)$ consist of a digraph G (not necessarily acyclic) with a set V of vertices and a set E of edges, a weight function $w : V \cup E \to \mathbb{R}$ and a set F of functions $f_v : \mathbb{R} \to \mathbb{R}$, $v \in V$, where $w(u, v)$ denotes the weight of an edge (u, v) from a vertex $u \in V$ and a vertex $v \in V$. A solution to system $\mathcal{S}$ is defined to be a set of reals y_v, $v \in V$ such that $y_v = f_v(w(v)+\sum_{(u,v)\in E} w(u, v)y_u)$. Finding solutions to a given system has an important application in Artificial Neural Network (ANN). In this paper, we show that when each function f_v is a continuous piece-wise linear function, the problem of finding a solution to a system $\mathcal{S}$ can be formulated as a Mixed Integer Linear Programming Problem (MILP) with $O(|V| + n_{\mathrm{b}})$ variables and constraints, where n_{b} denotes the total number of break points over all functions f_v, $v \in V$. Based on this, we can solve the inverse problem to an ANN $\mathcal{N}$ as an MILP after approximating the activation function in $\mathcal{N}$ as a piece-wise linear function.

1 Introduction

Computational design of a novel chemical compound that has desirable properties is an important challenge in information science because it may lead to discovery of new and

[1]Technical Report 2019-001, January 23, 2019

useful drugs and materials. To this end, extensive studies have been done under the name of inverse QSAR/QSPR (quantitative structure-activity and structure-property relationships) [13, 21]. This problem can be formulated as computation of a graph structure representing a chemical compound that maximizes (or minimizes) an objective function under various constraints, where objective functions are often derived from a set of training data consisting of known molecules and their activities/properties using statistical and/or machine learning methods. Various heuristic and statistical methods have been developed for finding optimal or near optimal graph structures under given objective functions [7, 13, 17]. In QSAR/QSPR, chemical compounds are often represented as a vector of real or integer numbers, which is called a feature vector or (a set of) descriptors. Therefore, it is an important subtask in inverse QSAR/QSPR to infer or enumerate graph structures from a given feature vector. Extensive studies have also been done [8, 16] for enumerating chemical graphs from a given feature vector, which is a molecular formula in the simplest case. In our previous studies, we analyzed the computational complexity of this inference problem [1, 14] and developed efficient enumeration algorithms [2, 11].

Recently, novel approaches have been proposed for design of novel chemical compounds, based on the significant progress of Artificial Neural Network (ANN) and deep learning technologies. For example, methods using variational autoencoder [4], grammar variational autoencoder [10], and recurrent neural networks [20, 22] have been developed. In these approaches, ANNs are trained using existing chemical compound data and then novel chemical graphs are obtained by solving a kind of inverse problem on ANN, in which an input vector of real numbers is computed from given ANN and output vector. In order to solve this inverse problem or its variants, various statistical methods have been employed. However, the optimality of the solution is not necessarily guaranteed by statistical methods. Therefore, an integer linear programming (ILP)-based method has been proposed for solving a kind of inverse problem on ANNs with linear threshold functions [12]. However, linear threshold functions are not widely used in recent ANNs, instead, sigmoid functions and ReLU functions have been widely used. Therefore, in this work, we develop novel methods for solving the inverse problem on ANNs with ReLU functions and sigmoid functions. Since it is known that the inverse problem is NP-hard even for ANNs with linear threshold functions [12], we emply Mixed Integer Linear Programming Problem (MILP) formulations, where MILP is one of widely used approaches to solving NP-hard problems. In our proposed methods, activation functions on neurons are represented as piece-wise linear functions, which can exactly represent ReLU functions and well approximate sigmoid functions. The important feature of our proposed methods is that the inverse problem is efficiently encoded into MILP: the resulting MILP instance consists of $O(|V| + n_{\rm b})$ variables and constraints, where V is a set of neurons in a given ANN and $n_{\rm b}$ denotes the total number of break points over all functions f_v, $v \in V$. In this paper, we focus on theoretical aspects of our MILP formulations and prove their theoretical properties.

The paper is organized as follows. Section 2 reviews basic notions on MILP and introduces a "system" as an generalization of ANN. Section 3 presents a method of representing piece-wise linear functions as MILPs. Section 4 shows how to represent a system as an MILP so that the solutions to a "system" is equal to the feasible solutions to the MILP. Section 5 presents MILPs for ANNs with some types of activation functions. Section 6 makes some concluding remarks including a preliminary result on the practical efficiency of our proposed approach.

2 Preliminary

Let $\mathbb{R}$ and $\mathbb{R}_+$ denote the sets of reals and non-negative reals, respectively. For two reals $a, b \in \mathbb{R}$, define sets of reals as follows:

$[a, b] \triangleq \{c \in \mathbb{R} \mid a \leq c \leq b\}$, $(a, b] \triangleq \{c \in \mathbb{R} \mid a < c \leq b\}$,
$[a, b) \triangleq \{c \in \mathbb{R} \mid a \leq c < b\}$, $(a, b) \triangleq \{c \in \mathbb{R} \mid a < c < b\}$,
$(-\infty, b] \triangleq \{c \in \mathbb{R} \mid c \leq b\}$, $(-\infty, b) \triangleq \{c \in \mathbb{R} \mid c < b\}$,
$[a, \infty) \triangleq \{c \in \mathbb{R} \mid a \leq c\}$, $(a, \infty) \triangleq \{c \in \mathbb{R} \mid a < c\}$.

Let $\mathbb{Z}$ denote the set of integers. For a set X of elements and a real x_v for each element $v \in X$, we may denote a set $\{x_v \mid v \in X\}$ as a vector of these elements, denote by x; i.e., $x = (x_1, x_2, \ldots, x_n)$ when $X = \{1, 2, \ldots, n\}$.

Mixed Integer Linear Programming Problem Given positive integers n and m, reals $a_{i,j}$, b_i and c_j, $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$ and a subset $J \subseteq \{1, 2, \ldots, n\}$, the following problem is called an *integer programming problem* or an *integer linear programming problem*.

MILP(a, b, c):

constants

$$a_{i,j}, \qquad i = 1, 2, \ldots, m, j = 1, 2, \ldots, n$$

$$b_i, \qquad i = 1, 2, \ldots, m$$

$$c_j, \qquad j = 1, 2, \ldots, n$$

real variables

$$x_j \geq 0, \qquad j = 1, 2, \ldots, n$$

integer variables

$$x_j \in \mathbb{Z}, \qquad j \in J$$

subject to

$$\sum_{j=1}^{n} a_{i,j} x_j \geq b_i, \qquad i = 1, 2, \ldots, m$$

objective

$$\text{maximize} \sum_{j=1}^{n} c_j x_j.$$

When $J \neq \{1, 2, \ldots, n\}$, the problem is called a *mixed integer linear programming problem* (MILP for short). A *feasible solution* to the problem is defined to be a set of values for variables x_j, $j = 1, 2, \ldots, n$ that satisfies the constraint $\sum_{j=1}^{n} a_{i,j} x_j \geq b_i$ of inequality for each $i = 1, 2, \ldots, m$. An *optimal solution* to the problem is defined to be a feasible solution that maximizes the objective function $\sum_{j=1}^{n} c_j x_j$, where the value of objective function attained by an optimal solution is called the *optimal value*. Given an MILP instance I, let $\mathcal{F}(I)$ denote the set of feasible solutions to I, and $\mathcal{OPT}(I)$ denote the set of optimal solutions to I. For a subset $\{x_i \mid i \in X\}$ of variables, where $X \subseteq \{1, 2, \ldots, n\}$ in the above instance $I = \mathbf{MILP}(a, b, c)$, let $\mathcal{F}(x; I)$ denote the set of vectors $a = (a_{i_1}, a_{i_2}, \ldots, a_{i_k})$, $X = \{i_1, i_2, \ldots, i_k\}$ such that there is a feasible solution $x \in \mathcal{F}(I)$ such that $x_{i_j} = a_{i_j}$ for each $i_j \in X$.

When $J = \emptyset$, the problem is a linear programming problem (LP for short). It is known that LP can be solved in polynomial time [9]. In general, MILP is an NP-hard problem. One simple reason for this is that MILP can represent many discrete optimization problems within a polynomial reduction, including several NP-hard problems such as the travelling salesman problem (see [3] for details on NP-hardness). We also use MILP to represent problems on ANN. Although MILP is NP-hard, there have been many results on theory and practice for designing exact algorithms to solve MILP

[15, 18, 19]. One of efficient softwares for solving LP and MILP is CPLEX [6].

Graphs A digraph is called *simple* if it has neither of self-loops and multiple edges. Let $G = (V, E)$ be a simple digraph with a set V of vertices and a set E of edges. For each edge $e \in E$, let $V(e)$ denote the set of end-vertices of e, and e is denoted by a pair (u, v) of the tail $u \in V(e)$ and the head $v \in V(e)$, where e is directed from u to v. For each vertex $v \in V$, a vertex $u \in V$ with $(u, v) \in E$ (resp., $(v, u) \in E$) is called an *in-neighbor* (resp., *out-neighbor*) of v, and we let $N^-(v)$ and $N^+(v)$ denote the sets of in-neighbors and out-neighbors of v, respectively, and define the *in-degree* $d^-(v)$ and the *out-degree* $d^+(v)$ of a vertex $v \in V$ to be $|N^-(v)|$ and $|N^+(v)|$, respectively. A vertex $v \in V$ with $d^-(v) = 0$ (resp., $d^+(v) = 0$) is called a *source* (resp., *sink*) in G. We let V_{in} and V_{out} denote the sets of sources and sinks in G, respectively.

A digraph is called *acyclic* or a *DAG* (directed acyclic graph) if it does not contain any directed cycle. A digraph $G = (V, E)$ is called *layered* if it is a DAG and the length of any path from a source $s \in V_{\text{in}}$ to a sink $t \in V_{\text{out}}$ is a constant, say k, where V is partitioned into $k+1$ disjoint subsets V_0 $(= V_{\text{in}})$, $V_1, V_2, \ldots, V_k$ $(= V_{\text{out}})$ so that each edge (u, v) satisfies $u \in V_i$ and $v \in V_{i+1}$ for some i. Let $n = |V|$, $n_{\text{in}} = |V_{\text{in}}|$ and $n_{\text{out}} = |V_{\text{out}}|$.

Network Systems We define a *network system* $\mathcal{S} = (G, F)$ to be a pair of a digraph $G = (V, E)$ and a set F of functions $f_v : \mathbb{R}^{d^-(v)} \to \mathbb{R}$, $v \in V \setminus V_{\text{in}}$. We let y denote a vector of reals y_v, $v \in V$; i.e., $y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^+$ for $V = \{1, 2, \ldots, n\}$.

We call a set $\{y_v \mid v \in V\}$ of reals (or a vector $y \in \mathbb{R}^V$ on V) *admissible* to system $\mathcal{S}$ if they satisfy the following condition:

$$\begin{array}{c} y_v = f_v(y_{u_1}, y_{u_2}, \ldots, y_{u_d}) \text{ for each vertex } v \in V \setminus V_{\text{in}} \text{ with} \\ N^-(v) = \{u_1, u_2, \ldots, u_d\} \ (d = d^-(v)). \end{array} \tag{1}$$

Let $\mathcal{A}(\mathcal{S})$ denote the set of admissible vectors $y \in \mathbb{R}^V$ to a network system $\mathcal{S}$. Given a network system $\mathcal{S}$, our aim is to find an admissible vector to the network system. In some cases, part of vector y may be required to be fixed as prescribed values. For example, if variables y_s for each source $s \in V_{\text{in}}$ in a system is prescribed, then the problem is described as follows.

Forward Problem$(\mathcal{S}, V_{\text{in}}, \alpha)$:
Input: A network system $\mathcal{S} = (G, F)$ and a set $\{\alpha_s \in \mathbb{R} \mid s \in V_{\text{in}}\}$ of reals.
Output: A set $\{y_t \mid t \in V_{\text{out}}\}$ of reals such that there is a vector $y \in \mathcal{A}(\mathcal{S})$ such that $y_s = \alpha_s$ for each source $s \in V_{\text{in}}$.

Note that the underlying graph G in a network system $\mathcal{S}$ is not necessarily a DAG, and an admissible set to the forward problem may not be uniquely determined. When G is a DAG, we easily see that an admissible set to the forward problem can be uniquely determined from the sources to the sinks according to (1). Analogously when variables y_t for each sink $t \in V_{\text{out}}$ in a system is prescribed, the problem is described as follows.

Backward Problem$(\mathcal{S}, V_{\text{out}}, \beta)$:
Input: A network system $\mathcal{S} = (G, F)$ and a set $\{\beta_t \in \mathbb{R} \mid t \in V_{\text{out}}\}$ of reals.
Output: A set $\{y_s \mid s \in V_{\text{in}}\}$ of reals such that there is a vector $y \in \mathcal{A}(\mathcal{S})$ such that $y_t = \beta_t$ for each sink $t \in V_{\text{out}}$.

Weight Systems When all functions f_v, $v \in V \setminus V_{\text{in}}$ in a network system are linear functions, it is not difficult to formulate a linear programming problem $\mathbf{LP}(\mathcal{S})$ so that the set of admissible vectors corresponds to the set of feasible vectors to the $\mathbf{LP}(\mathcal{S})$. When some function f_v is not linear, we approximate all those functions with piece-wise linear functions to formulate an MILP. In this paper, we consider the case where a given set F of functions in a network system $\mathcal{S}$ consists of f_v, $v \in V \setminus V_{\text{in}}$ that is a function of a linear combination of variables y_u, $u \in N^-(v)$; i.e., there are constants w_{uv} and w_v such that

$$y_v = f_v\Big(\sum_{u \in N^-(v)} w_{uv} y_u + w_v\Big) \text{ for each vertex } v \in V \setminus V_{\text{in}}. \tag{2}$$

In this case, there exists a weight function $w : V \cup E \to \mathbb{R}$ on the digraph G in $\mathcal{S}$, where we call w_{uv} a *weight* on directed edge $(u, v) \in E$ and w_v a *weight* on vertex $v \in V$. We call such a network system $\mathcal{S} = (G, F)$ a *weight system* and denote it by $\mathcal{S} = (G, w, F)$, where $f_v \in F$ represents a function $f_v(x_v)$ of $x_v = \sum_{u \in N^-(v)} w_{uv} y_u + w_v$.

Artificial Neural Networks In this paper, we define an *artificial neural network* (ANN) $\mathcal{N}$ to be a weight system (G, w, F) such that G is a layered digraph, where a function $f_v \in F$ for a vertex $v \in V \setminus (V_{\text{in}} \cup V_{\text{out}})$ is called an *activation function*. Common activation functions are the logistic sigmoid function, the rectified linear unit function, and the hyperbolic tangent function. For sinks $t \in V_{\text{out}}$ in an ANN $\mathcal{N}$, we may use the identity function or a threshold function f_t. Note that a threshold function is not continuous in many cases.

As already observed, the forward problem to a system $\mathcal{S}$ on a DAG G is computationally easy. In fact, the forward problem on an ANN $\mathcal{N}$ corresponds to a problem of evaluating an input vector α with $y_s = \alpha_s$, $s \in V_{\text{in}}$, called a *feature vector* to guess its output value. Contrary to this, the backward problem on a DAG is not trivial. To overcome this, we formulate the problem of finding an admissible set to a weight system $\mathcal{S}$ as an MILP when functions f_v, $v \in V \setminus V_{\rm in}$ are piece-wise linear.

Piece-wise Linear Functions A function $f : \mathbb{R} \to \mathbb{R}$ is called *piece-wise linear* if there are reals $a_1 < a_2 < \cdots < a_p$, $b_0, b_1, \ldots, b_{p+1}$ and $c_0, c_1, \ldots, c_{p+1}$ such that

$$f(x) = c_j(x - a_j) + b_j, \quad x \in (a_j, a_{j+1}), \quad j = 0, 1, \ldots, p,$$
$$f(a_j) \in \{c_{j-1}(a_j - a_{j-1}) + b_{j-1}, b_j\}, \quad j = 1, 2, \ldots, p,$$

where we regard (a_j, a_{j+1}) for $j = 0$ as $(-\infty, a_1)$ and (a_j, a_{j+1}) for $j = p$ as (a_p, ∞). In the above f, we call each a_j a *break point*, and denote

$$b'_j \triangleq c_{j-1}(a_j - a_{j-1}) + b_{j-1},\ j = 1, 2, \ldots, p,$$

where f is continuous if $b'_j = b_j$ for all j.

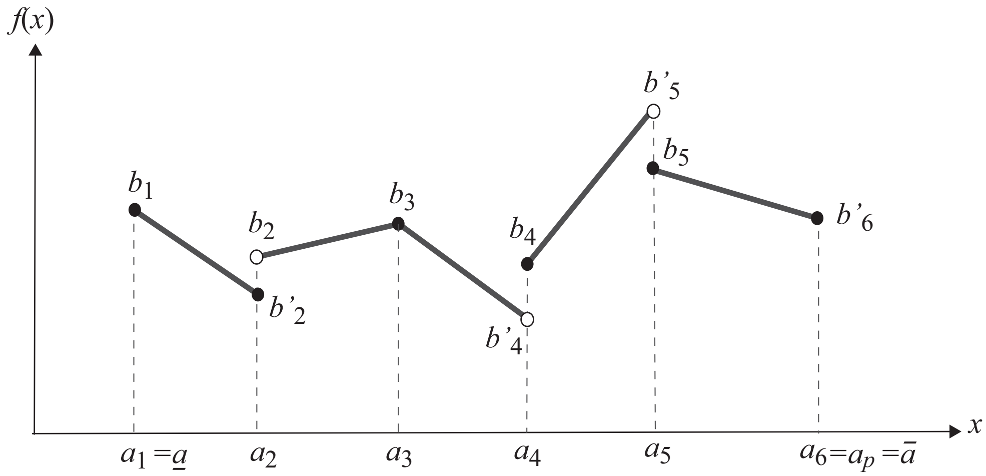

Figure 1: An illustration of a piece-wise linear function f with a domain $[\underline{a}, \overline{a}]$, where $b'_j = c_{j-1}(a_j - a_{j-1}) + b_{j-1},\ j = 2, 3, \ldots, p,\ B = \{b'_2, b_3 = b'_3, b_4, b_5\},\ f(a_2) = b'_2,$ $f(a_4) = b_4$. and $f(a_5) = b_5$.

We denote a piece-wise linear function $f : [\underline{a}, \overline{a}] \to \mathbb{R}$ with a domain $[\underline{a}, \overline{a}]$ by a sequence $((a_1, b_1, c_1), (a_2, b_2, c_3), \ldots, (a_{p-1}, b_{p-1}, c_{p-1}))$ of lines and a set B of boundaries such that

$$
\begin{aligned}
&\underline{a} = a_1 < a_2 < \cdots < a_{p-1} < a_p = \overline{a} \text{ (where } a_p \text{ is defined to be } \overline{a}), \\
&B \subseteq \{b_j, b'_j \mid j = 2, 3, \ldots, p-1\} \text{ with } |B \cap \{b_j, b'_j\}| = 1, j = 2, 3, \ldots, p-1, \\
&f(x) = c_j(x - a_j) + b_j, \quad x \in (a_j, a_{j+1}), \quad j = 1, \ldots, p-1, \\
&f(a_1) = b_1, \\
&f(a_j) = \{b'_j, b_j\} \cap B, \quad j = 2, \ldots, p-1, \\
&f(a_p) = b'_p,
\end{aligned}
$$

where we call a_j with $2 \leq j \leq p-1$ a *break point* of f and a_j with $j \in \{1, p\}$ an *end point* of f. For the above f, we also define

$$
\begin{aligned}
&\overline{b} \triangleq \max\{b_1, b'_2, b_2, b'_3, \ldots, b_{p-1}, b'_p\}, \qquad \underline{b} \triangleq \min\{b_1, b'_2, b_2, b'_3, \ldots, b_{p-1}, b'_p\}, \\
&\rho \triangleq \max_{j=1,2,\ldots,p-1} |c_j|, \qquad \widehat{b} \triangleq \rho \cdot (\overline{a} - \underline{a}) + \overline{b} - \underline{b}.
\end{aligned}
$$

See Fig. 1 for an illustration of piece-wise linear function on a domain $[\underline{a}, \overline{a}]$.

3 Piece-wise Linear Functions with MILP

This section presents how to represent a piece-wise linear function f as an MILP with no objective function.

Let $f : [\underline{a}, \overline{a}] \to \mathbb{R}$ be a piece-wise linear function with a sequence $((a_1, b_1, c_1), (a_2, b_2, c_3), \ldots, (a_{p-1}, b_{p-1}, c_{p-1}))$ of lines and a set B of boundaries. We introduce an instance $\mathbf{MILP}_1(f)$ so that the set of feasible solutions (x, y) to the MILP is equal to the set of pairs x and $y = f(x)$.

MILP$_1(f)$:

constants

$\underline{a} = a_1 < a_2 < \cdots < a_p = \overline{a}$,

$b_1, b_2, \ldots, b_{p-1}, b'_2, b'_3, \ldots, b'_p, \underline{b}, \overline{b}, \widehat{b}$,

$c_1, c_2, \ldots, c_{p-1}$,

$z_1 = 1, z_p = 0$,

real variables

$x \in [\underline{a}, \overline{a}], y \in [\underline{b}, \overline{b}]$,

binary variables

$z_2, z_3, \ldots, z_{p-1} \in \{0, 1\}$,

subject to

$$x - a_i < (\overline{a} - \underline{a})z_i, \qquad i=2,3,\ldots,p-1, b_i \in B \qquad (3)$$

$$x - a_i \geq (\underline{a} - \overline{a})(1-z_i), \qquad i=2,3,\ldots,p-1, b_i \in B \qquad (4)$$

$$x - a_i \leq (\overline{a} - \underline{a})z_i, \qquad i=2,3,\ldots,p-1, b'_i \in B \qquad (5)$$

$$x - a_i > (\underline{a} - \overline{a})(1-z_i), \qquad i=2,3,\ldots,p-1, b'_i \in B \qquad (6)$$

$$y \leq c_i(x-a_i) + b_i + \widehat{b}(1+z_{i+1}-z_i), \qquad i=1,2,\ldots,p-1 \qquad (7)$$

$$y \geq c_i(x-a_i) + b_i - \widehat{b}(1+z_{i+1}-z_i), \qquad i=1,2,\ldots,p-1. \qquad (8)$$

Note that when $p = 2$, **MILP**$_1(f)$ contains no binary variables and is a linear programming problem.

Lemma 1. *Let $f : [\underline{a}, \overline{a}] \to \mathbb{R}$ be a piece-wise linear function with a sequence $((a_1, b_1, c_1), (a_2, b_2, c_3), \ldots, (a_{p-1}, b_{p-1}, c_{p-1}))$ of lines and a set B of boundaries. For any two reals $x_0 \in [\underline{a}, \overline{a}]$ and $y_0 \in [\underline{b}, \overline{b}]$,* **MILP**$_1(f)$ *admits a feasible solution $(x = x_0, y = y_0, z)$ if and only if $y_0 = f(x_0)$ holds.*

Proof. Let $y_0 \in \{f(x) \mid x \in [\underline{a}, \overline{a}]\}$. Choose an arbitrary real $x_0 \in [\underline{a}, \overline{a}]$ such that $y_0 = f(x_0)$. To prove the lemma, it suffices to show the next proposition.

there is a feasible solution $(x = x_0, y, z)$ to **MILP**$_1(f)$ and
every feasible solution $(x = x_0, y, z)$ to **MILP**$_1(f)$ satisfies $y = f(x_0)$. (9)

Since $\underline{a} = a_1 < a_2 < \cdots < a_p = \overline{a}$, we see that any pair x and a_i of reals $x \in [\underline{a}, \overline{a}]$ and a_i, $i \in \{2, 3, \ldots, p-1\}$ satisfies

$$\underline{a} - \overline{a} < x - a_i < \overline{a} - \underline{a}.$$

Let $x_0 \in [\underline{a}, \overline{a}]$, and let i^* denote the index $i \in \{1, 2, \ldots, p-1\}$ such that $a_{i^*} \le x_0 < a_{i^*+1}$. We first claim that (3), (4), (5) and (6) hold for all $i \in \{2, 3, \ldots, p-1\} \setminus \{i^*\}$ if and only if

$$z_1 = z_2 = \cdots = z_{i^*-1} = 1, \quad z_{i^*+1} = \cdots = z_p = 0. \tag{10}$$

For each $i < i^*$, we see that $z_i = 1$ satisfies

$$x_0 - a_i < \overline{a} - \underline{a} = (\overline{a} - \underline{a}) z_i, \quad x_0 - a_i > x_0 - a_{i^*} \ge 0 = (\underline{a} - \overline{a})(1 - z_i),$$

whereas $z_i = 0$ would violate

$$0 \le x_0 - a_{i^*} < x_0 - a_i \le (\overline{a} - \underline{a}) z_i.$$

For each $i > i^*$, $z_i = 0$ satisfies

$$x_0 - a_i > \underline{a} - \overline{a} = (\underline{a} - \overline{a})(1 - z_i), \quad x_0 - a_i \le x_0 - a_{i^*+1} < 0 = (\overline{a} - \underline{a}) z_i,$$

whereas $z_i = 1$ would violate

$$0 > x_0 - a_{i^*+1} \ge x_0 - a_i \ge (\underline{a} - \overline{a})(1 - z_i).$$

Therefore the claim holds. We observe that (10) implies

$$1 + z_{i+1} - z_i = \begin{cases} 0 & \text{if } z_i \ne z_{i+1} \\ 1 & \text{otherwise,} \end{cases}$$

where $z_i \ne z_{i+1}$ if and only if "$i = i^*$ and $z_{i^*} = 1$" or "$i = i^* - 1$ and $z_{i^*} = 0$." For each i such that $1 + z_{i+1} - z_i = 1$, we see that

$$c_i(x_0 - a_i) + b_i + \widehat{b} \cdot 1 \ge -|c_i|(\overline{a} - \underline{a}) + \underline{b} + (\rho \cdot (\overline{a} - \underline{a}) + \overline{b} - \underline{b}) \ge \overline{b},$$

$$c_i(x_0 - a_i) + b_i - \widehat{b} \cdot 1 \le |c_i|(\overline{a} - \underline{a}) + \overline{b} - (\rho \cdot (\overline{a} - \underline{a}) + \overline{b} - \underline{b}) \le \underline{b},$$

implying that (7) and (8) for this i trivially hold since $y \in [\underline{b}, \overline{b}]$. For the j with $1 + z_{j+1} - z_j = 0$, where $j \in \{i^* - 1, i^*\}$,

$$\text{(7) and (8) for this } j \text{ hold if and only if } y = c_j(x_0 - a_j) + b_j. \tag{11}$$

We distinguish three cases.

Case 1. $x_0 > a_{i^*}$: We see that (3)- (6) for $i = i^*$ hold if and only if $z_{i^*} = 1$, since $z_{i^*} = 1$ satisfies

$$x_0 - a_{i^*} < \overline{a} - \underline{a} = (\overline{a} - \underline{a}) z_{i^*}, \quad x_0 - a_{i^*} > 0 = (\underline{a} - \overline{a})(1 - z_{i^*}),$$

whereas $z_{i^*} = 0$ would violate

$$0 < x_0 - a_{i^*} \le (\overline{a} - \underline{a}) z_{i^*}.$$

In this case, (11) implies that $y = c_j(x_0 - a_j) + b_j$ for $j = i^*$ is a unique feasible solution, where $y = f(x_0)$ for $a_{i^*} < x_0 < a_{i^*+1}$.

Case 2. $x_0 = a_{i^*}$ and $b_{i^*} \in B$: We see that (3) and (4) for $i = i^*$ with $b_{i^*} \in B$ hold if and only if $z_{i^*} = 1$, since $z_{i^*} = 1$ satisfies

$$x_0 - a_{i^*} < \overline{a} - \underline{a} = (\overline{a} - \underline{a})z_{i^*}, \quad x_0 - a_{i^*} \geq 0 = (\underline{a} - \overline{a})(1 - z_{i^*}),$$

whereas $z_{i^*} = 0$ would violate

$$x_0 - a_{i^*} = 0 < (\overline{a} - \underline{a})z_{i^*}.$$

In this case, (11) implies $y = c_j(x_0 - a_j) + b_j$ for $j = i^*$ is a unique feasible solution, where $y = c_{i^*}(a_{i^*} - a_{i^*}) + b_{a_{i^*}} = b_{i^*}$, which is $f(a_{i^*}) = f(x_0)$ by $b_{i^*} \in B$.

Case 3. $x_0 = a_{i^*}$ and $b'_{i^*} \in B$: We see that (5) and (6) for $i = i^*$ with $b'_{i^*} \in B$ hold if and only if $z_{i^*} = 0$, since $z_{i^*} = 0$ satisfies

$$x_0 - a_{i^*} = 0 \leq (\overline{a} - \underline{a})z_{i^*}, \quad x_0 - a_{i^*} > \underline{a} - \overline{a} = (\underline{a} - \overline{a})(1 - z_{i^*}),$$

whereas $z_{i^*} = 1$ would violate

$$0 = x_0 - a_{i^*} > (\underline{a} - \overline{a})(1 - z_{i^*}).$$

In this case, (11) implies that $y = c_j(x_0 - a_j) + b_j$ for $j = i^* - 1$ is a unique feasible solution, where $y = c_{i^*-1}(a_{i^*} - a_{i^*-1}) + b_{i^*-1} = b'_{i^*}$, which is $f(a_{i^*}) = f(x_0)$ by $b'_{i^*} \in B$.

From the above, we see that there is a feasible solution $(x = x_0, y = f(x_0), z)$ to $\mathbf{MILP}_1(f)$, and every feasible solution $(x = x_0, y, z)$ satisfies $y = f(x_0)$, proving (9). □

We remark that whether $x > a$ or $x \geq a$ holds may not be tested precisely in a numerical computation with any high precision tolerance. For this, we study the following MILP obtained from $\mathbf{MILP}_2(f)$ by adding equality to the strict inequalities in (3) and (5), where the information on B is no longer used.

MILP$_2(f)$:

constants

$\underline{a} = a_1 < a_2 < \cdots < a_p = \overline{a}$,

$b_1, b_2, \ldots, b_{p-1}, b'_2, b'_3, \ldots, b'_p, \underline{b}, \overline{b}, \widehat{b}$,

$c_1, c_2, \ldots, c_{p-1}$,

$z_1 = 1, z_p = 0$,

real variables

$x \in [\underline{a}, \overline{a}], y \in [\underline{b}, \overline{b}]$,

binary variables

$z_2, z_3, \ldots, z_{p-1} \in \{0, 1\}$,

subject to

$$x - a_i \le (\overline{a} - \underline{a})z_i, \qquad i=2,3,\ldots,p-1 \tag{12}$$

$$x - a_i \ge (\underline{a} - \overline{a})(1-z_i), \qquad i=2,3,\ldots,p-1 \tag{13}$$

$$y \le c_i(x-a_i) + b_i + \widehat{b}(1+z_{i+1}-z_i), \qquad i=1,2,\ldots,p-1 \tag{14}$$

$$y \ge c_i(x-a_i) + b_i - \widehat{b}(1+z_{i+1}-z_i), \qquad i=1,2,\ldots,p-1. \tag{15}$$

Lemma 2. *Let $f : [\underline{a}, \overline{a}] \to \mathbb{R}$ be a piece-wise linear function with a sequence $((a_1, b_1, c_1), (a_2, b_2, c_3), \ldots, (a_{p-1}, b_{p-1}, c_{p-1}))$ of lines and a set B of boundaries. For any two reals $x_0 \in [\underline{a}, \overline{a}]$ and $y_0 \in [\underline{b}, \overline{b}]$,* **MILP**$_2(f)$ *admits a feasible solution $(x = x_0, y = y_0, z)$ if and only if $y_0 = f(x_0)$ or $(x_0 = a_i, y_0 \in \{b_i, b'_i\})$ for some $i \in \{2, 3, \ldots, p-1\}$.*

Proof. Let $x_0 \in [\underline{a}, \overline{a}]$, and let i^* denote the index $i \in \{1, 2, \ldots, p-1\}$ such that $a_{i^*} \le x_0 < a_{i^*+1}$. As in the proof of Lemma 1, we see that (12) and (13) hold for all $i \in \{2, 3, \ldots, p-1\} \setminus \{i^*\}$ if and only if

$$z_1 = z_2 = \cdots = z_{i^*-1} = 1, \quad z_{i^*+1} = \cdots = z_p = 0. \tag{16}$$

This implies

$$1+z_{i+1}-z_i = \begin{cases} 0 & \text{if } z_i \ne z_{i+1} \\ 1 & \text{otherwise,} \end{cases}$$

where $z_i \ne z_{i+1}$ if and only if "$i = i^*$ and $z_{i^*} = 1$" or "$i = i^* - 1$ and $z_{i^*} = 0$," and we see that (14) and (15) for any i with $z_i \ne z_{i+1} = 1$ trivially hold. For the j with $1+z_{j+1}-z_j = 0$, where $j \in \{i^*-1, i^*\}$, (14) and (15) hold if and only if $y = c_j(x_0-a_j)+b_j$.

We distinguish two cases.

Case 1. $x_0 > a_{i^*}$: As in Case 1 of proof of Lemma 1, we see that when $x_0 > a_{i^*}$, (12) and (13) with $i = i^*$ hold if and only if $z_{i^*} = 1$, where $(x_0, y = f(x_0), z)$ is a unique feasible solution to $\mathbf{MILP}_2(f)$.
Case 2. $x_0 = a_{i^*}$: In this case, (12) and (13) with $i = i^*$ hold for any of $z_{i^*} = 1$ and $z_{i^*} = 0$. When $z_{i^*} = 1$, $y = c_j(x_0 - a_j) + b_j = c_{i^*}(a_{i^*} - a_{i^*}) + b_{i^*} = b_{i^*}$ holds and $(x_0 = a_{i^*}, y = b_{i^*}, z)$ is a feasible solution to $\mathbf{MILP}_2(f)$. When $z_{i^*} = 0$, $y = c_j(x_0 - a_j) + b_j = c_{i^*-1}(a_{i^*} - a_{i^*-1}) + b_{i^*-1} = b'_{i^*}$ holds and $(x_0 = a_{i^*}, y = b'_{i^*}, z)$ is a feasible solution to $\mathbf{MILP}_2(f)$. We see that there is no feasible solution to $\mathbf{MILP}_2(f)$ other than the above two solutions. □

When a given piece-wise linear function f is not continuous, i.e., $b_i \neq b'_i$ for some i with $2 \leq i \leq p-1$, the set of feasible solutions to $\mathbf{MILP}_2(f)$ may contain a pair (x, y) such that $y \neq f(x)$ for a special case of $x = a_i$. However, when f is a a continuous piece-wise linear function, $\mathbf{MILP}_2(f)$ completely represents f in the sense that the set of feasible solutions is equal to the pairs of x and $y = f(x)$.

4 Representing Systems with MILP

In this section, we show how to formulate a given weight system $\mathcal{S}$ as an MILP so that the set of admissible solutions to $\mathcal{S}$ is preserved as the set of feasible solution to the resulting MILP. Let $\mathcal{S} = (G, w, F)$ be a weight system with a simple digraph $G = (V, E)$, a weight function $w : V \cup E \to \mathbb{R}$ and a set F of piece-wise linear functions, where G is not necessarily acyclic and possibly $V_{\text{in}} = \emptyset$ or $V_{\text{out}} = \emptyset$. For each $v \in V \setminus V_{\text{in}}$, let $f_v : [\underline{a}_v, \overline{a}_v] \to [\underline{b}_v, \overline{b}_v]$ be a piece-wise linear function with a sequence $((a_{v,1}, b_{v,1}, c_{v,1}),$ $(a_{v,2}, b_{v,2}, c_{v,3}), \ldots, (a_{v,p-1}, b_{v,p-1}, c_{v,p_v-1}))$ of lines and a set B_v of boundaries, where

$$b'_{v,j} \triangleq c_{v,j-1}(a_{v,j} - a_{v,j-1}) + b_{v,j-1},\ j = 1, 2, \ldots, p_v - 1.$$

Let n_{b} denote the total number of break points over all functions f_v, $v \in V$; i.e., $n_{\text{b}} = \sum\{p_v - 2 \mid v \in V \setminus V_{\text{in}}\}$. Define

$$\rho_v \triangleq \max\{|c_{v,i}| \mid i = 1, 2, \ldots, p_v - 1\}, \quad \widehat{b}_v \triangleq \rho_v \cdot (\overline{a}_v - \underline{a}_v) + \overline{b}_v - \underline{b}_v.$$

To formulate $\mathcal{S}$ as an MILP so that $\mathcal{A}(\mathcal{S})$ is preserved as the set of feasible solutions to the MILP, we prepare y_v, $v \in V$ as *main variables*, which directly correspond to values on vertices in the weight system $\mathcal{S}$, and introduce a vector of *auxiliary variables* z to represent each function f_v, $v \in V \setminus V_{\text{in}}$ as an $\mathbf{MILP}(f_v)$ in the previous section. After representing each function f_v as $\mathbf{MILP}_1(f_v)$, we next introduce *auxiliary variables* x_v, $v \in V \setminus V_{\text{in}}$ to prepare an input $x_v = \sum_{u \in N^-(v)} w_{uv} y_u + w_v$ for each function f_v. The resulting MILP, $\mathbf{MILP}^*_1(\mathcal{S})$, is a collection of these variables and linear constraints from $\mathbf{MILP}_1(f_v)$, $v \in V \setminus V_{\text{in}}$.

MILP$^*_1(\mathcal{S})$

$$
\begin{array}{ll}
\underline{a}_v = a_{v,1} < a_{v,2} < \cdots < a_{v,p_v} = \overline{a}_v, & v \in V \setminus V_{\mathrm{in}} \\
b_{v,1}, b_{v,2}, \ldots, b_{v,p_v-1}, b'_{v,2}, b'_{v,3}, \ldots, b'_{v,p_v}, & v \in V \\
\underline{b}_v, \overline{b}_v, \widehat{b}_v, & v \in V \setminus V_{\mathrm{in}} \\
c_{v,1}, c_{v,2}, \ldots, c_{v,p_v-1}, & v \in V \setminus V_{\mathrm{in}} \\
z_{v,1} = 1, z_{v,p_v} = 0, & v \in V \setminus V_{\mathrm{in}}
\end{array}
$$

real variables

$$
\begin{array}{ll}
y_v \in [\underline{b}_v, \overline{b}_v], & v \in V \\
x_v \in [\underline{a}_v, \overline{a}_v], & v \in V \setminus V_{\mathrm{in}}
\end{array}
$$

binary variables

$$
\begin{array}{ll}
z_{v,2}, z_{v,3}, \ldots, z_{v,p_v-1} \in \{0,1\}, & v \in V \setminus V_{\mathrm{in}}
\end{array}
$$

subject to

$$
\begin{array}{ll}
x_v = \displaystyle\sum_{u \in N^-(v)} w_{uv} y_u + w_v, & v \in V \setminus V_{\mathrm{in}} \\
x_v - a_{v,i} < (\overline{a}_v - \underline{a}_v) z_{v,i}, & v \in V \setminus V_{\mathrm{in}}, i = 2, \ldots, p_v - 1, b_{v,i} \in B_v \\
x_v - a_{v,i} \geq (\underline{a}_v - \overline{a}_v)(1 - z_{v,i}), & v \in V \setminus V_{\mathrm{in}}, i = 2, \ldots, p_v - 1, b_{v,i} \in B_v \\
x_v - a_{v,i} \leq (\overline{a}_v - \underline{a}_v) z_{v,i}, & v \in V \setminus V_{\mathrm{in}}, i = 2, \ldots, p_v - 1, b'_{v,i} \in B_v \\
x_v - a_{v,i} > (\underline{a}_v - \overline{a}_v)(1 - z_{v,i}), & v \in V \setminus V_{\mathrm{in}}, i = 2, \ldots, p_v - 1, b'_{v,i} \in B_v \\
y_v \leq c_{v,i}(x_v - a_{v,i}) + b_{v,i} + \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), & v \in V \setminus V_{\mathrm{in}}, i = 1, 2, \ldots, p_v - 1 \\
y_v \geq c_{v,i}(x_v - a_{v,i}) + b_{v,i} - \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), & v \in V \setminus V_{\mathrm{in}}, i = 1, 2, \ldots, p_v - 1.
\end{array}
$$

Let $\mathrm{b}^{(V)}$ denote the domain of a vector y of main variables y_v, $v \in V$; i.e., $\mathrm{b}^{(V)} = [\underline{b}_1, \overline{b}_1] \times [\underline{b}_2, \overline{b}_2] \times \cdots \times [\underline{b}_n, \overline{b}_n]$ for $y = (y_1, y_2, \ldots, y_n)$.

For instance $I = \mathbf{MILP}^*_1(\mathcal{S})$, remember that $\mathcal{F}(y; I) \subseteq \mathrm{b}^{(V)}$ is the set of vectors of reals on y_v, $v \in V$, i.e., $y' \in \mathcal{F}(y; I)$ means that $\mathbf{MILP}^*_1(\mathcal{S})$ admits a feasible solution such that $y_v = y'_v$, $v \in V$. Then $\mathcal{A}(\mathcal{S})$ is equal to $\mathcal{F}(y; I)$.

Theorem 3. *Let $\mathcal{S} = (G, w, F)$ be a weight system with a set F of piece-wise linear functions, and $I = \mathbf{MILP}^*_1(\mathcal{S})$. Then $\mathcal{A}(\mathcal{S}) = \mathcal{F}(y; I)$. For any subset $Y \subseteq \mathrm{b}^{(V)}$, $\mathcal{A}(\mathcal{S}) \cap Y = \mathcal{F}(y; I) \cap Y$.*

Proof. We see that $\mathcal{A}(\mathcal{S}) = \mathcal{F}(y; I)$ immediately from Lemma 1 applied to each $\mathbf{MILP}_1(f_v)$, $v \in V \setminus V_{\mathrm{in}}$. Also $\mathcal{A}(\mathcal{S}) \cap Y = \mathcal{F}(y; I) \cap Y$ is immediate from $\mathcal{A}(\mathcal{S}) = \mathcal{F}(y; I)$. □

We observe that $\mathbf{MILP}_1^*(\mathcal{S})$ contains $O(|V|+n_{\rm b})$ variables and constraints.

When we impose an additional constraint on $\mathcal{A}(\mathcal{S})$ to obtain $\mathcal{A}(\mathcal{S})\cap Y$ for a subset $Y\subseteq \mathrm{b}^{(V)}$, it also holds that $\mathcal{A}(\mathcal{S})\cap Y=\mathcal{F}(y;I)\cap Y$. In particular, if Y is described as a set of linear constraints and integer constraints, then $\mathcal{F}(y;I)\cap Y$ can be the set $\mathcal{F}(y;I')$ of feasible solutions to a modified MILP I'.

Now we introduce another MILP by avoiding constraints with strict inequalities in $\mathbf{MILP}_1^*(\mathcal{S})$.

$\mathbf{MILP}_2^*(\mathcal{S})$

constants

$$\underline{a}_v=a_{v,1}<a_{v,2}<\cdots<a_{v,p_v}=\overline{a}_v, \qquad v\in V\setminus V_{\rm in}$$
$$b_{v,1},b_{v,2},\ldots,b_{v,p_v-1},b'_{v,2},b'_{v,3},\ldots,b'_{v,p_v}, \qquad v\in V$$
$$\underline{b}_v,\overline{b}_v,\widehat{b}_v, \qquad v\in V\setminus V_{\rm in}$$
$$c_{v,1},c_{v,2},\ldots,c_{v,p_v-1}, \qquad v\in V\setminus V_{\rm in}$$
$$z_{v,1}=1,z_{v,p_v}=0, \qquad v\in V\setminus V_{\rm in}$$

real variables

$$y_v\in[\underline{b}_v,\overline{b}_v], \qquad v\in V$$
$$x_v\in[\underline{a}_v,\overline{a}_v], \qquad v\in V\setminus V_{\rm in}$$

binary variables

$$z_{v,2},z_{v,3},\ldots,z_{v,p_v-1}\in\{0,1\}, \qquad v\in V\setminus V_{\rm in}$$

subject to

$$x_v=\sum_{u\in N^-(v)}w_{uv}y_u+w_v, \qquad v\in V\setminus V_{\rm in}$$
$$x_v-a_{v,i}\le(\overline{a}_v-\underline{a}_v)z_{v,i}, \qquad v\in V\setminus V_{\rm in}, i=2,3,\ldots,p_v-1$$
$$x_v-a_{v,i}\ge(\underline{a}_v-\overline{a}_v)(1-z_{v,i}), \qquad v\in V\setminus V_{\rm in}, i=2,3,\ldots,p_v-1$$
$$y_v\le c_{v,i}(x_v-a_{v,i})+b_{v,i}+\widehat{b}_v(1+z_{v,i+1}-z_{v,i}), \qquad v\in V\setminus V_{\rm in}, i=1,2,\ldots,p_v-1$$
$$y_v\ge c_{v,i}(x_v-a_{v,i})+b_{v,i}-\widehat{b}_v(1+z_{v,i+1}-z_{v,i}), \qquad v\in V\setminus V_{\rm in}, i=1,2,\ldots,p_v-1.$$

Observe that $\mathbf{MILP}_2^*(\mathcal{S})$ also contains $O(|V|+n_{\rm b})$ variables and constraints.

Theorem 4. *Let $\mathcal{S}=(G,w,F)$ be a weight system with a set F of continuous piece-wise linear functions and $I=\mathbf{MILP}_2^*(\mathcal{S})$. Then $\mathcal{A}(\mathcal{S})=\mathcal{F}(y;I)$. For any subset $Y\subseteq\mathrm{b}^{(V)}$, $\mathcal{A}(\mathcal{S})\cap Y=\mathcal{F}(y;I)\cap Y$.*

Proof. For each continuous piece-wise linear function f_v, it holds $b_{v,i} = b'_{v,i}$ $(2 \le i \le p_v - 1)$. Hence by Lemma 2, $\mathbf{MILP}_2(f_v)$ represents f_v completely as in Lemma 1 with $\mathbf{MILP}_1(f_v)$. Therefore the theorem holds, as proved in Theorem 4. $\square$

We now show how a subset Y can be chosen so that $\mathcal{F}(y; I) \cap Y$ is still given as the set $\mathcal{F}(y; I')$ of a modified MILP I'. For example, choose a subset $X \subseteq \{y_v \mid v \in V\}$ of variables by introducing a new auxiliary variable $z_X \in \mathbb{R}$ (or $z_X \in \mathbb{Z}$) and new constants $\underline{d}_X$, $\overline{d}_X$, d_x, $x \in X$ such that

$$z_X = \textstyle\sum_{x \in X} d_x x; \quad \underline{d}_X \le z_X \le \overline{d}_X.$$

Then set Y to be the set of vectors $y \in \mathrm{b}^{(V)}$ such that there is a value of z_X satisfying the above new constraints. We easily observe that $\mathcal{F}(y; I) \cap Y = \mathcal{F}(y; I')$ for the MILP I' obtained from I' by adding the above auxiliary variable z_X and constraints.

We can set a subset $Y \subseteq \mathrm{b}^{(V)}$ by introducing the above type of constraints on a sequence of variable subsets $X_1, X_2, \ldots, X_q$, where some X_i may contain a variable z_{X_j} with $i < j$.

For the forward and backward problems to a weight system $\mathcal{S}$, a subset Y is specified as follows. Let I be an MILP such that $\mathcal{A}(\mathcal{S}) = \mathcal{F}(y; I)$. For the forward problem $(\mathcal{S}, V_{\mathrm{in}}, \alpha)$, we set $Y = \{y \in \mathrm{b}^{(V)} \mid y_s = \alpha_s, s \in V_{\mathrm{in}}\}$. In this case, we add I $|V_{\mathrm{in}}|$ new constraints

$$y_s = \alpha_s,\ s \in V_{\mathrm{in}}$$

so that $\mathcal{F}(y; I) \cap Y = \mathcal{F}(y; I')$ for the resulting MILP I'.

Analogously for the backward problem $(\mathcal{S}, V_{\mathrm{out}}, \beta)$, we add I $|V_{\mathrm{out}}|$ new constraints

$$y_t = \beta_t,\ t \in V_{\mathrm{out}}$$

so that $\mathcal{F}(y; I) \cap Y = \mathcal{F}(y; I')$ for the resulting MILP I'. When we want to find an admissible set to a weight system $\mathcal{S}$ such that

$$\underline{\beta}_t \le y_t \le \overline{\beta}_t,\ t \in V_{\mathrm{out}}$$

for some constants $\underline{\beta}_t, \overline{\beta}_t \in \mathbb{R}$, $t \in V_{\mathrm{out}}$, we add to I these constraints so that $\mathcal{F}(y; I) \cap Y = \mathcal{F}(y; I')$ for the resulting MILP I'.

We consider the case where a weight system $\mathcal{S}$ satisfies $|V_{\mathrm{out}}| = 1$ and the function $f_t : [\underline{a}, \overline{a}] \to [0, 1]$ of the sink in V_{out} is a threshold function

$$f_t(x) = \begin{cases} 0 & \text{if } \underline{a} \le x < 0 \\ 1 & \text{if } 0 \le x \le \overline{a}. \end{cases}$$

Since f_t is not continuous, $I = \mathbf{MILP}_2^*(\mathcal{S})$ may not preserve the set $\mathcal{A}(\mathcal{S})$ of admissible sets. Any admissible set $y \in \mathcal{A}(\mathcal{S})$ with $y_t = 1$ satisfies $y_t = 1 = f_t(\sum_{u \in N^-(t)} w_{ut} y_u + w_t)$.

When we aim to find an admissible set $y \in \mathcal{A}(\mathcal{S})$ such that $y_t = 1$ and the input $\sum_{u \in N^-(t)} w_{ut} y_u + w_t$ to f_t is maximized, we add to I the following constraint and objective function:

$$y_t \geq 1,$$

$$\text{objective: maximize } x_t,$$

where x_t is an auxiliary variable already introduced in I satisfying $x_t = \sum_{u \in N^-(t)} w_{ut} y_u + w_t$.

5 Representing Inverse ANN by MILP

This section introduces examples of MILPs for the inverse problem of ANN with some activation functions.

5.1 Initialization

Assume that we are given a weight system $\mathcal{S} = (G, w, F)$ with a DAG G and a set F of piece-wise linear functions and ranges $[\underline{b}_s, \overline{b}_s]$ for sources $s \in V_{\text{in}}$, where the end points of each function $f_v \in F$ may not be specified. Before we formulate the backward problem on the weight system, we compute domains $[\underline{a}_v, \overline{a}_v]$ and ranges $[\underline{b}_v, \overline{b}_v]$ for other main variables y_v, $v \in V \setminus V_{\text{in}}$ as follows.
For each vertex $v \in V \setminus V_{\text{in}}$ such that the domains and ranges on variables y_u with $u \in N^-(v)$ have been determined, set the domain and range on variable y_v so that

$$\begin{aligned}
\overline{a}_v &:= \max\{\sum_{u \in N^-(v)} w_{uv} y_u \mid \underline{b}_u \leq y_u \leq \overline{b}_u, u \in N^-(v)\} + w_v \\
&= \sum\{w_{uv}\overline{b}_u \mid w_{uv} > 0, u \in N^-(v)\} + \sum\{w_{uv}\underline{b}_u \mid w_{uv} < 0, u \in N^-(v)\} + w_v; \\
\underline{a}_v &:= \min\{\sum_{u \in N^-(v)} w_{uv} y_u \mid \underline{b}_u \leq y_u \leq \overline{b}_u, u \in N^-(v)\} + w_v \\
&= \sum\{w_{uv}\overline{b}_u \mid w_{uv} < 0, u \in N^-(v)\} + \sum\{w_{uv}\underline{b}_u \mid w_{uv} > 0, u \in N^-(v)\} + w_v; \\
\overline{b}_v &:= \max\{f_v(x) \mid \underline{a}_v \leq x \leq \overline{a}_v\}; \\
\underline{b}_v &:= \min\{f_v(x) \mid \underline{a}_v \leq x \leq \overline{a}_v\}.
\end{aligned}$$

Then for each vertex $v \in V \setminus V_{\text{in}}$, we set $\underline{a}_v$ and $\overline{a}_v$ to be the end points and of f_v so that f_v is given as $((a_{v,1}, b_{v,1}, c_{v,1}), (a_{v,2}, b_{v,2}, c_{v,3}), \ldots, (a_{v,p-1}, b_{v,p-1}, c_{v,p_v-1}))$, where $a_{v,1} = \underline{a}_v$ and $a_{v,p-1} < \overline{a}_v$ $(= a_{v,p})$, and also set ρ_v and $\widehat{b}_v$ so that

$$\rho_v := \max\{|c_{v,i}| \mid i = 1, 2, \ldots, p_v - 1\};$$

$$\widehat{b}_v := \rho_v \cdot (\overline{a}_v - \underline{a}_v) + \overline{b}_v - \underline{b}_v.$$

5.2 Case of ReLU Function

We here consider an ANN with the ReLU function. Let $\mathcal{N} = (G, w, F)$ be an ANN with $F = \{f_v : \mathbb{R} \to \mathbb{R} \mid v \in V \setminus V_{\mathrm{in}}\}$ such that for each vertex $v \in V \setminus (V_{\mathrm{in}} \cup V_{\mathrm{out}})$

$$f_v(x) = \max\{0, x\} = \begin{cases} 0 & \text{if } x \leq 0, \\ x & \text{if } 0 \leq x. \end{cases}$$

Let $f_v = ((a_{v,1} = \underline{a}, b_{v,1} = 0, c_{v,1} = 0), (a_{v,2} = 0, b_{v,2} = 0, c_{v,2} = 1))$ denote a piece-wise linear function with a domain $[\underline{a}_v, \overline{a}_v]$ ($\underline{a}_v < 0 < \overline{a}_v$), where $\underline{b}_v = 0$, $\overline{b}_v = b'_3 = \overline{a}_v$, $b'_{v,2} = b_{v,2} = 0$ and $\widehat{b}_v = (\overline{a}_v - \underline{a}_v) + \overline{b}_v - \underline{b}_v = 2\overline{a}_v - \underline{a}_v$. See Fig. 2 for an illustration of the ReLU function on a domain $[\underline{a}_v, \overline{a}_v]$.

We also assume that $|V_{\mathrm{out}}| = 1$ and $f_t(x) = x$ for the sink $t \in V_{\mathrm{out}}$.

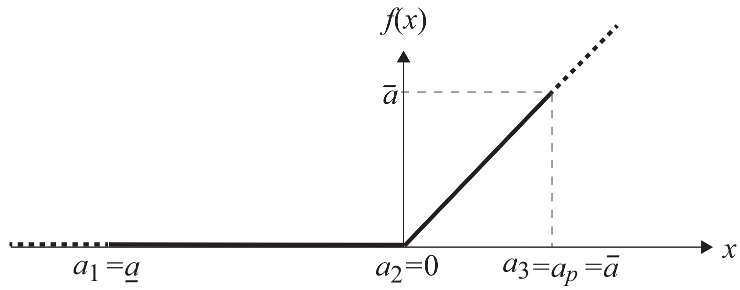

Figure 2: An illustration of the ReLU function f with a domain $[\underline{a}, \overline{a}]$ ($\underline{a} < 0 < \overline{a}$).

MILP$_2^*(\mathcal{S}, \beta_t)$

constants

$$\underline{a}_v = a_{v,1} < a_{v,2} = 0, a_{v,3} = \overline{a}_v, \qquad v \in V \setminus V_{\text{in}}$$

$$\underline{b}_v = b_{v,1} = b_{v,2} = 0, b'_{v,3} = \overline{b}_v = \overline{a}_v, \qquad v \in V \setminus V_{\text{in}}$$

$$\widehat{b}_v = 2\overline{a}_v - \underline{a}_v, \qquad v \in V \setminus V_{\text{in}}$$

$$c_{v,1} = 0, c_{v,2} = 1, \qquad v \in V \setminus V_{\text{in}}$$

$$z_{v,1} = 1, z_{v,3} = 0, \qquad v \in V \setminus V_{\text{in}}$$

$$\beta_t \in [\underline{b}_t, \overline{b}_t] = [0, \overline{a}_t],$$

real variables

$$y_v \in [0, \overline{b}_v], \qquad v \in V$$

$$x_v \in [\underline{a}_v, \overline{a}_v], \qquad v \in V \setminus V_{\text{in}}$$

binary variables

$$z_{v,2} \in \{0, 1\}, \qquad v \in V \setminus V_{\text{in}}$$

subject to

$$x_v = \sum_{u \in N^-(v)} w_{uv} y_u + w_v, \qquad v \in V \setminus V_{\text{in}}$$

$$x_v - a_{v,2} \leq (\overline{a}_v - \underline{a}_v) z_{v,2}, \qquad v \in V \setminus V_{\text{in}}$$

$$x_v - a_{v,2} \geq (\underline{a}_v - \overline{a}_v)(1 - z_{v,2}), \qquad v \in V \setminus V_{\text{in}}$$

$$y_v \leq c_{v,i}(x_v - a_{v,i}) + b_{v,i} + \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), \qquad v \in V \setminus V_{\text{in}}, i = 1, 2$$

$$y_v \geq c_{v,i}(x_v - a_{v,i}) + b_{v,i} - \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), \qquad v \in V \setminus V_{\text{in}}, i = 1, 2$$

$$y_t = \beta_t.$$

5.3 Case of Approximating Sigmoid Function

The logistic sigmoid function $f(x) = 1/(1 + e^{-x})$ is not piece-wise linear. We here approximate this with a continuous piece-wise linear with two break points. Let $\mathcal{N} = (G, w, F)$ be an ANN with $F = \{f_v : \mathbb{R} \to \mathbb{R} \mid v \in V \setminus V_{\rm in}\}$ such that for each vertex $v \in V \setminus (V_{\rm in} \cup V_{\rm out})$

$$f_v(x) = \begin{cases} 0 & \text{if } x \leq -3, \\ (x+3)/6 & \text{if } -3 \leq x \leq 3, \\ 1 & \text{if } 3 \leq x. \end{cases} \tag{17}$$

Let $f_v = ((a_{v,1} = \underline{a}, b_{v,1} = 0, c_{v,1} = 0), (a_{v,2} = -3, b_{v,2} = 0, c_{v,2} = 1/6), (a_{v,3} = 3, b_{v,3} = 1, c_{v,3} = 0))$ denote a piece-wise linear function with a domain $[\underline{a}, \overline{a}]$ ($\underline{a} < -3$ and $3 < \overline{a}$), where $\underline{b}_v = 0$, $\overline{b}_v = 1$, $b'_{v,2} = b_{v,2} = 0$. $b'_{v,3} = b_{v,3} = b'_{v,4} = 1$, and $\widehat{b}_v = (1/6)(\overline{a}_v - \underline{a}_v) + \overline{b}_v - \underline{b}_v = (1/6)(\overline{a}_v - \underline{a}_v) + 1$. See Fig. 3 for an illustration of the above function.

Assume that $|V_{\rm out}| = 1$ and $f_t(x) = x$ for the sink $t \in V_{\rm out}$.

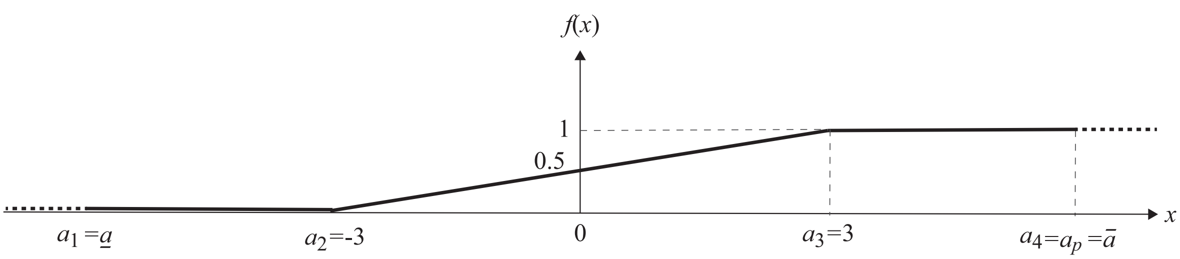

Figure 3: An illustration of a piece-wise linear function with two break points $(-3, 0)$ and $(3, 1)$ in a domain $[\underline{a}, \overline{a}]$ ($\underline{a} < -3$ and $3 < \overline{a}$).

MILP$_2^*(\mathcal{S}, \beta_t)$

constants

$$\underline{a}_v = a_{v,1} < a_{v,2} = -3, a_{v,3} = 3 < a_{v,4} = \overline{a}_v, \qquad v \in V \setminus V_{\rm in}$$

$$\underline{b}_v = b_{v,1} = b_{v,2} = b'_{v,2} = 0, b_{v,3} = b'_{v,4} = \overline{b}_v = 1, \qquad v \in V \setminus V_{\rm in}$$

$$\widehat{b}_v = (1/6)(\overline{a}_v - \underline{a}_v) + 1, \qquad v \in V \setminus V_{\rm in}$$

$$c_{v,1} = 0, c_{v,2} = 1/6, c_{v,3} = 0, \qquad v \in V \setminus V_{\rm in}$$

$$z_{v,1} = 1, z_{v,4} = 0, \qquad v \in V \setminus V_{\rm in}$$

$$\beta_t \in [0, 1],$$

real variables

$$y_v \in [0, 1], \qquad v \in V$$

$$x_v \in [\underline{a}_v, \overline{a}_v], \qquad v \in V \setminus V_{\rm in}$$

binary variables

$$z_{v,2}, z_{v,3}, \in \{0, 1\}, \qquad v \in V \setminus V_{\rm in}$$

subject to

$$x_v = \sum_{u \in N^-(v)} w_{uv} y_u + w_v, \qquad v \in V \setminus V_{\rm in}$$

$$x_v - a_{v,i} \le (\overline{a}_v - \underline{a}_v) z_{v,i}, \qquad v \in V \setminus V_{\rm in}, i{=}2,3$$

$$x_v - a_{v,i} \ge (\underline{a}_v - \overline{a}_v)(1 - z_{v,i}), \qquad v \in V \setminus V_{\rm in}, i{=}2,3$$

$$y_v \le c_{v,i}(x_v - a_{v,i}) + b_{v,i} + \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), \qquad v \in V \setminus V_{\rm in}, i{=}1,2,3$$

$$y_v \ge c_{v,i}(x_v - a_{v,i}) + b_{v,i} - \widehat{b}_v(1 + z_{v,i+1} - z_{v,i}), \qquad v \in V \setminus V_{\rm in}, i{=}1,2,3$$

$$\beta_t = y_t.$$

6 Concluding Remarks

In this paper, we observed that a piece-wise linear function f can be represented in an MILP so that the pairs of reals x and $f(x)$ are preserved as the feasible solutions to the MILP except for break points in discontinuous functions. Although MILP is an NP-hard problem, there have been developed several practically efficient solvers for LP and MILP. In our preliminary experiments on an ANN $\mathcal{N}$ with an input layer of 200 nodes, one hidden layer of 200 nodes and an output layer of a single node constructed based on the feature vectors proposed in [5], it took less than 2 seconds to solve the inverse problem on $\mathcal{N}$ by using CPLEX (ILOG CPLEX version 12.8) [6] on a PC with

Intel Core i5 1.8 GHz CPU and 8GB RAM running under the Mac OS operating system version 10.13.6. This result suggests that our proposed formulation is practically useful. One of important future works is to modify and apply this formulation for design of novel chemical structures.

References

[1] T. Akutsu, D. Fukagawa, J. Jansson and K. Sadakane. Inferring a graph from path frequency, Discrete Applied Mathematics, vol. 160, 10-11, 2012, 1416-1428.

[2] H. Fujiwara, J. Wang, L. Zhao, H. Nagamochi and T. Akutsu. Enumerating treelike chemical graphs with given path frequency, Journal of Chemical Information and Modeling, vol. 48, 7, 2008, 1345-1357.

[3] M. R. Gary and D. S. Johnson, Computers and Intractability, a Guide to the Theory of NP-completeness, Freeman, San Francisco 1978.

[4] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hermández-Lobato, B. Sanchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules, ACS Central Science, vol. 4, 2018, 268-276.

[5] P.-A. Grenier, L. Brun and D. Villemin, Chemoinformatics and stereoisomerism: A stereo graph kernel together with three new extensions, Pattern Recognition Letters, vol. 87, 1, 2017, 222–230

[6] IBM ILOG CPLEX Optimization Studio 12.8, `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcplex.pdf`

[7] H. Ikebata, K. Hongo, T. Isomura and R. Maezono. Bayesian molecular design with a chemical language model, Journal of Computer Aided Molecular Design, vol. 31, 2017, 379-391.

[8] A. Kerber, R. Laue, T. Gruner and M. Meringer. MOLGEN 4.0, Match Communications in Mathematical and in Computer Chemistry, vol. 37, 1998, 205-208.

[9] L. G. Khachiyan, A polynomial algorithm in linear programming [in Russian], Doklady Akademi Nuak SSSR 244, 1979, 1093-1096. English translation: Soviet Mathematics Doklady 20, 1979, 191-194.

[10] M. J. Kusner, B. Paige, J. M. Hernández-Lobato. Grammar variational autoencoder, In: Proc. 34th International Conference on Machine Learning (ICML 2017), 2017, 1945-1954.

[11] J. Li, H. Nagamochi and T. Akutsu. Enumerating substituted benzene isomers of tree-like chemical graphs, IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 15, 2, 2018, 633-646,

[12] P. Liu, Y. Bao, M. Hayashida and T. Akutsu. Finding pre-images for neural networks: an integer linear programming approach, Poster abstract, 17th International Workshop on Bioinformatics and Systems Biology, 2017.

[13] T. Miyao, H. Kaneko and K. Funatsu. Inverse QSPR/QSAR analysis for chemical structure generation (from y to x), Journal of Chemical Information and Modeling, vol. 56, 2, 2016, 286-299.

[14] H. Nagamochi. A detachment algorithm foriInferring a graph from path frequency, Algorithmica, vol. 53, 2, 2009, 207-224.

[15] G. L. Nemhauser and L. A. Wolsey, Integer and Combinatorial Optimization, John Wiley and Sons, New York, 1988.

[16] J-L. Reymond. The chemical space project, Accounts of Chemical Research, vol, 48, 2015, 722-730.

[17] C. Rupakheti, A. Virshup, W. Yang and D. N. Beratan. Strategy to discover diverse optimal molecules in the small molecule universe, Journal of Chemical Information and Modeling, vol. 55, 2, 2015, 529-537.

[18] A. Schrijver, Theory of Linear and Integer Programming, John Wiley and Sons, Chichester, 1986.

[19] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, Volumes A, B, C, Springer, Berlin, 2003.

[20] M. H. S. Segler, T. Kogej, C. Tyrchan and M. P. Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks, ACS Central Science, vol. 4, 2018, 120-131.

[21] M. I. Skvortsova, I. I. Baskin, O. L. Slovokhotova, V. A. Palyulin and N. S. Zefirov. Inverse problem in QSAR/QSPR studies for the case of topological indices characterizing molecular shape (Kier indices), Journal of Chemical Information and Computer Science, vol. 33, 1993, 630-640.

[22] X. Yang, J. Zhang, K. Yoshizoe, K. Terayama and K. Tsuda. ChemTS: an efficient python library for de novo molecular generation, Journal of Science and Technology of Advanced Materials, vol. 18, 1, 2017, 972-976.