Extended formulations via decision diagrams

Yuta Kurokawa[1] Ryotaro Mitsuboshi[1,2] Haruki Hamasaki[1]
Kohei Hatano[1,2] Eiji Takimoto[1] Holakou Rahmanian[3]

[1]Kyushu University

[2]RIKEN AIP

[3]Amazon

[COCOON 2023]

Background and motivation

Consider the optimization problems of the form:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n,$$

where $A \in \{0,1\}^{m \times n}$, $\boldsymbol{b} \in \{0,1\}^m$, $f : \mathbb{R}^n \to \mathbb{R}$ is an arbitrary function, and $\mathcal{X}$ represents other constraints, e.g., discrete or semidefinite, etc.

Ex. LP, QP, IP, and SDP with binary coefficients.

Parallel to the development of computers, m becomes enormous.

Our goal

Generate an equivalent formulation with a smaller problem size.

Background and motivation

Consider the optimization problems of the form:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n,$$

where $A \in \{0,1\}^{m \times n}$, $\boldsymbol{b} \in \{0,1\}^m$, $f : \mathbb{R}^n \to \mathbb{R}$ is an arbitrary function, and $\mathcal{X}$ represents other constraints, e.g., discrete or semidefinite, etc.

Ex. LP, QP, IP, and SDP with binary coefficients.
Parallel to the development of computers, m becomes enormous.

Our goal

Generate an equivalent formulation with a smaller problem size.

Background and motivation

Consider the optimization problems of the form:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n,$$

where $A \in \{0,1\}^{m \times n}$, $\boldsymbol{b} \in \{0,1\}^m$, $f : \mathbb{R}^n \to \mathbb{R}$ is an arbitrary function, and $\mathcal{X}$ represents other constraints, e.g., discrete or semidefinite, etc.

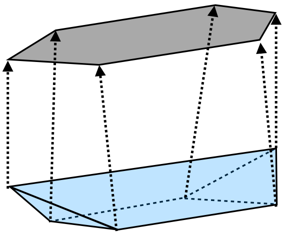
Ex. LP, QP, IP, and SDP with binary coefficients.
Parallel to the development of computers, m becomes enormous.

Our goal

Generate an equivalent formulation with a smaller problem size.

Extended formulation

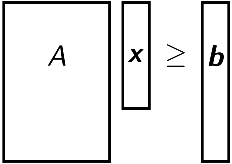
A class of methods that reduces the number of constraints by slightly increasing the number of variables.



- Original (2-dim. space)
  - 2 variables,
  - 6 constraints.
- Ext. form. (3-dim. space)
  - 3 variables,
  - 5 constraints.

Sketch

Original formulation



$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Construct NZDD representation

Each path corresponds to a constraint in $A\boldsymbol{x} \geq \boldsymbol{b}$.

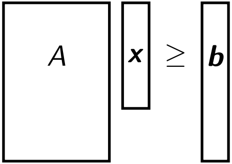
Establish Extended formulation

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \geq \boldsymbol{b}',$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Hopefully, the number of constraints significantly decreases, while the number of variables slightly increases.

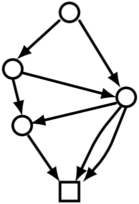
Sketch

Original formulation



$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Construct NZDD representation



Each path corresponds to a constraint in $A\boldsymbol{x} \geq \boldsymbol{b}$.

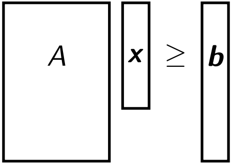
Establish Extended formulation

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \geq \boldsymbol{b}',$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Hopefully, the number of constraints significantly decreases, while the number of variables slightly increases.

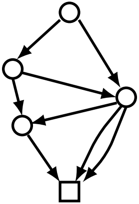
Sketch

Original formulation



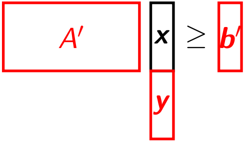
$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A\boldsymbol{x} \geq \boldsymbol{b},$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Construct NZDD representation



Each path corresponds to a constraint in $A\boldsymbol{x} \geq \boldsymbol{b}$.

Establish Extended formulation



$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{s.t. } A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \geq \boldsymbol{b}',$$
$$\boldsymbol{x} \subset \mathcal{X}$$

Hopefully, the number of constraints significantly decreases, while the number of variables slightly increases.

Related work and our contribution

- Bergman et al. represent a feasible region by ZDD and reduce a discrete optimization problem to the shortest path problem over the ZDD[Bregman et al., '16].
- Fujita et al. emulate AdaBoostV[Rätsch et al., '05] over an NZDD[Fujita et al., '20].

Our contribution

- Generates an extended formulation algorithmically.

  Input: optimization problem with lineaer constraints.

  Output: an equiv. optimization problem with fewer constraints.

- our method can solve the continuous optimization problem.

Non-deterministic Zero-suppressed Decision Diagram

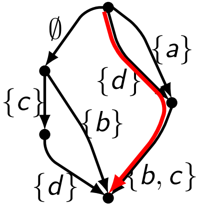
A data structure that represents a subset family.

Definition (NZDD [Fujita et al., '13])

An NZDD G is a quadruple (V, E, Σ, ψ) where

- (V, E) is a DAG with a single root and single leaf,
- Σ is the ground set,
- $\psi : E \to 2^{\Sigma}$ labels each edge $e \in E$ with a subset $\psi(e) \subset \Sigma$ s.t. $\forall$ root-leaf path $P \subset E$, $\forall e_1, e_2 \in P, \psi(e_1) \cap \psi(e_2) = \emptyset$.
- G represents the subset family $\{\bigcup_{e \in P} \psi(e) \mid P$ is a root-leaf path$\}$

Ex.



The left NZDD represents the family $\{\{a, b, c\}, \{b\}, \{b, c, d\}, \{c, d\}\}$.
The red path represents $\{b, c, d\}$.

Our approach (1/3)

Convert the constraints to the subset family of indices.

We have linear constraints $A\boldsymbol{x} \geq \boldsymbol{b}$, where $A \in \{0,1\}^{m \times n}$ and $\boldsymbol{b} \in \{0,1\}^m$.
Let $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$ be the ith row of $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Pick a constraint $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$.
   Ex. $x_1 + x_2 + x_4 \geq 1$.
2. Construct a vector $\boldsymbol{c}_i := \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix} \in \{0,1\}^{n+1}$.
   Ex. $\boldsymbol{c}_i = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix}$.
3. Collect the non-zero indices
   $\mathrm{Ix}(\boldsymbol{c}) = \{j \in [n+1] \mid c_j = 1\}$.
   Ex. $\mathrm{Ix}(\boldsymbol{c}_i) = \{1,2,4,5\}$.

At this point, we have $\mathcal{C} = \{\mathrm{Ix}(\boldsymbol{c}_i) \mid \boldsymbol{c}_i = \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix}, i \in [m]\}$.
Now we construct an NZDD that represents $\mathcal{C}$.

Our approach (1/3)

Convert the constraints to the subset family of indices.

We have linear constraints $A\boldsymbol{x} \geq \boldsymbol{b}$, where $A \in \{0,1\}^{m \times n}$ and $\boldsymbol{b} \in \{0,1\}^m$.
Let $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$ be the ith row of $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Pick a constraint $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$.
   Ex. $x_1 + x_2 + x_4 \geq 1$.
2. Construct a vector $\boldsymbol{c}_i := \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix} \in \{0,1\}^{n+1}$.
   Ex. $\boldsymbol{c}_i = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix}$.
3. Collect the non-zero indices
   $\mathrm{Ix}(\boldsymbol{c}) = \{j \in [n+1] \mid c_j = 1\}$.
   Ex. $\mathrm{Ix}(\boldsymbol{c}_i) = \{1, 2, 4, 5\}$.

At this point, we have $\mathcal{C} = \{\mathrm{Ix}(\boldsymbol{c}_i) \mid \boldsymbol{c}_i = \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix}, i \in [m]\}$.
Now we construct an NZDD that represents $\mathcal{C}$.

Our approach (1/3)

Convert the constraints to the subset family of indices.

We have linear constraints $A\boldsymbol{x} \geq \boldsymbol{b}$, where $A \in \{0,1\}^{m \times n}$ and $\boldsymbol{b} \in \{0,1\}^m$.
Let $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$ be the ith row of $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Pick a constraint $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$.
   Ex. $x_1 + x_2 + x_4 \geq 1$.
2. Construct a vector $\boldsymbol{c}_i := \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix} \in \{0,1\}^{n+1}$.
   Ex. $\boldsymbol{c}_i = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix}$.
3. Collect the non-zero indices
   $\mathrm{Ix}(\boldsymbol{c}) = \{j \in [n+1] \mid c_j = 1\}$.
   Ex. $\mathrm{Ix}(\boldsymbol{c}_i) = \{1,2,4,5\}$.

At this point, we have $\mathcal{C} = \{\mathrm{Ix}(\boldsymbol{c}_i) \mid \boldsymbol{c}_i = \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix}, i \in [m]\}$.
Now we construct an NZDD that represents $\mathcal{C}$.

Our approach (1/3)

Convert the constraints to the subset family of indices.

We have linear constraints $A\boldsymbol{x} \geq \boldsymbol{b}$, where $A \in \{0,1\}^{m \times n}$ and $\boldsymbol{b} \in \{0,1\}^m$.
Let $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$ be the ith row of $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Pick a constraint $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$.
   Ex. $x_1 + x_2 + x_4 \geq 1$.
2. Construct a vector $\boldsymbol{c}_i := \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix} \in \{0,1\}^{n+1}$.
   Ex. $\boldsymbol{c}_i = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix}$.
3. Collect the non-zero indices
   $\mathrm{Ix}(\boldsymbol{c}) = \{ j \in [n+1] \mid c_j = 1 \}$.
   Ex. $\mathrm{Ix}(\boldsymbol{c}_i) = \{1,2,4,5\}$.

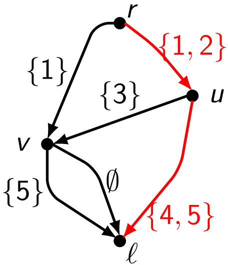
At this point, we have $\mathcal{C} = \{ \mathrm{Ix}(\boldsymbol{c}_i) \mid \boldsymbol{c}_i = \begin{bmatrix} \boldsymbol{a}_i & b_i \end{bmatrix}, i \in [m] \}$.
Now we construct an NZDD that represents $\mathcal{C}$.

Our approach (1/3)

Convert the constraints to the subset family of indices.

We have linear constraints $A\boldsymbol{x} \geq \boldsymbol{b}$, where $A \in \{0,1\}^{m\times n}$ and $\boldsymbol{b} \in \{0,1\}^m$.
Let $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$ be the ith row of $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Pick a constraint $\boldsymbol{a}_i \cdot \boldsymbol{x} \geq b_i$.
   Ex. $x_1 + x_2 + x_4 \geq 1$.
2. Construct a vector $\boldsymbol{c}_i := \begin{bmatrix}\boldsymbol{a}_i & b_i\end{bmatrix} \in \{0,1\}^{n+1}$.
   Ex. $\boldsymbol{c}_i = \begin{bmatrix}1 & 1 & 0 & 1 & 1\end{bmatrix}$.
3. Collect the non-zero indices
   $\mathrm{Ix}(\boldsymbol{c}) = \{j \in [n+1] \mid c_j = 1\}$.
   Ex. $\mathrm{Ix}(\boldsymbol{c}_i) = \{1,2,4,5\}$.



At this point, we have $\mathcal{C} = \{\mathrm{Ix}(\boldsymbol{c}_i) \mid \boldsymbol{c}_i = \begin{bmatrix}\boldsymbol{a}_i & b_i\end{bmatrix}, i \in [m]\}$.
Now we construct an NZDD that represents $\mathcal{C}$.

Our approach (2/3)

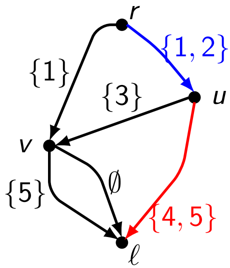
Establish an extended formulation from an NZDD.

We can reproduce the original constraints $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. Define variables $\{s_v \mid v \in V\}$.
2. For each edge $(p, q) \in E \subset V \times V$, define a constraint $s_p + \sum_{i \in \psi((p,q))} \mathrm{sgn}(i) x_i \geq s_q$, where $\mathrm{sgn} : [n+1] \to \{\pm 1\}$ is the function s.t. $\mathrm{sgn}(i) = 1 \iff i \neq n+1$.

Ex.

$$\underbrace{\left[\begin{array}{ccccc|cccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & -1 \end{array}\right]}_{=:A'} \begin{bmatrix} x_1 \\ \vdots \\ x_5 \\ s_r \\ s_u \\ s_v \\ s_\ell \end{bmatrix} \geq \mathbf{0}$$



Ex.

$s_r + x_1 + x_2 \geq s_u$,

$s_u + x_4 - x_5 \geq s_\ell$.

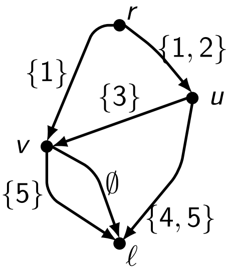
Consider the opposite direction.

$$\text{If } \boldsymbol{x} \text{ satisfies } A\boldsymbol{x} \geq \boldsymbol{b}, \qquad \exists \boldsymbol{s} \in \mathbb{R}_+^V \text{ s.t. } A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{bmatrix} \geq \boldsymbol{0}.$$

Let $\boldsymbol{x}$ be a vector satisfying original constraints $A\boldsymbol{x} \geq \boldsymbol{b}$.

1. For each edge $e \in E$, assign weight $\sum_{i \in \psi(e)} \mathrm{sgn}(i) x_i$.
2. For each vertex $v \in V$, set s_v to the shortest path length from root to v.
3. The resulting vector $\boldsymbol{s} \in \mathbb{R}_+^V$ satisfies

$$A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{bmatrix} \geq \boldsymbol{0}$$



Satisfying $A\boldsymbol{x} \geq \boldsymbol{b} \iff$ the corresponding path length ≥ 0

Main result

Theorem

- *Let $A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{bmatrix} \geq \mathbf{0}'$ be the constraints constructed from $A\boldsymbol{x} \geq \boldsymbol{b}$. Then, the constraints represent the same feasible region in terms of $\boldsymbol{x}$.*
- *One can find the optimal solution to the original problem from the compressed one.*

Let $G = (V, E, \Sigma, \Psi)$ be the NZDD for the compressed problem.

- # of constraints: $O(|E|)$,
- # of variables: $O(n + |V|)$.

Thus, constructing a small NZDD in the sense the number of edges highly reduces the constraints.

Main result

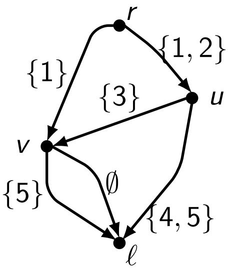
Theorem

- *Let* $A' \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{s} \end{bmatrix} \geq \mathbf{0}'$ *be the constraints constructed from* $A\boldsymbol{x} \geq \boldsymbol{b}$*. Then, the constraints represent the same feasible region in terms of* $\boldsymbol{x}$*.*
- *One can find the optimal solution to the original problem from the compressed one.*

Let $G = (V, E, \Sigma, \Psi)$ be the NZDD for the compressed problem.

- # of constraints: $O(|E|)$,
- # of variables: $O(n + |V|)$.

Thus, constructing a small NZDD in the sense the number of edges highly reduces the constraints.



Construction of NZDD

- No good algorithms known to construct a concise NZDD from a given subset family.
- `ZCOMP`[1] [Toda, '15], a tool for constructing ZDDs is available.

Currently, we use the following procedure:

1. Construct a ZDD by `ZCOMP`.
2. Contract edges to remove all nodes with 1 indegree or 1 outdegree. (Heuristics)

[1] http://www.sd.is.uec.ac.jp/toda/code/zcomp.html

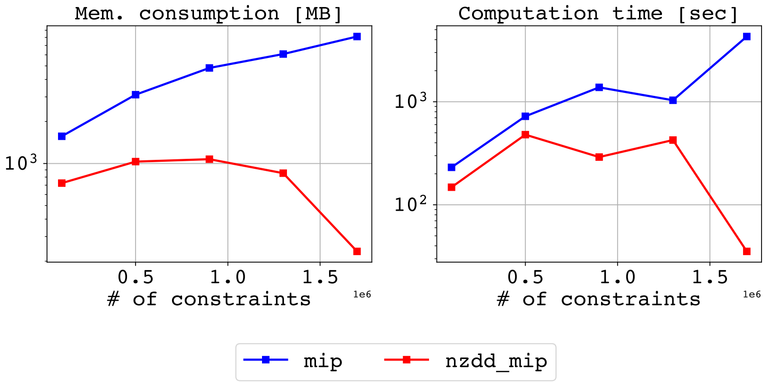
Experiment: NZDD construction

- # of variables $n = 25$.
- Each row of the linear constraints has 10 non-zero entries.

m	ZCOMP (sec.)	Heuristics (sec.)	Total (sec.)
4×10^5	0.39	1.02	1.41
8×10^5	0.76	1.38	2.14
12×10^5	1.08	1.41	2.49
16×10^5	1.36	1.10	2.46
20×10^5	1.60	0.33	1.93

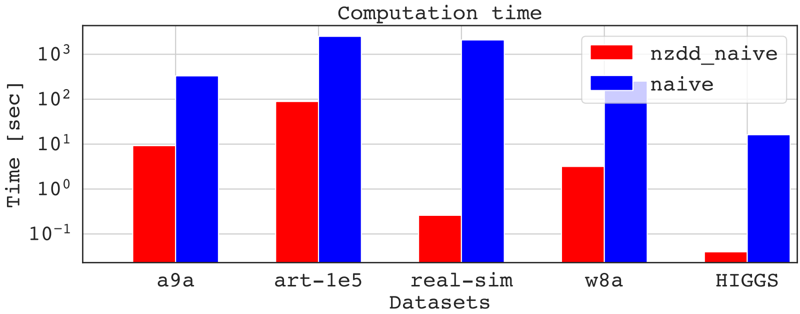
Experiment: Artificial dataset

- The number of variables $n = 25$, in which 13 take continuous values and 12 take discrete ones.
- The number of constraints $m = k \times 10^5$, where $k \in \{4, 8, 12, 16, 20\}$.



Experiment: Real dataset

- Solves L_1-norm regularized soft margin optimization problem.
- The # of constraints equals to the # of variables.
  - Cannot reduce the # of constraints by our extended formulation.
  - Solves an approximate problem.
  - We verified its effectiveness by measuring test loss.
- The dataset is from `LIBSVM`[2].



[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Conclusion

- Proposed a general algorithm to generate an extended formulation from a given linear constraints.
- Experimental results demonstrate its effectiveness.
- Sometimes the construction time for an NZDD is problematic.
  - Are there any effective construction for NZDDs?