Extended Formulations via Decision Diagrams

A Preprint

Yuta Kurokawa
Kyushu University
ie19004r@outlook.jp

Ryotaro Mitsuboshi
Kyushu University/RIKEN AIP
ryotaro.mitsuboshi@inf.kyushu-u.ac.jp

Haruki Hamasaki
Kyushu University
hamasaki.haruki.897@s.kyushu-u.ac.jp

Kohei Hatano
Kyushu University/RIKEN AIP
hatano@inf.kyushu-u.ac.jp

Eiji Takimoto
Kyushu University
eiji@inf.kyushu-u.ac.jp

Holakou Rahmanian
Amazon
holakou@amazon.com

September 7, 2023

Abstract

We propose a general algorithm of constructing an extended formulation for any given set of linear constraints with integer coefficients. Our algorithm consists of two phases: first construct a decision diagram \((V, E)\) that somehow represents a given \(m \times n\) constraint matrix, and then build an equivalent set of \(|E|\) linear constraints over \(n + |V|\) variables. That is, the size of the resultant extended formulation depends not explicitly on the number \(m\) of the original constraints, but on its decision diagram representation. Therefore, we may significantly reduce the computation time and space for optimization problems with integer constraint matrices by solving them under the extended formulations, especially when we obtain concise decision diagram representations for the matrices. Then, we consider the 1-norm regularized soft margin optimization over the binary instance space \(\{0, 1\}^n\), a standard optimization problem in the machine learning literature. This problem is motivating since the naive application of our extended formulation produces decision diagrams of size \(\Omega(m)\). For this problem, we give a modified formulation which works in practice and efficient algorithms whose time complexity depends on the size of the diagrams. This problem can be formulated as a linear programming problem with \(m\) constraints with \(\{-1, 0, 1\}\)-valued coefficients over \(n\) variables, where \(m\) is the size of the given sample. We demonstrate the effectiveness of our extended formulations for mixed integer programming and the 1-norm regularized soft margin optimization tasks over synthetic and real datasets.

Keywords
Extend formulation · Decision diagrams · Mixed integer programs · Soft margin optimization

1 Introduction

Large-scale optimization tasks appear in many areas such as machine learning, operations research, and engineering. Time/memory-efficient optimization techniques are more in demand than ever. Various approaches have been proposed to efficiently solve optimization problems over huge data, e.g., stochastic gradient descent methods (e.g., Duchi et al. [2011]) and concurrent computing techniques using GPUs (e.g., Raina et al. [2009]). Among them, we focus on the “computation on compressed data” approach, where we first compress the given data somehow and then employ an algorithm that works directly on the compressed data (i.e., without decompressing the data) to complete the task, in an attempt to reduce computation time and/or space. Algorithms on compressed data are mainly studied in string processing (e.g., Goto et al. [2013], Hermelin et al. [2009], Lifshits [2007], Lohrey [2012], Rytter [2004], enumera-
In particular, in the work on combinatorial optimization, they compress the set of feasible solutions that satisfy given constraints into a decision diagram so that minimizing a linear objective can be done by finding the shortest path in the decision diagram. Although we can find the optimal solution very efficiently when the size of the decision diagram is small, the method can only be applied to specific types of discrete optimization problems where the feasible solution set is finite, and the objective function is linear.

Whereas, we mainly consider a more general form of discrete/continuous optimization problems that include linear constraints with integer coefficients:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad Ax \geq b$$

for some $A \in \mathbb{C}^{m \times n}$ and $b \in \mathbb{C}^m$, where $X$ denotes the constraints other than $Ax \geq b$, and $C$ is a finite subset of integers. This class of problems includes LP, QP, SDP, and MIP with linear constraints of integer coefficients. So our target problem is fairly general. Without loss of generality, we assume $m > n$, and we are particularly interested in the case where $m$ is huge.

In this paper, we propose a pre-processing method that "rewrites" integer-valued linear constraints with equivalent but more concise ones. More precisely, we propose a general algorithm that, when given an integer-valued constraint matrix $(A, b) \in \mathbb{C}^{m \times n} \times \mathbb{C}^m$ of an optimization problem $[1]$, produces a matrix $(A', b') \in \mathbb{C}^{m' \times (n+n')} \times \mathbb{C}^{m'}$ that represents its extended formulation, that is, it holds that

$$\exists s \in \mathbb{R}^{n'}, A' \left[ \begin{array}{c} x \\ s \end{array} \right] \geq b' \iff Ax \geq b$$

for some $n'$ and $m'$, with the hope that the size of $(A', b')$ is much smaller than that of $(A, b)$ even at the cost of adding $n'$ extra variables. Using the extended formulation, we obtain an equivalent optimization problem to $[1]$:

$$\min_{x \in \mathbb{R}^n \cup \mathbb{R}^{n'}} f(x) \quad \text{s.t.} \quad A' \left[ \begin{array}{c} x \\ s \end{array} \right] \geq b'. \quad (2)$$

Then, we can apply any existing generic solvers, e.g., MIP/QP/LP solvers if $f$ is linear or quadratic, to $[2]$, combined with our pre-processing method, which may significantly reduce the computation time/space than applying them to the original problem $[1]$. To obtain a matrix $(A', b')$, we first construct a variant of a decision diagram called a Non-Deterministic Zero-Suppressed Decision Diagram (NZDD, for short) [Fujita et al., 2020] that somehow represents the matrix $(A, b)$. Observing that the constraint $Ax \geq b$ can be restated in terms of the NZDD constructed as "every path length is lower bounded by 0" for an appropriate edge weighting, we establish the extended formulation $(A', b') \in \mathbb{C}^{m' \times (n+n')} \times \mathbb{C}^{m'}$ with $m' = |E|$ and $n' = |V|$, where $V$ and $E$ are the sets of vertices and edges of the NZDD, respectively. One of the advantages of the result is that the size of the resulting optimization problem depends only on the size of the NZDD and the number $n$ of variables, but not on the number $m$ of the constraints in the original problem. Therefore, if the matrix $(A, b)$ is well compressed into a small NZDD, then we obtain an equivalent but concise optimization problem $[2]$.

To clarify the differences between our work and previous work regarding optimization using decision diagrams, we summarize the characteristics of both results in Table $[1]$. Notable differences are that (i) ours can treat optimization problems with any types of variables (discrete, or real), any types of objectives (including linear ones) but with integer coefficients on linear constraints, and (ii) ours uses decision diagrams for representing linear constraints while previous work uses them for representing feasible solutions of particular classes of problems. So, for particular classes of discrete optimization problems, the previous approach would work better with specific construction methods for decision diagrams. On the other hand, ours is suitable for continuous optimization problems or/discrete optimization problems for which efficient construction methods for decision diagrams representing feasible solutions are not known. See the later section for more detailed descriptions of related work.

| Table 1: Characteristics of previous work on optimization with decision diagrams (DDs) and ours. |
|---------------------------------------------------------------|
|                  | coeff. of lin. consts. | variables | objectives | DDs                        |
| Previous work    | any type              | binary/integer | any type  | linear                       |
| Ours             | binary/integer       | any type        | any type  | feasible solutions           |

Among various linear optimization problems, we consider the 1-norm regularized soft margin optimization as a non-trivial application of our method. This problem is a standard optimization problem in the machine learning literature,
categorized as LP, for finding sparse linear classifiers given labeled instances. This problem is motivating and challenging in that it has \( n + m \) variables and \( m \) linear constraints, so the naive application of our method will not be successful as the size of the NZDD representing the constraints is \( \Omega(m) \). For this problem, we propose a modified formulation that suffices to work well in practice, and we show efficient algorithms whose time complexity depends only on the size of the NZDD for the modified problem.

Furthermore, to realize succinct extended formulations, we propose practical heuristics for constructing NZDDs, which is our third contribution. Since it is not known to construct an NZDD of small size, we first construct a ZDD of minimal size, where the ZDD is a restricted form of the NZDD representation. To this end, we use a ZDD compression software called zcomp [Toda 2013]. Then, we give rewriting rules for NZDDs that reduce both the numbers of vertices and edges, and apply them to obtain NZDDs of smaller size of \( V \) and \( E \). Although the rules may increase the size of NZDDs (i.e., the total number of edge labels), the rules seem to work effectively since reducing \( |V| \) and \( |E| \) is more important for our purpose.

Experimental results on synthetic and real data sets show that our algorithms improve time/space efficiency significantly, especially when (i) \( m \gg n \), and (ii) the set \( C \) of integer coefficients is small, e.g., binary, where the datasets tend to have concise NZDD representations.

2 Related work

Various computational tasks over compressed strings or texts are investigated in algorithms and data mining literature, including, e.g., pattern matching over strings and computing edit distances or \( q \)-grams [Goto et al. 2009], [Hermelin et al. 2007], [Lohrey 2012], [Rytter 2004]. The common assumption is that strings are compressed using the straight-line program, which is a class of context-free grammars generating only one string (e.g., LZ77 and +LZ78). As notable applications of string compression techniques to data mining and machine learning, Nishino et al. [Nishino et al. 2014] and Tabei et al. (Tabei et al. 2016) reduce the space complexity of matrix-based computations. So far, however, string compression-based approaches do not seem to be useful for representing linear constraints.

Decision diagrams are used in the enumeration of combinatorial objects, discrete optimization and so on. In short, a decision diagram is a directed acyclic graph with a root and a leaf, representing a subset family of some finite ground set \( \Sigma \) or, equivalently, a boolean function. Each root-to-leaf path represents a set in the set family. The Binary Decision Diagram (BDD) [Bryant 1986], [Knuth 2011] and its variant, the Zero-Suppressed Binary Decision Diagram (ZDD) [Knuth 2011], [Minato 1993], are popular in the literature. These support various set operations (such as intersection and union) in efficient ways. Thanks to the DAG structure, linear optimization problems over combinatorial sets \( X \subset \{0, 1\}^n \) can be reduced to shortest/longest path problems over the diagrams representing \( X \). This reduction is used to solve the exact optimization of NP-hard combinatorial problems (see, e.g., [Bergman et al. 2016], [Castro et al. 2019], [Inoue et al. 2014], [Morrison et al. 2016]) and enumeration tasks [Minato 2017], [Minato and Uno 2010], [Minato et al. 2008]. Among work on decision diagrams, the work of Fujita et al. [Fujita et al. 2020] would be closest to ours. They propose a variant of ZDD called the Non-deterministic ZDD (NZDD) to represent labeled instances and show how to emulate the boosting algorithm AdaBoost [Rätsch and Warmuth 2005], a variant of AdaBoost [Freund and Schapire 1997] that maximizes the margin, over NZDDs. We follow their NZDD-based representation of the data. But our work is different from Fujita et al. in that, they propose specific algorithms running over NZDDs, whereas our work presents extended formulations based on NZDDs, which could be used with various algorithms.

The notion of extended formulation arises in combinatorial optimization (e.g., [Conforti et al. 2010], [Yannakakis 1991]). The idea is to re-formulate a combinatorial optimization with an equivalent different form, so that the size of the problem is reduced. For example, a typical NP-hard combinatorial optimization problem has an integer programming formulation of exponential size. Then a good extended formulation should have a smaller size than the exponential. Typical work on extended formulation focuses on some characterization of the problem to obtain succinct formulations (see, e.g., [Fiorini et al. 2021]). Our work is different from these in that we focus on the redundancy of the data and try to obtain succinct extended formulations for optimization problems described with data.

3 Preliminaries

The non-deterministic Zero-suppressed Decision Diagram (NZDD) [Fujita et al. 2020] is a variant of the Zero-suppressed Decision Diagram (ZDD) [Minato 1993], [Knuth 2011], representing subsets of some finite ground set \( \Sigma \). More formally, NZDD is defined as follows.

**Definition 1** (NZDD). An NZDD \( G \) is a tuple \( G = (V, E, \Sigma, \Phi) \), where \( (V, E) \) is a directed acyclic graph (\( V \) and \( E \) are the sets of nodes and edges, respectively) with a single root with no-incoming edges and a leaf with no outgoing
edges, $\Sigma$ is the ground set, and $\Phi : E \rightarrow 2^\Sigma$ is a function assigning each edge $e$ a subset $\Phi(e)$ of $\Sigma$. More precisely, we allow $(V, E)$ to be a multigraph, i.e., two nodes can be connected with more than one edge.

Furthermore, an NZDD $G$ satisfies the following additional conditions. Let $P_G$ be the set of paths in $G$ starting from the root to the leaf, where each path $P \in P_G$ is represented as a subset of $E$, and for any path $P \in P_G$, we abuse the notation and let $\Phi(P) = \bigcup_{e \in P} \Phi(e)$.

1. For any path $P \in P_G$ and any edges $e, e' \in P$, $\Phi(e) \cap \Phi(e') = \emptyset$. That is, for any path $P$, an element $a \in \Sigma$ appears at most once in $P$.
2. For any paths $P, P' \in P_G$, $\Phi(P) \neq \Phi(P')$. Thus, each path $P$ represents a different subset of $\Sigma$.

Then, an NZDD $G$ naturally corresponds to a subset family of $\Sigma$. Formally, let $L(G) = \{\Phi(P) \mid P \in P_G\}$. Figure 1 illustrates an NZDD representing a subset family $\{\{a, b, c\}, \{b\}, \{b, c, d\}, \{c, d\}\}$.

A ZDD [Minato 1993, Knuth 2011] can be viewed as a special form of NZDD $G = (V, E, \Sigma, \Phi)$ satisfying the following properties: (i) For each edge $e \in E$, $\Phi(e) = \{a\}$ for some $a \in \Sigma$ or $\Phi(e) = \emptyset$. (ii) Each internal node has at most two outgoing edges. If there are two edges, one is labeled with $\{a\}$ for some $a \in \Sigma$ and the other is labeled with $\emptyset$. (iii) There is a total order over $\Sigma$ such that, for any path $P \in P_G$ and for any $e, e' \in P$ labeled with singletons $\{a\}$ and $\{a'\}$ respectively, if $e$ is an ancestor of $e'$, $a$ precedes $a'$ in the order.

We believe that constructing a minimal NZDD for a given subset family is NP-hard since closely related problems are NP-hard. For example, constructing a minimal ZDD (over all orderings of $\Sigma$) is known to be NP-hard [Knuth 2011], and construction of a minimal NFA which is equivalent to a given DFA is P-space hard [Jiang and Ravikumar 1993]. On the other hand, there is a practical construction algorithm of ZDDs given a subset family and a fixed order over $\Sigma$ using multi-key quicksort [Toda 2013].

4. **NZDDs for linear constraints with binary coefficients**

In this section, we show an NZDD representation for linear constraints in problem (1) when linear constraints have $\{0, 1\}$-valued coefficients, that is, $C = \{0, 1\}$. We will discuss its extensions to integer coefficients in the later section. Let $a_i \in \{0, 1\}^n$ be the vector corresponding to the $i$-th row of the matrix $A \in \{0, 1\}^{m \times n}$ (for $i \in [m]$). For $x \in \{0, 1\}^n$, let $\text{idx}(x) = \{j \in [n] \mid x_j \neq 0\}$, i.e., the set of indices of nonzero components of $x$. Then, we define $I = \{\text{idx}(c_i) \mid c_i = (a_i, b_i), i \in [m]\}$. Note that $I$ is a subset family of $2^{[n+1]}$. Then we assume that we have some NZDD $G = (V, E, [n + 1], \Phi)$ representing $I$, that is, $L(G) = I$. We will later show how to construct NZDDs.

The following theorem shows the equivalence between the original problem (1) and a problem described with the NZDD $G$.

**Theorem 1.** Let $G = (V, E, [n + 1], \Phi)$ be an NZDD such that $L(G) = I$. Then the following optimization problem is equivalent to problem (1):

\[
\min_{x \in \mathbb{R}^n, a \in \mathbb{R}^{|V|}} f(x) \quad \text{subject to} \quad s_{e, u} + \sum_{j \in \Phi(e)} x_j \geq s_{e, v}, \quad \forall e \in E,
\]

where $e, u$ and $e, v$ are nodes that the edge $e$ is directed from and to, respectively.

Before going through the proof, let us explain some intuition on problem (3). Intuitively, each linear constraint in (1) is encoded as a path from the root to the leaf in the NZDD $G$, and a new variable $s_e$ for each node $v$ represents a lower
bound of the length of the shortest path from the root to \( v \). The inequalities in (3) reflect the structure of the standard dynamic programming of Dijkstra, so that all inequalities are satisfied if and only if the length of all paths is larger than zero. In Figure 2 we show an illustration of the extended formulation.

**Proof.** Let \( x_\ast \) and \( (\hat{x}', \hat{s}) \) be the optimal solutions of problems (1) and (3), respectively. It suffices to show that each optimal solution can construct a feasible solution of the other problem.

Let \( \hat{x} \) be the vector consisting of the first \( n \) components of \( \hat{x}' \). For each constraint \( a_i^\top x \geq b_i \) \((i \in [m])\) in problem (1), there exists the corresponding path \( P_i \in \mathcal{P}_G \). By repeatedly applying the first constraint in (3) along the path \( P_i \), we have \( \sum_{e \in P_i} \sum_{j \in \Phi(e)} \bar{z}_{i,j} \geq s_{\text{leaf}} = 0 \). Further, since \( \Phi(P_i) \) represents the set of indices of nonzero components of \( c_i \), \( \sum_{e \in P_i} \sum_{j \in \Phi(e)} \bar{z}_{i,j} = c_i^\top \hat{x}' - b_i \). By combining these inequalities, we have \( a_i^\top \hat{x} - b_i \geq 0 \). This implies that \( \hat{x} \) is a feasible solution of (1) and thus \( f(x_\ast) \leq f(\hat{x}) \).

Let \( \hat{x}'_\ast = (x_\ast, -1) \). Assuming a topological order on \( V \) (from the root to the leaf), we define \( s_{\ast, \text{root}} = s_{\ast, \text{leaf}} = 0 \) and \( s_{\ast, v} = \min_{e \in E, e.v = v} s_{\ast, e.u} + \sum_{j \in \Phi(e)} \bar{z}'_{i,j} \) for each \( v \in V \setminus \{\text{root, leaf}\} \). Then, we have, for each \( e \in E \) s.t. \( e.v \neq \text{leaf} \), \( s_{\ast, e.v} \leq s_{\ast, e.u} + \sum_{j \in \Phi(e)} \bar{z}'_{i,j} \) by definition. Now, \( \min_{e \in E, e.v = \text{leaf}} s_{\ast, e.u} + \sum_{j \in \Phi(e)} \bar{z}'_{i,j} \) is achieved by a path \( P \in \mathcal{P}_G \) corresponding to \( \arg \min_{i \in [m]} a_i^\top x_\ast - b_i \), which is \( \geq 0 \) since \( x_\ast \) is feasible w.r.t. (1). Therefore, \( s_{\ast, e.u} \leq s_{\ast, e.u} + \sum_{j \in \Phi(e)} \bar{z}'_{i,j} \) for \( e \in E \) s.t. \( e.v = \text{leaf} \) as well. Thus, \( (x'_\ast, s_\ast) \) is a feasible solution of (3) and \( f(\hat{x}) \leq f(x'_\ast) \).

Given the NZDD \( G = (V, E) \), problem (3) contains \( n + |V| \) variables and \( |E| \) linear constraints, where \( |V| \) variables are real. In particular, if problem (1) is LP or IP, then problem (3) is LP, or MIP, respectively.

## 5 Extensions to integer coefficients

We briefly discuss how to extend our NZDD representation of linear constraints to the cases where coefficients of linear constraints belong to a finite set \( C \) of integers. There are two ways to do so.

**Binary encoding of integers** We assume some encoding of integers in \( C \) with \( O(\log |C|) \) bits. Then, each bit can be viewed as a binary-valued variable. Each integer coefficient can be also recovered with its binary representation. Under this attempt, the resulting extended formulation has \( O(n \log |C| + |V|) \) variables and \( O(|E|) \) linear constraints.

**Extending \( \Sigma \)** Another attempt is to extend the domain \( \Sigma \) of an NZDD \( G = (V, E, \Sigma, \Phi) \). The extended domain \( \Sigma' \) consists of all pairs of integers in \( C \) and elements in \( \Sigma \). Again, integer coefficients are recovered through the new domain \( \Sigma' \). The resulting extended formulation has \( O(n|C| + |V|) \) variables and \( O(|E|) \) linear constraints. While the size of the problem is larger than the binary encoding, its implementation is easy in practice and could be effective for \( C \) of small size.

## 6 1-norm regularized soft margin optimization

The 1-norm regularized soft margin optimization is a standard linear programming formulation of finding a sparse linear classifier with large margin (see, e.g., [Demiriz et al. 2002], [Warmuth et al. 2007, 2008]). We are given...
sequence $S$ of labeled instances $S = ((x_1, y_1), \ldots, (x_m, y_m)) \in (X \times \{-1, 1\})^m$, where $X \subset \mathbb{R}^n$ is the set of instances. For clarity, we assume that the domain is the set of binary vectors, i.e., $X = \{0, 1\}^n$. This is common in many applications when we employ the bag-of-words representation of instances. Given a parameter $\nu \in (0, 1]$ and a sequence $S$ of labeled instances, the 1-norm regularized soft margin optimization is defined as follows\footnote{For the sake of simplicity, we show a restricted version of the formulation. We can easily extend the formulation with positive and negative weights by considering positive and negative weights $w_j^+$ and $w_j^-$ instead of $w_j$ ad replacing $w_j$ with $w_j^+ - w_j^-$.}.

$$\max_{\rho, w, \xi} \quad \rho - \frac{1}{\nu m} \sum_{i=1}^m \xi_i$$

s.t.  $y_i (w^T x_i - b) \geq \rho - \xi_i \quad \forall i = 1, \ldots, m,$

$$\sum_j w_j + b = 1, \quad w \in \mathbb{R}_+^n, b \geq 0, \xi \in \mathbb{R}_+^m.$$ 

For the parameter $\nu \in (0, 1]$ and the optimal solution $(\rho^*, w^*, b^*, \xi^*)$, by a duality argument, it can be verified that there are at least $(1 - \nu)m$ instances that has margin larger than $\rho^*$, i.e., $y_i (w^*^T x_i - b^*) \geq \rho^*$\cite{Demiriz2002}.

We formulate a variant of the 1-norm regularized soft margin optimization based on NZDDs and propose efficient algorithms. Our generic re-formulation of (3) can be applied to the soft margin problem (4) as well. However, a direct application is not successful since problem (4) contains $O(m)$ slack variables $\xi$ and each $\xi_i$ appears only once in $m$ linear constraints. That implies a resulting NZDD contains $\Omega(m)$ edges. Therefore, we are motivated to formulate a soft margin optimization for which a succinct NZDD representation exists.

Our basic idea is as follows: Suppose that we have some NZDD $G = (V, E, \Phi, \Sigma)$ such that each path $P$ corresponds to a constraint $y_i (w^\top x_i + b) \geq \rho$. Our idea is to introduce a slack variable $\beta_e$ on each edge $e$ along the path $P$, instead of using a slack variable $\xi_i$ for each instance $i \in [m]$.

Given some NZDD $G = (V, E, \Sigma, \Phi),$

$$\max_{\rho, \beta, w} \quad \rho - \frac{1}{\nu m} \sum_{i=1}^m \sum_{e \in P_i} \beta_e$$

s.t.  $y_i (w^\top x_i - b) \geq \rho - \sum_{e \in P_i} \beta_e, \forall i = 1, \ldots, m,$

$$\sum_{j=1}^n w_j + b = 1, \quad b \geq 0, w \in \mathbb{R}_+^n, \beta \in \mathbb{R}^E_+.$$ 

Note that, the sum of the slack variables $\sum_{e \in P_i} \beta_e$ for each instance $x_i$ is more restricted than the original slack variable $\xi_i$. This observation implies the following.

**Proposition 1.** An optimal solution of problem (5) is a feasible solution of problem (4).
Algorithm 1 Column Generation

Input: NZDD $G = (V, E, \Sigma, \Phi)$.

1: Let $d_1 \in [0, 1]^{|E|}$ be any vector satisfying (9), (10), and (11). Let $J_0 = \emptyset$.
2: for $t = 1, 2, \ldots$ do
3: Let $j_t = \arg \max_{j \in [n+1]} \text{sign}(j) \sum_{e \in \Phi(e)} \text{sign}(e) d_e$ and let $\hat{\gamma}_t$ be its objective value.
4: If $\hat{\gamma}_t \leq \gamma_t + \varepsilon$, let $T = t - 1$ and break.
5: Let $J_t = J_{t-1} \cup \{j_t\}$ and update $(d_{t+1}, \gamma_{t+1})$ as:

\[
\min_{d, \gamma} \gamma
\]
\[
\text{s.t. } \sum_{e \in \Phi(e)} \text{sign}(e) d_e \leq \gamma, \forall j \in J_t,
\]

constraints (9), (10), and (11).
6: end for

Output: Output the Lagrangian coefficients $(w_T, \beta_T, \rho_T)$ for the subproblem w.r.t. $J_T$.

with $\Phi(e) = \emptyset$, and (iii) for other edges $e \in E$, $\Phi(e) = \Phi^+(e)$ if $e \in E^+$ and $\Phi(e) = \Phi^-(e)$ if $e \in E^-$.

\[
\max_{\rho, w, \beta, s} \rho - \frac{1}{\nu m} \sum_{e \in E} m_e \beta_e
\]
\[
\text{s.t. } s_{e,u} + \text{sign}(e) \sum_{j \in \Phi(e)} w_j + \beta_e \geq s_{e,v}, \forall e \in E,
\]

$s_{\text{root}} = 0$, $s_{\text{leaf}} \geq \rho$,

$\sum_{j=1}^n w_j - w_{n+1} = 1$,

$w_j \geq 0, i \in [n], w_{n+1} \leq 0, \beta \geq 0$,

where $m_e$ is the number of paths going thorough the edge $e$, and $\text{sign}(e)$ is defined as $\text{sign}(e) = -1$ if $e \in E^-$ and $\text{sign}(e) = 1$, otherwise. The constants $m_e$ can be computed in time $O(|E|)$ a priori by a dynamic programming over $G$. Note that the bias term $-b$ correspond to $w_{n+1}$ for notatinal convenience. Then, by following the same proof argument of Theorem 1, we have the following corollary.

**Corollary 1.** Problem (6) is equivalent to problem (5).

Problem (6) has $O(n + |V| + |E|)$ variables and $O(|E|)$ linear constraints, whereas the original formulation (4) has $O(n + m)$ variables and $O(m)$ linear constraints. So, with a concise NZDD representation of the sample, we obtain an extended formulation whose size is independent of $m$ of the sample size. In later subsections, we propose two efficient solving methods for problem (6).

6.1 Column Generation

In this subsection, we propose a column generation-based method for solving the modified 1-norm soft margin optimization problem (6). Although the size of the extended formulation (6) does not depend on the size of linear constraints $m$ of the original problem, it still depends on $n$, the size of variables. The column generation is a standard approach of linear programming that tries to reduce either the size of linear constraints/variables by solving smaller subproblems. In our case, we try to avoid problems depending on the size of variables $n$. 
By a standard dual argument of linear programming, the equivalent dual problem of (6) is given as follows.

\[
\min_{\gamma, d} \gamma \quad \text{s.t.} \quad \sum_{e \in E} \text{sign}(j) \sum_{e \in \Phi(e)} \text{sign}(e)d_e \leq \gamma \quad (j \in [n+1])
\]

\[
\sum_{e \in E} d_e = \sum_{e \in E} d_e, \quad \forall u \in V \setminus \{\text{root}, \text{leaf}\},
\]

\[
\sum_{e \in E} d_e = 1, \quad \sum_{e \in E} d_e = 1,
\]

\[
0 \leq d_e \leq \frac{m_e}{\mu_m} \quad (e \in E),
\]

where \(\text{sign}(j) = 1\) for \(j \in [n]\) and \(\text{sign}(j) = -1\) for \(j = n + 1\). Here, the dual problem (8) has \(|E| + 1\) variables and \(n\) linear constraints. Roughly speaking, this problem is to find a vector \(d\) that represents a “flow” from the root to the leaf in the NZDD optimizing some objective, where the total flow is 1. The objective is \(\gamma\), the upper bound of \(\text{sign}(j) \sum_{e \in \Phi(e)} \text{sign}(e)d_e\) for each \(j \in [n+1]\). The column generation-based algorithm is given in Algorithm 1. The algorithm repeatedly solves the subproblems whose constraints related to \(\gamma\) are only restricted to a subset \(J_t \subseteq [n+1]\). Then it adds \(J_{t+1} \setminus J_t\) (updated as \(J_{t+1}\), where \(J_{t+1}\) corresponds to the constraint that violates condition (9) the most with respect to the current solution \((\gamma_t, d_t)\). It can be shown that the column-generation algorithm finds an \(\varepsilon\)-approximate solution.

**Theorem 2.** Algorithm 1 outputs an \(\varepsilon\)-approximate solution of (6).

**Proof.** Let \((d^*, \gamma^*)\) be an optimal solution of (5) and let \(\pi^*\) and \(\pi_T\) be the optimum of (6) and the primal one of the dual subproblem for \(T\), respectively. By the duality, we have \(\gamma_T = \pi_T\) and \(\gamma^* = \pi^*\). We show that \(\pi_T \geq \pi^* - \varepsilon\). By definition of \(T\), for any \(j \in [n+1] \setminus J_T\), \(\text{sign}(j) \sum_{e \in \Phi(e)} \text{sign}(e)d_e \leq \hat{\gamma}_{T+1} \leq \gamma_T + \varepsilon\). Then, \((d_{T+1}, \gamma_T + \varepsilon)\) is a feasible solution of (6). So, \(\text{sign}(j_{T+1}) \sum_{e \in \Phi(e)} \text{sign}(e)d_e = \hat{\gamma}_{T+1} \geq \gamma^*\). Combining these observations, \(\pi_T = \gamma_T \geq \gamma^* - \varepsilon = \pi^* - \varepsilon\) as claimed. □

As for its time complexity analysis, similar to other column generation techniques, we do not have non-trivial iteration bounds. In the next section, we propose another algorithm with theoretical guarantee of an iteration bound.

### 6.2 Performing ERLPBoost over an NZDD

We can emulate ERLPBoost [Warmuth et al. 2008] on an NZDD. The algorithm is the same as ERLPBoost except for the update rule. Let \(d_e^t = \frac{m_e}{\mu_m}\) for all \(e \in E\). In each iteration \(t\), the compressed version of ERLPBoost solves the following sub-problem:

\[
\min_{\gamma, d} \gamma + \frac{1}{\eta} \left[ \sum_{e \in E} d_e \ln \frac{d_e}{d_e^t} - d_e + d_e^t \right] \quad \text{s.t.} \quad \sum_{e \in \Phi(e)} \text{sign}(e)d_e \leq \gamma, \quad \forall j \in J_t,
\]

\[
\text{constraints (9), (10), and (11)}.
\]

Here, \(\eta > 0\) is some parameter. One can rewrite (12) in terms of \(d\) by introducing \(\max\) function. We denote the resulting objective function as \(P^t(d)\). Algorithm 2 shows ERLPBoost over an NZDD. We can also obtain a similar iteration bound like ERLPBoost.

**Theorem 3.** If \(\eta = \frac{2}{\varepsilon} \text{depth}(G) \max\{1, \ln \frac{1}{\mu}\}\), then Algorithm 2 finds an \(\varepsilon\)-approximate solution to (8) in \(T \leq \frac{144}{\varepsilon^2} \text{depth}(G)^2 \max\{1, \ln \frac{1}{\mu}\}\) iterations.

Here, we prove Theorem 3 with weaker assumption; we assume that the weak learner returns a hypothesis \(j_t \in \{1, 2, \ldots, n\}\) satisfying

\[
\sum_{e \in E} \text{sign}(e)d_e^{t-1} \geq g
\]

for some unknown value \(g > 0\). This assumption is similar to the one in ERLPBoost [Warmuth et al. 2008]. Under the above assumption, we state the iteration bound.
Algorithm 2 ERLPBoost over an NZDD

**Input:** NZDD $G = (V, E, \Sigma, \Phi)$

1. Set $d^0_e = \frac{\nu e}{m}$ for all $e \in E$. Let $J_0 = \emptyset$.
2. for $t = 1, 2, \ldots$ do
3. \hspace{1em} Find a hypothesis $j_t \in [n + 1]$ that maximizes the edge w.r.t. $d^{t-1}$.
4. \hspace{1em} Set $\delta^t := \min_{1 \leq q \leq L} P^t(d^{q-1}) - P^{t-1}(d^{t-1})$.
5. \hspace{1em} If $\delta^t \leq \varepsilon/2$, Set $T = t - 1$ and break.
6. \hspace{1em} Compute the minimizer $d^t$ of (12).
7. end for
8. Solve $J$ over $J_T = \{j_t\}^T_{t=1}$ to get the optimal weights $u^T$ on hypotheses.

**Output:** Output $f = \sum_{j \in J} u^T_j h_j$.

**Theorem 4.** If $\eta = \frac{1}{2} \cdot \text{depth}(G) \max\{1, \ln \frac{1}{\nu}\}$, then Algorithm 2 outputs a solution whose value of $\mathcal{J}$ is at least $g - \varepsilon$. Algorithm 2 runs at most $T \leq \frac{144}{\varepsilon^2} \text{depth}(G)^2 \max \{1, \ln \frac{1}{\nu}\}$ iterations.

To prove Theorem 4, we give some technical lemmata.

**Lemma 1.** Let $\text{depth}(G) = \max_{P \in \mathcal{G} \subseteq 2^E} |P|$. Let $d \in \mathbb{R}^{|E|}$ be any feasible solution of (12). Then, $\sum_{e \in E} d_e \leq \text{depth}(G)$.

**Proof.** Let $G'$ be a layered NZDD of $G$ obtained by adding some redundant nodes. Since this operation only increases the number of edges, $ \sum_{e \in E} d_e \leq \sum_{e \in E'} d_e $ holds, where $E'$ is the set of edges of $G'$. Let $E'_k = \{e \in E' \mid e \text{ is an edge at depth } k\}$. Then,

$$ \sum_{e \in E} d_e \leq \sum_{e \in E'} d_e = \sum_{k=1}^{\text{depth}(G)} \sum_{e \in E'_k} d_e = \sum_{k=1}^{\text{depth}(G)} 1 = \text{depth}(G), $$

where at the second equality, we used the fact that for a feasible $d$, the total flow at any depth equals to 1.

The following lemma shows an upper bound of the unnormalized relative entropy for a feasible distribution.

**Lemma 2.** Let $d$ be a feasible solution to (12) for an NZDD $G$. Then, the following inequality holds.

$$ \sum_{e \in E} d_e \ln \frac{d_e}{d^0_e} - d_e + d^0_e \leq \text{depth}(G) \ln \frac{1}{\nu} $$

**Proof.** Without loss of generality, we can assume that the NZDD is layered. Indeed, if the NZDD is not layered, we can add dummy nodes and dummy edges to make the NZDD layered. Further, we can increase the edges on the NZDD to make $m_e = 1$ for all edge $e \in E$. Figure 3 and 4 depict these manipulations. With these conversions, the initial distribution $d^0$ becomes $d^0_e = 1/m$ for all $e \in E$ and the feasible region becomes $E_k \subseteq E$ be the set of edges at depth $k$.

$$ \sum_{e \in E} d_e \ln \frac{d_e}{d^0_e} - d_e + d^0_e = \sum_{k=1}^{\text{depth}(G)} \left( \sum_{e \in E_k} d_e \ln \frac{d_e}{d^0_e} - d_e + d^0_e \right) = \sum_{k=1}^{\text{depth}(G)} \sum_{e \in E_k} d_e \ln \frac{d_e}{d^0_e}, $$

where the last equality holds since in each layer, the total flow equals to one. For each edge $e \in E$ with $m_e > 1$, define a set $\hat{E}(e)$ of duplicated edges of size $|\hat{E}(e)| = m_e$. Further, define

$$ \forall \hat{e} \in \hat{E}(e), \quad \hat{d}_e = \frac{d_e}{m_e}, \quad \hat{d}^0_e = \frac{d^0_e}{m_e} = \frac{1}{\nu m}. $$

Then,

$$ \sum_{\hat{e} \in \hat{E}(e)} \hat{d}_e \ln \frac{\hat{d}_e}{\hat{d}^0_e} = \sum_{\hat{e} \in \hat{E}(e)} \frac{d_e}{m_e} \ln \frac{d_e/m_e}{d^0_e/m_e} = d_e \ln \frac{d_e}{d^0_e} $$

9
Lemma 3. Let criterion for algorithm 2.

Subtracting \( P_T \) holds. Thus, for any feasible solution \( d \), we can realize the same relative entropy value by \( \hat{d} \). Therefore, we can rewrite (14) as

\[
\sum_{k=1}^{\text{depth}(G)} \sum_{e \in E_k} d_e \ln \frac{d_e}{d^0_e} = \sum_{k=1}^{\text{depth}(G)} \sum_{\hat{e} \in \bigcup_{e \in E} E(\hat{e})} \hat{d}_e \ln \frac{\hat{d}_e}{d^0_e}.
\]

(15)

By construction of \( \hat{E}(\epsilon) \), \( \hat{d}_e \in [0, 1/\nu m] \). Therefore, we can bound eq. (15) by

\[
\sum_{k=1}^{\text{depth}(G)} \sum_{e \in E_k} d_e \ln \frac{d_e}{d^0_e} \leq \sum_{k=1}^{\text{depth}(G)} \ln \frac{m}{\nu m} = \text{depth}(G) \frac{1}{\nu}.
\]

Note that if the algorithm always finds a hypothesis that maximizes the right-hand-side of (13), then this lemma guarantees the \( \epsilon \)-accuracy to the optimal solution of (8).

Lelem.

Let \( \delta_t = \min_{k \in [T]} P^k(d^{k-1}) - P^{t-1}(d^{t-1}) \) be the optimality gap. The following lemma justifies the stopping criterion for algorithm 2.

Lemma 3. Let \( P^T_{LP} \) be an optimal solution of (8) over \( J_T = \{j_1, j_2, \ldots, j_T\} \). If \( \eta \geq \frac{2}{\epsilon} \text{depth}(G) \ln \frac{1}{\nu} \), then \( \delta^{T+1} \leq \epsilon/2 \) implies \( g - P^T_{LP} \leq \epsilon \), where \( g \) is the guarantee of the base learner.

Let \( g \) be an optimal solution of (8), then this lemma guarantees the \( \epsilon \)-accuracy to the optimal solution of (8).

Proof. Let \( P^T(d) \) be the objective function of (12) over \( J_T = \{j_1, j_2, \ldots, j_T\} \) obtained by algorithm 2 and let \( d^T \) be the optimal feasible solution of it. By the choice of \( \eta \) and lemma 2 we get

\[
\frac{1}{\eta} \sum_{e \in E} \left[ d_e \ln \frac{d_e}{d^0_e} - d_e + d^0_e \right] \leq \frac{1}{\eta} \text{depth}(G) \ln \frac{1}{\nu} \leq \frac{\epsilon}{2}.
\]

Thus, \( P^T(d^T) \leq P^T_{LP} + \epsilon/2 \). On the other hand, by the assumption on the weak learner, for any feasible solution \( d^{t-1} \), we obtain a \( j_t \in [n+1] \) such that

\[
g \leq \sum_{e \in E} \text{sign}(e) d^t_e - 1.
\]

Using the nonnegativity of the unnormalized relative entropy, we get

\[
g \leq \min_{t \in [T+1]} \sum_{e \in E} \text{sign}(e) d^t_e - 1 \leq \min_{t \in [T+1]} P^t(d^{t-1}) \leq P^T(d^{t-1}).
\]

Subtracting \( P^T(d^T) \) from both sides, we get

\[
g - P^T(d^T) \leq \min_{t \in [T+1]} P^t(d^{t-1}) - P^t(d^t) = \delta^{T+1} \leq \frac{\epsilon}{2}.
\]

Thus, \( \delta^{T+1} \leq \epsilon/2 \) implies \( g - P^T_{LP} \leq \epsilon \).
With lemma 3, we can obtain a similar iteration bound like ERLPBoost. To see that, we need to derive the dual problem of (12). By standard calculation, you can verify that the dual problem becomes:

$$\begin{align*}
\max_{w, s, \beta} & \quad \frac{1}{\eta} \sum_{e \in E} d_e^0 \left( 1 - e^{-\eta A_e'(w, s, \beta)} \right) + s_{\text{root}} - s_{\text{leaf}} - \frac{1}{\nu m} \sum_{e \in E} m_e \beta_e \\
\text{sub. to} & \quad \sum_{j \in J} w_j = 1, w \geq 0, \beta \geq 0
\end{align*}$$

(16)

Let $P^t(d)$, $D^t(w, s, \beta)$ be the objective function of the optimization sub-problems (12), (16) respectively. Let $d^t$ be the optimal solution of (12) at round $t$ and similarly, let $(w^t, s^t, \beta^t)$ be the one of (16). Then, by KKT conditions, the following hold.

$$\begin{align*}
\sum_{j \in J} \text{sign}(j) w_j \left[ \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e \right] &= \max_{e \in E, j \in \Phi(e)} \sum_{j \in J} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e \tag{17}
\end{align*}$$

We will prove the following lemma, which corresponds to Lemma 2 in [Warmuth et al. 2008].

**Lemma 4.** If $\eta \geq 1/3$, then

$$P^t(d^t) - P^{t-1}(d^{t-1}) \geq \frac{1}{18\eta \text{depth}(G)} \left[ P^t(d^{t-1}) - P^{t-1}(d^{t-1}) \right]^2,$$

where $\text{depth}(G)$ denotes the max depth of graph $G$.

**Proof.** First of all, we examine the right hand side of the inequality. By definition, $P^t(d^{t-1}) \geq P^{t-1}(d^{t-1})$ and

$$P^t(d^{t-1}) - P^{t-1}(d^{t-1}) = \sum_{e \in E, j \in \Phi(e)} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^{t-1} - \max_{j \in J_{t-1}} \sum_{e \in E, j \in \Phi(e)} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^{t-1}$$

$$= \sum_{e \in E, j \in \Phi(e)} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^{t-1} - \sum_{j \in J_{t-1}} \sum_{e \in E, j \in \Phi(e)} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^{t-1}$$

$$= \sum_{e \in E, j \in \Phi(e)} \text{sign}(j) \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^{t-1} \left[ \text{sign}(j) I_{j \in \Phi(e)} - \sum_{j \in J_{t-1}} \text{sign}(j) I_{j \in \Phi(e)} w_j^{t-1} \right]$$

$$= \sum_{e \in E} d_e^{t-1} x_e^t,$$

where the second equality holds from (17).

Now, we will bound $P^t(d^t) - P^{t-1}(d^{t-1})$ from below. For $\alpha \in [0, 1]$, let

$$w^t(\alpha) := (1 - \alpha) \begin{bmatrix} w^{t-1} \\ 0 \end{bmatrix} + \alpha \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{18}$$

Since $(w^t, s^t, \beta^t)$ is the optimal solution of (12) at round $t$, $D^t(w^t, s^t, \beta^t) \geq D^t(w^t(\alpha), s^{t-1}, \beta^{t-1})$ holds. By strong duality,

$$P^t(d^t) - P^{t-1}(d^{t-1}) = D^t(w^t, s^t, \beta^t) - D^{t-1}(w^{t-1}, s^{t-1}, \beta^{t-1})$$

$$\geq D^t(w^t(\alpha), s^{t-1}, \beta^{t-1}) - D^{t-1}(w^{t-1}, s^{t-1}, \beta^{t-1})$$

$$= \frac{1}{\eta} \sum_{e \in E} d_e^0 \left( 1 - e^{-\eta A_e'(w^t(\alpha), s^{t-1}, \beta^{t-1})} \right) - \frac{1}{\eta} \sum_{e \in E} d_e^0 \left( 1 - e^{-\eta A_e'(w^{t-1}, s^{t-1}, \beta^{t-1})} \right)$$

$$= -\frac{1}{\eta} \sum_{e \in E} d_e^0 \left( e^{-\eta A_e'(w^t(\alpha), s^{t-1}, \beta^{t-1})} - e^{-\eta A_e'(w^{t-1}, s^{t-1}, \beta^{t-1})} \right).$$
By KKT conditions, we can write $d^{t-1}$ in terms of the dual variables $(w^{t-1}, s^{t-1}, \beta^{t-1})$:

$$d_j^{t-1} = d_0^j \exp \left[ -\eta \left( \text{sign}(e) \sum_{j \in \mathcal{J}_{t-1}} \text{sign}(j) I_{[j \in \Phi(e)]} w_j^{t-1} + s_u^{t-1} - s_v^{t-1} + \beta_c^{t-1} \right) \right] = d_0^j e^{-\eta A_c^{t-1} (w^{t-1}, s^{t-1}, \beta^{t-1})}$$

Therefore,

$$P^t(d^t) - P^{t-1}(d^{t-1}) \geq -\frac{1}{\eta} \sum_{e \in E} d_0^e e^{-\eta A_c^{t-1} (w^{t-1}, s^{t-1}, \beta^{t-1})} \left( e^{-\eta \alpha x_e^t} - 1 \right) = -\frac{1}{\eta} \sum_{e \in E} d_0^e \left( e^{-\eta \alpha x_e^t} - 1 \right).$$

Since $x_e^t \in [-2, +2]$, $\frac{\alpha + x_e^t}{6} \in [0, 1]$ and $\frac{\alpha + x_e^t}{6} + \frac{\alpha - x_e^t}{6} = 1$. Thus, using Jensen’s inequality, we get

$$P^t(d^t) - P^{t-1}(d^{t-1}) \geq -\frac{1}{\eta} \sum_{e \in E} d_0^e \left( \exp \left[ \frac{3 + x_e^t}{6} (-3\eta \alpha) + \frac{3 - x_e^t}{6} (3\eta \alpha) \right] - 1 \right) \geq -\frac{1}{\eta} \sum_{e \in E} d_0^e \left( \frac{3 + x_e^t}{6} e^{-3\eta \alpha} + \frac{3 - x_e^t}{6} e^{3\eta \alpha} - 1 \right) =: R(\alpha).$$

The above inequality holds for all $\alpha \in [0, 1]$. Here, $R(\alpha)$ is a concave function w.r.t. $\alpha$ so that we can choose the optimal $\alpha \in [0, 1]$. By standard calculation, we get that the optimal $\alpha \in \mathbb{R}$ is

$$\alpha = \frac{1}{6\eta} \ln \frac{\sum_{e \in E} d_0^e (3 + x_e^t)}{\sum_{e \in E} d_0^e (3 - x_e^t)} \quad (19)$$

Since $x_e^t \leq 2$ for all $e \in E$, $\alpha \leq \frac{1}{6\eta} \ln \frac{1}{\frac{1}{3}} \leq \frac{1}{3\eta}$. Thus, $\alpha \leq 1$ holds. On the other hand, $\sum_{e \in E} d_0^e x_e^t \geq 0$ so that $\alpha \geq \frac{1}{6\eta} \ln 1 = 0$. Therefore, we can use (19) to lower-bound $P^t(d^t) - P^{t-1}(d^{t-1})$.

$$P^t(d^t) - P^{t-1}(d^{t-1}) \geq -\frac{1}{\eta} \left[ \frac{1}{3} \left( \sum_{e \in E} d_0^e (3 + x_e^t) \right) \left( \sum_{e \in E} d_0^e (3 - x_e^t) \right) - \sum_{e \in E} d_0^e \right]$$

$$= -\frac{1}{\eta} \left[ \frac{1}{3} \left( \sum_{e \in E} d_0^e \right)^2 - \left( \sum_{e \in E} d_0^e x_e^t \right)^2 - \sum_{e \in E} d_0^e \right]$$

By using the inequality

$$\forall a, b \geq 0, 3a \geq b \implies \frac{1}{3} \sqrt{9a^2 - b^2} - a \leq -\frac{b^2}{18a},$$

we get

$$P^t(d^t) - P^{t-1}(d^{t-1}) \geq \frac{1}{18\eta} \left( \sum_{e \in E} d_0^e x_e^t \right)^2$$

$$\geq \frac{1}{18\eta} \left( \sum_{e \in E} d_0^e \right)^2 = \frac{1}{18\eta \text{depth}(G)} \left[ P^t(d^{t-1}) - P^{t-1}(d^{t-1}) \right]^2,$$

which is the inequality we desire. \qed

We introduce the following lemma to prove the iteration bound of the compressed ERLPBoost.

**Lemma 5 (Abe et al. [2001]).** Let $(\delta^t)_{t \in \mathbb{N}} \subset \mathbb{R}_{\geq 0}$ be a sequence such that

$$\exists c > 0, \forall t \geq 1, \delta^t - \delta^{t+1} \geq \frac{(\delta^t)^2}{c}.$$ 

Then, the following inequality holds for all $t \geq 1$.

$$\delta^t \leq \frac{c}{t - 1 + \frac{c}{\delta^t}}$$
Specifically, in the case of type (ii), we have nodes with one outgoing edge. A node constraints. More precisely, the output ZDDs of the zcomp often contains (i) nodes with one incoming edge or (ii) produces concise ZDDs compared to inputs. But, concise ZDDs do not always imply concise representations of linear

Thus, we get the following relation:

\[ \delta' - \delta \geq \frac{(\delta')^2}{c \eta}, \] where \( c = 18 \text{depth}(G) \).

Lemma 5 gives us \( \delta' \leq \frac{c \eta}{1 + \frac{\eta}{\varepsilon}} \). Rearranging this inequality, we get \( T \leq \frac{c \eta}{\varepsilon} - \frac{c \eta}{\eta} + 1 \). Since \( \delta' = \sum_{e \in E, j \in \Phi(e)} \text{sign}(e) d_e^0 \leq \text{depth}(G) \), we have \( \frac{c \eta}{\varepsilon} \geq \frac{72}{1} \text{depth}(G) > 1 \). Therefore, the above inequality implies that \( T \leq \frac{c \eta}{\eta} \). As long as the stopping criterion does not satisfy, \( \delta' > \varepsilon/2 \) hold. Thus,

\[ T \leq \frac{1}{T^2} \cdot 36 \text{depth}(G) \cdot \frac{2}{\varepsilon} \text{depth}(G) \cdot \max \left(1, \ln \frac{1}{\nu}\right) \]

\[ \leq \frac{144}{\varepsilon^2} \text{depth}(G)^2 \cdot \max \left(1, \ln \frac{1}{\nu}\right). \]

\[ \square \]

7 Construction of NZDDs

We propose heuristics for constructing NZDDs given a subset family \( S \subseteq 2^\Sigma \). We use the zcomp \( \text{zcomp} \), developed by Toda, to compress the subset family \( S \) to a ZDD. The zcomp is designed based on multikey quicksort \( \text{Bentley and Sedgewick [1997]} \) for sorting strings. The running time of the zcomp is \( O(N \log^2 |S|) \), where \( N \) is an upper bound of the nodes of the output ZDD and \( |S| \) is the sum of cardinalities of sets in \( S \). Since \( N \leq |S| \), the running time is almost linear in the input.

A naive application of the zcomp is, however, not very successful in our experiences. We observe that the zcomp often produces concise ZDDs compared to inputs. But, concise ZDDs do not always imply concise representations of linear constraints. More precisely, the output ZDDs of the zcomp often contains (i) nodes with one incoming edge or (ii) nodes with one outgoing edge. A node \( v \) of these types introduces a corresponding variable \( s_v \) and linear inequalities. Specifically, in the case of type (ii), we have \( s_v \leq \sum_{j \in \Phi(e)} z_j + s_{e.u} \) for each \( e \in E \) s.t. e.v = v, and for its child node \( v' \) and edge \( e' \) between \( v \) and \( v' \), \( s_{v'} \leq \sum_{j \in \Phi(e')} z_j + s_{v} \). These inequalities are redundant since we can obtain equivalent inequalities by concatenating them: \( s_{v'} \leq \sum_{j \in \Phi(e')} z_j + \sum_{j \in \Phi(e)} z_j + s_{e.u} \) for each \( e \in E \) s.t. e.v = v, where \( s_v \) is removed.

Based on the observation above, we propose a simple reduction heuristics removing nodes of type (i) and (ii). More precisely, given an NZDD \( G = (V, E) \), the heuristics outputs an NZDD \( G' = (V', E') \) such that \( L(G) = L(G') \) and \( G' \) does not contain nodes of type (i) or (ii). The heuristics can be implemented in \( O(|V'| + |E'| + \sum_{e \in E'} |\Phi(e)|) \) time by going through nodes of the input NZDD \( G \) in the topological order from the leaf to the root and in the reverse order, respectively. The details of the heuristics is given in Appendix.

8 Experiments

We show preliminary experimental results on synthetic and real large data sets\(^2\). We performed mixed integer programming and 1-norm regularized soft margin optimization. Our experiments are conducted on a server with 2.60

---

\(^2\) Codes are available at [https://bitbucket.org/kohei_hatano/codes_extended_formulation_nzdd/](https://bitbucket.org/kohei_hatano/codes_extended_formulation_nzdd/)
8.1 Mixed Integer programming on synthetic datasets

First, we apply our extended formulation (1) to mixed integer programming tasks over synthetic data sets. The problems are defined as the linear optimization with \( n \) variables and \( m \) linear constraints of the form \( Ax \geq b \), where (i) each row of \( A \) has \( k \) entries of 1 and others are 0s and nonzero entries are chosen randomly without repetition (ii) coefficients \( a_i \) of linear objective \( \sum_{i=1}^{n} a_i x_i \) is chosen from 1,..,100 randomly, and (iii) first \( l \) variables take binary values in \( \{0, 1\} \) and others take real values in \( [0, 1] \). In our experiments, we fix \( n = 25 \), \( k = 10 \), \( l = 12 \) and \( m \in \{4 \times 10^5, 8 \times 10^5,..., 20 \times 10^5\} \). We apply the Gurobi optimizer directly to the problem denoted as \textit{mip} and the solver with pre-processing the problem by our extended formulation (denoted as \textit{nzdd_mip}, respectively. The results are summarized in Figure 5. Our method consistently improves computation time for these datasets. This makes sense since it can be shown that when \( m = O(n^k) \) there exists an NZDD of size \( O(nk) \) representing the constraint matrix.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5.png}
\caption{The comparison for synthetic datasets of a MIP problem. The horizontal axis represents the number of constraints of the original problem.}
\end{figure}

8.2 1-norm soft margin optimization on real data sets

Next, we apply our methods on the task of the 1-norm soft margin optimization. This problem is a standard optimization problem in the machine learning literature, categorized as LP, for finding sparse linear classifiers given labeled instances. We compare the following methods using a naive LP solver.

1. a naive LP solver (denoted as \textit{naive}),
2. LPBoost [Demiriz et al. [2002], denoted as \textit{lpb}), a column generation-based method,
3. ERLPBoost [Warmuth et al. [2008], denoted as \textit{erlp}), a modification of LPBoost with a non-trivial iteration bound.
4. a naive LP solver (denoted as \textit{nzdd_naive}),
5. Algorithm 1 (denoted as \textit{nzdd_lpb}),
6. Algorithm 2 (denoted as \textit{nzdd_erlpb}).

Methods 1–3 solves the original soft margin optimization problem (4), while 4–6 solves the problem (6). We measure its computation time (CPU time) and maximum memory consumption, respectively, and compare their averages over parameters. Further, we performed 5-fold cross validation to check the test error rates of our methods on real data sets. Table 2 shows that our formulation (6) is competitive with the original soft margin optimization.

Methods 1 and 2 solves the original soft margin optimization problem (4), while 3 and 4 solves the problem (6). We measure its computation time (CPU time) and maximum memory consumption, respectively, and compare their averages over parameters. Further, we performed 5-fold cross validation to check the test error rates of our methods on real data sets. Table 2 shows that our formulation (6) is competitive with the original soft margin optimization.

We compare methods on some real data sets in the libsvm datasets Chang and Lin [2011] to see the effectiveness of our approach in practice. Generally, the datasets contain huge samples (\( m \) varies from \( 3 \times 10^4 \) to \( 10^7 \)) with a relatively small size of features (\( n \) varies from \( 20 \) to \( 10^5 \)). The features of instances of each dataset is transformed into binary values. Note that these results exclude NZDD construction times since the compression takes around 1 second, except for the HIGGS dataset (around 13 seconds). Furthermore, the construction time of NZDDs can be neglected in the following reason: We often need to try multiple choices of the hyperparameters (\( \nu \) in our case) and solve the optimization problem for each set of choices. But once we construct an NZDD, we can be re-use it for different values of hyperparameters without reconstructing NZDDs.
Extended formulations via decision diagrams

1-norm soft margin optimization on synthetic datasets  We show experimental results for the 1-norm soft margin optimization on synthetic datasets. We use a class of synthetic datasets that have small NZDD representations when the samples are large. First, we choose $m$ instances in $\{0, 1\}^n$ uniformly randomly without repetition. Then we consider the following linear threshold function $f(x) = \text{sign}(\sum_{j=1}^{k} x_j - r + 1/2)$, where $k$ and $r$ are positive integers such that $1 \leq r \leq k \leq n$. That is, $f(x) = 1$ if and only if at least $r$ of the first $k$ components are 1. Each label of the instance $x \in \{0, 1\}^n$ is labeled by $f(x)$. It can be easily shown that the whole labeled $2^n$ instances of $f$ is represented by an NZDD (or ZDD) of size $O(kr)$, which is exponentially small w.r.t. the sample size $m = 2^n$. We fix $n = 20$, $k = 10$ and $r = 5$. Then we use $m \in \{1 \times 10^5, 2 \times 10^5, \ldots, 10^6\}$.

The results are given in Figure 8. As expected, our methods are significantly faster than comparators. Generally, our methods perform better than the standard counterparts. In particular, nzdd_lpb improves efficiency at least 10 to 100 times over others. Similar results are obtained for maximum memory consumption.
Table 2: Test error rates for real datasets

| Data sets | lpb  | nzdd_naive | nzdd_erlp |
|-----------|------|------------|-----------|
| a9a       | 0.174| 0.159      | 0.157     |
| art-100000| 0.000| 0.0004     | 0.004     |
| real-sim  | 0.179| 0.169      | 0.532     |
| w8a       | 0.030| 0.030      | 0.029     |

9 Conclusion

We proposed a generic algorithm of constructing an NZDD-based extended formulation for any given set of linear constraints with integer constraints as well as specific algorithms for the 1-norm soft margin optimization and practical heuristics for constructing NZDDs. Our algorithms improve time/space efficiency on artificial and real datasets, especially when the datasets have concise NZDD representations.

10 Acknowledgements

We thank Mohammad Amin Mansouri for valuable discussions and initial development. This work was supported by JSPS KAKENHI Grant Numbers JP19H014174, JP19H04067, JP20H05967, and JP22H03649, respectively.

References

N. Abe, J. Takeuchi, and M. K. Warmuth. Polynomial learnability of stochastic rules with respect to the KL-divergence and quadratic distance. *IEICE Transactions on Information and Systems*, E84-D(3):299–316, 2001.

J. L. Bentley and R. Sedgewick. Fast algorithms for sorting and searching strings. In *Proceedings of the eighth annual ACM-SIAM symposium on Discrete algorithms (SODA ’97)*, pages 360–369, 1997.

D. Bergman, W. J. van Hoeve, and J. N. Hooker. Manipulating MDD Relaxations for Combinatorial Optimization. In *Proceedings of the 8th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 2011)*, volume 6697 of LNCS, pages 20–35, 2011.

D. Bergman, A. A. Cire, W.-J. van Hoeve, and J. Hooker. *Decision Diagrams for Optimization*. Springer Cham, 2016.

R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, C-35 (8):677–691, aug 1986.

M. P. Castro, A. A. Cire, and J. C. Beck. An MDD-Based Lagrangian Approach to the Multicommodity Pickup-and-Delivery TSP. *INFORMS Journal on Computing*, 32(2):263–278, sep 2019. ISSN 15265528. doi:10.1287/IJOC.2018.0881.

C.-C. Chang and C.-J. Lin. LIBSVM:a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(27):1–27, 2011.

M. Conforti, G. Cornuéjols, and G. Zambelli. Extended formulations in combinatorial optimization. *4OR*, 8(1):1–48, 2010.

A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear Programming Boosting via Column Generation. *Machine Learning*, 46(1-3):225–254, 2002.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7):2121–2159, 2011.

S. Fiorini, T. Huynh, and S. Weltge. Strengthening convex relaxations of 0/1-sets using Boolean formulas. *Mathematical Programming*, 190(1):467–482, 2021. ISSN 1436-4646. doi:10.1007/s10107-020-01542-w URL https://doi.org/10.1007/s10107-020-01542-w

Y. Freund and R. E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

T. Fujita, K. Hatano, and E. Takimoto. Boosting over non-deterministic ZDDs. *Theoretical Computer Science*, 806: 81–89, 2020.
K. Goto, H. Bannai, S. Inenaga, and M. Takeda. Fast q-gram mining on SLP compressed strings. Journal of Discrete Algorithms, 18:89–99, 2013.

D. Hermelin, G. M. Landau, S. Landau, and O. Weimann. A Unified Algorithm for Accelerating Edit-Distance Computation via Text-Compression. In Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science (STACS’09), volume LIPIcs, pages 529–540, 2009.

T. Inoue, K. Takano, T. Watanabe, J. Kawahara, R. Yoshinaka, A. Kishimoto, K. Tsuda, S.-i. Minato, and Y. Hayashi. Distribution Loss Minimization With Guaranteed Error Bound. IEEE Transactions on Smart Grid, 5(1):102–111, 2014.

T. Jiang and B. Ravikumar. Minimal NFA Problems are Hard. SIAM Journal on Computing, 22(6):1117–1141, 1993.

D. E. Knuth. The Art of Computer Programming, Volume 4A, Combinatorial Algorithms, Part 1. Addison-Wesley Professional, 2011.

Y. Lifshits. Processing Compressed Texts: A Tractability Border. In Proceedings of the 18th Annual Symposium on Combinatorial Pattern Matching (CPM’07), volume 4580 of LNCS, pages 228–240, 2007.

M. Lohrey. Algorithmics on SLP-compressed strings: A survey. Groups - Complexity - Cryptology, 4(2):241–299, 2012.

S.-i. Minato. Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems. In Proceedings of the 30th international Design Automation Conference (DAC’93), pages 272–277, 1993.

S.-i. Minato. Power of Enumeration — Recent Topics on BDD/ZDD-Based Techniques for Discrete Structure Manipulation. IEICE Transactions on Information and Systems, E100.D(8):1556–1562, 2017.

S.-i. Minato and T. Uno. Frequentness-Transition Queries for Distinctive Pattern Mining from Time-Segmented Databases. In Proceedings of the 2010 SIAM International Conference on Data Mining (SDM), pages 339–349, 2010.

S.-i. Minato, T. Uno, and H. Arimura. LCM over ZBDDs: Fast Generation of Very Large-Scale Frequent Itemsets Using a Compact Graph-Based Representation. In Proceedings of the 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD’08), volume 5012 of LNAI, pages 234–246, 2008.

D. R. Morrison, E. C. Sewell, and S. H. Jacobson. Solving the Pricing Problem in a Branch-and-Price Algorithm for Graph Coloring Using Zero-Suppressed Binary Decision Diagrams. INFORMS Journal on Computing, 28(1):67–82, Jan 2016. ISSN 15265528. doi:10.1287/INOC.2015.0667.

M. Nishino, N. Yasuda, S.-i. Minato, and M. Nagata. Accelerating Graph Adjacency Matrix Multiplications with Adjacency Forest. In Proceedings of the 2014 SIAM International Conference on Data Mining (SDM 14), pages 1073–1081, 2014.

R. Raina, A. Madhavan, and A. Y. Ng. Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th annual international conference on machine learning (ICML’09), pages 873–880, 2009.

G. Rätsch and M. K. Warmuth. Efficient Margin Maximizing with Boosting. Journal of Machine Learning Research, 6:2131–2152, 2005.

W. Rytter. Grammar Compression, LZ-Encodings, and String Algorithms with Implicit Input. In Proceedings of the 31st International Colloquium of Automata, Languages and Programming (ICALP’04), volume 3142 of LNCS, pages 15–27, Berlin, Heidelberg, 2004.

Y. Tabei, H. Saigo, Y. Yamanishi, and S. J. Puglisi. Scalable Partial Least Squares Regression on Grammar-Compressed Data Matrices. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’16), pages 1875–1884, 2016.

T. Toda. Fast Compression of Large-Scale Hypergraphs for Solving Combinatorial Problems. In Proceedings of the 16th International Conference on Discovery Science (DS’13), volume 8140 of LNAI, pages 281–293, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-40897-7.

T. Toda. ZCOMP: Fast Compression of Hypergraphs into ZDDs, 2015. URL http://www.sd.is.uec.ac.jp/toda/code/zcomp.html.

M. Warmuth, K. Glocer, and G. Rätsch. Boosting Algorithms for Maximizing the Soft Margin. In Advances in Neural Information Processing Systems 20 (NIPS 2007), pages 1585–1592, 2007.

M. Warmuth, K. Glocer, and S. V. N. Vishwanathan. Entropy Regularized LPBoost. In Proceedings of the 19th International Conference on Algorithmic Learning Theory (ALT’08), pages 256–271, 2008.

M. Yannakakis. Expressing combinatorial optimization problems by Linear Programs. Journal of Computer and System Sciences, 43(3):441–466, 1991. ISSN 0022-0000. doi:https://doi.org/10.1016/0022-0000(91)90024-Y.
Algorithm 3 Reducing procedure

Input: NZDD $G = (V, E, \Phi, \Sigma)$

1. For each $u \in V$ in a topological order (from leaf to root) and for each child node $v$ of $u$,
   (a) If indegree of $v$ is one,
      i. for the incoming edge $e$ from $u$ to $v$, each child node $v'$ of $v$ and each outgoing edge $e'$ from $v$ to $v'$, add a new edge $e''$ from node $u$ to $v'$ and set $\Phi(e'') = \Phi(e) \cup \Phi(e')$.
   (b) Remove the incoming edge $e$ and all outgoing edges $e'$.
2. For each $v \in V$ in a topological order (from root to leaf) and for each parent node $u$ of $v$,
   (a) If outdegree of $u$ is one,
      i. for the outgoing edge $e$ of $u$, each parent node $u'$ of $u$ and each outgoing edge $e'$ from $u'$ to $u$, add a new edge $e''$ from node $u'$ to $v$ and set $\Phi(e'') = \Phi(e) \cup \Phi(e')$.
   (b) Remove the outgoing edge $e$ and all incoming edges $e'$.
3. Remove all nodes with no incoming and outgoing edges from $V$ and output the resulting DAG $G' = (V', E')$.

A Details of heuristics for constructing NZDDs

A pseudo-code of the heuristics is given in Algorithm 3. Algorithm 3 consists of two phases. In the first phase, it traverses nodes in the topological order (from the leaf to the root), and for each node $v$ with one incoming edge $e$, it contracts $v$ with its parent node $u$ and $u$ inherits the edges $e'$ from $v$. Label sets $\Phi(e)$ and $\Phi(e')$ are also merged. The first phase can be implemented in $O(|V| + |E|)$ time, by using an adjacency list maintaining children of each node and lists of label sets for each edge. In the second phase, it does a similar procedure to simplify nodes with single outgoing edges. To perform the second phase efficiently, we need to re-organize lists of label sets before the second phase starts. This is because the lists of label sets could form DAGS after the first phase ends, which makes performing the second phase inefficient. Then, the second phase can be implemented in $O(|V'| + |E'| + \sum_{e \in E'} |\Phi(e)|)$ time.

B Details of experiments

Preprocessing of datasets The datasets for the 1-norm regularized soft margin optimization are obtained from the libsvm datasets. Some of them contain real-valued features. We convert them to binary ones by rounding them using 0.5 as a threshold.

NZDD construction time and summary of datasets Computation times for constructing NZDDs for the MIP synthetic datasets and synthetic and real datasets of the 1-norm regularized soft margin optimization are summarized in Table 3 and 4, respectively. Note that the NZDD construction time is not costly and negligible in general because once we construct NZDDs, we can re-use those NZDDs for solving optimization problems with different objective functions or hyperparameters such as $\nu$ in the soft margin optimization.

Table 3: Computation time (seconds) of NZDD construction for synthetic MIP datasets, described in Table 4. All datasets have $n = 25$ features and each instance have $k = 10$ nonzero components.

| $m$     | $t_{\text{zcomp}}$ | $t_{\text{reducing\ procedure}}$ | $t_{\text{total}}$ |
|---------|---------------------|-----------------------------------|-------------------|
| $4 \times 10^5$ | 0.39                | 1.02                              | 1.41              |
| $8 \times 10^5$ | 0.76                | 1.38                              | 2.14              |
| $12 \times 10^5$ | 1.08                | 1.41                              | 2.49              |
| $16 \times 10^5$ | 1.36                | 1.10                              | 2.46              |
| $20 \times 10^5$ | 1.60                | 0.33                              | 1.93              |

Table 8 summarizes the size of each problem for soft margin optimization. As this table shows, the extended formulations have fewer variables and constraints. This is not surprising since the extended formulation has $O(n + |V| + |E|)$ variables and $O(|E|)$ constraints, while the original formulation (4) has $O(n + m)$ variables and $O(m)$ constraints.
Table 4: Summary of synthetic datasets for MIP. The term “original” and “extended” mean the original and the extended formulations, respectively.

| Data size | NZDD size | Variables | Constraints |
|-----------|-----------|-----------|-------------|
| \( n \) | \( m \) | \(|V|\) | \(|E|\) | Original | Extended | Original | Extended |
| 25       | \( 4 \times 10^5 \) | 13,321    | 177,356     | 25       | 13,344   | 400,000   | 177,356   |
| 25       | \( 8 \times 10^5 \) | 17,771    | 241,488     | 25       | 17,794   | 800,000   | 241,488   |
| 25       | \( 12 \times 10^5 \) | 19,348    | 249,749     | 25       | 19,371   | 1,200,000 | 249,749   |
| 25       | \( 16 \times 10^5 \) | 17,819    | 204,723     | 25       | 17,842   | 1,600,000 | 204,723   |
| 25       | \( 20 \times 10^5 \) | 6,161     | 55,555      | 25       | 6,184    | 2,000,000 | 55,555    |

Table 5: Computation times (seconds) for NZDD construction for the synthetic datasets, described in Table 4. All data have \( n = 20 \) features.

| \( m \) | \( z\text{comp} \) | Reducing procedure | Total |
|---------|----------------|-------------------|-------|
| \( 1 \times 10^5 \) | 0.08 | 0.14 | 0.22 |
| \( 2 \times 10^5 \) | 0.15 | 0.14 | 0.29 |
| \( 3 \times 10^5 \) | 0.25 | 0.12 | 0.37 |
| \( 4 \times 10^5 \) | 0.31 | 0.08 | 0.39 |
| \( 5 \times 10^5 \) | 0.39 | 0.05 | 0.44 |
| \( 6 \times 10^5 \) | 0.46 | 0.05 | 0.51 |
| \( 7 \times 10^5 \) | 0.53 | 0.03 | 0.56 |
| \( 8 \times 10^5 \) | 0.60 | 0.02 | 0.62 |
| \( 9 \times 10^5 \) | 0.68 | 0.01 | 0.69 |
| \( 10 \times 10^5 \) | 0.72 | 0.00 | 0.72 |

Table 6: Computation times (seconds) for constructing NZDDs for real datasets of the soft margin optimization.

| dataset | \( z\text{comp} \) | Reducing procedure | Total |
|---------|----------------|-------------------|-------|
| a9a     | 0.04           | 0.20              | 0.24  |
| art-100000 | 0.10           | 0.43              | 0.53  |
| real-sim| 0.24           | 0.99              | 1.23  |
| w8a     | 0.27           | 4.20              | 4.47  |
| HIGGS   | 13.13          | 0.00              | 13.13 |

Table 7: Summary of synthetic datasets for the soft margin optimization. The term “original” and “extended” mean the original and the extended formulations, respectively.

| Data size | NZDD size | Variables | Constraints |
|-----------|-----------|-----------|-------------|
| \( n \) | \( m \) | \(|V|\) | \(|E|\) | Original | Extended | Original | Extended |
| 20       | \( 1 \times 10^5 \) | 4,202     | 55,147     | 100,021 | 59,370   | 200,001   | 110,297   |
| 20       | \( 2 \times 10^5 \) | 6,663     | 82,810     | 200,021 | 89,494   | 400,001   | 165,623   |
| 20       | \( 3 \times 10^5 \) | 11,022    | 103,323    | 300,021 | 114,366  | 600,001   | 206,649   |
| 20       | \( 4 \times 10^5 \) | 13,596    | 120,994    | 400,021 | 134,611  | 800,001   | 241,991   |
| 20       | \( 5 \times 10^5 \) | 12,565    | 128,670    | 500,021 | 141,256  | 1,000,001 | 257,343   |
| 20       | \( 6 \times 10^5 \) | 13,796    | 127,139    | 600,021 | 140,956  | 1,200,001 | 254,281   |
| 20       | \( 7 \times 10^5 \) | 13,513    | 114,471    | 700,021 | 128,005  | 1,400,001 | 228,945   |
| 20       | \( 8 \times 10^5 \) | 9,569     | 95,454     | 800,021 | 105,044  | 1,600,001 | 190,911   |
| 20       | \( 9 \times 10^5 \) | 6,725     | 75,318     | 900,021 | 82,064   | 1,800,001 | 150,639   |
| 20       | \( 10 \times 10^5 \) | 3,914     | 40,226     | 1,000,021 | 44,161 | 2,000,001 | 80,455   |
Table 8: Summary of sizes of the real datasets of the soft margin optimization. The term “original” and “extended” mean the original and the extended formulations (4) and (6), respectively.

| dataset | Data size | NZDD size | Variables | Constraints |
|---------|-----------|------------|------------|--------------|
|         | $n$       | $m$       | $|V|$       | $|E|$         | Original | Extended | Original | Extended |
| a9a     | 123       | 32,561    | 775        | 20,657       | 32,685   | 21,556   | 65,123    | 41,317    |
| art-100000 | 20    | 100,000   | 4,202      | 55,163       | 100,021  | 59,386   | 200,001   | 110,329   |
| real-sim | 20,955   | 72,309    | 38         | 7,922        | 93,265   | 28,916   | 144,619   | 15,847    |
| w8a     | 300       | 49,749    | 209        | 34,066       | 50,050   | 34,576   | 99,499    | 68,135    |
| HIGGS   | 28       | 11,000,000| 151        | 989          | 11,000,029| 1,169    | 22,000,001| 1,981     |