Revisiting Pseudo-Boolean Encodings from an Integer Perspective

Hendrik *Henk* Bierlee[1] Jip Dekker[2] Peter Stuckey[2]

CPAIOR'25, 12 Nov. 2025

[1]KU Leuven

[2]Monash University, OPTIMA

Introduction

Satisfiability (SAT) solvers are effective at solving SAT problems.

However, real-world problems containing **Pseudo-Boolean (PB) or Integer Linear (IL) constraints** require a SAT **encoding**.

Traditional PB encoding methods define a distinct **abstraction** (e.g. a binary decision diagram) and encoding procedure thereof.

In this work, we also propose an abstraction – but it covers and extends three fundamental PB encodings.

Background

A Constraint Satisfaction Problem (CSP) is $P \equiv \langle \mathcal{X}, D, \mathcal{C} \rangle$, where:

- $\mathcal{X}$ is a set of *decision variables*
- D maps every variable to a *domain* (i.e. finite set of discrete values)
- $\mathcal{C}$ is a set of *constraints* (i.e. relations between variables)

The Boolean Satisfiability (SAT) problem is a CSP, where:

- All domains map to *Boolean* values
- All constraints are *clauses* (i.e. disjunctions of literals, e.g. p or $\neg p$)

$$\langle \{p, q\}, \mathbb{B}, \{p \oplus q\} \rangle \equiv \langle \{p, q\}, \mathbb{B}, \{(p \vee q) \wedge (\neg p \vee \neg q)\} \rangle$$

Background

- A (normalized) PB constraint is of the form $\sum_{i=1}^{n} q_i b_i \leq k$ with Boolean literals b_i, e.g. $2b_1 + 3b_2 + 5b_3 \leq 6$
- An IL constraint is $\sum_{i=1}^{n} q_i x_i \leq k$ over integer variables x_i
- An **order** encoding of integer variable x creates Boolean variables $[\![x \geq d_2]\!], \ldots, [\![x \geq d_{D(x)}]\!]$ and encodes an implication chain:

$$[\![x \geq d_i]\!] \rightarrow [\![x \geq d_{i-1}]\!], d_i \in D(x)/\{d_1\}$$

- $[\![x \geq d_i]\!]$ is true iff x is assigned at least d_i
- Properties: $[\![x \geq \mathrm{lb}(x)]\!]$ is true, $[\![x \geq v]\!] \equiv [\![x \geq d_i]\!]$ if $d_{i-1} < v < d_i$

Decomposition

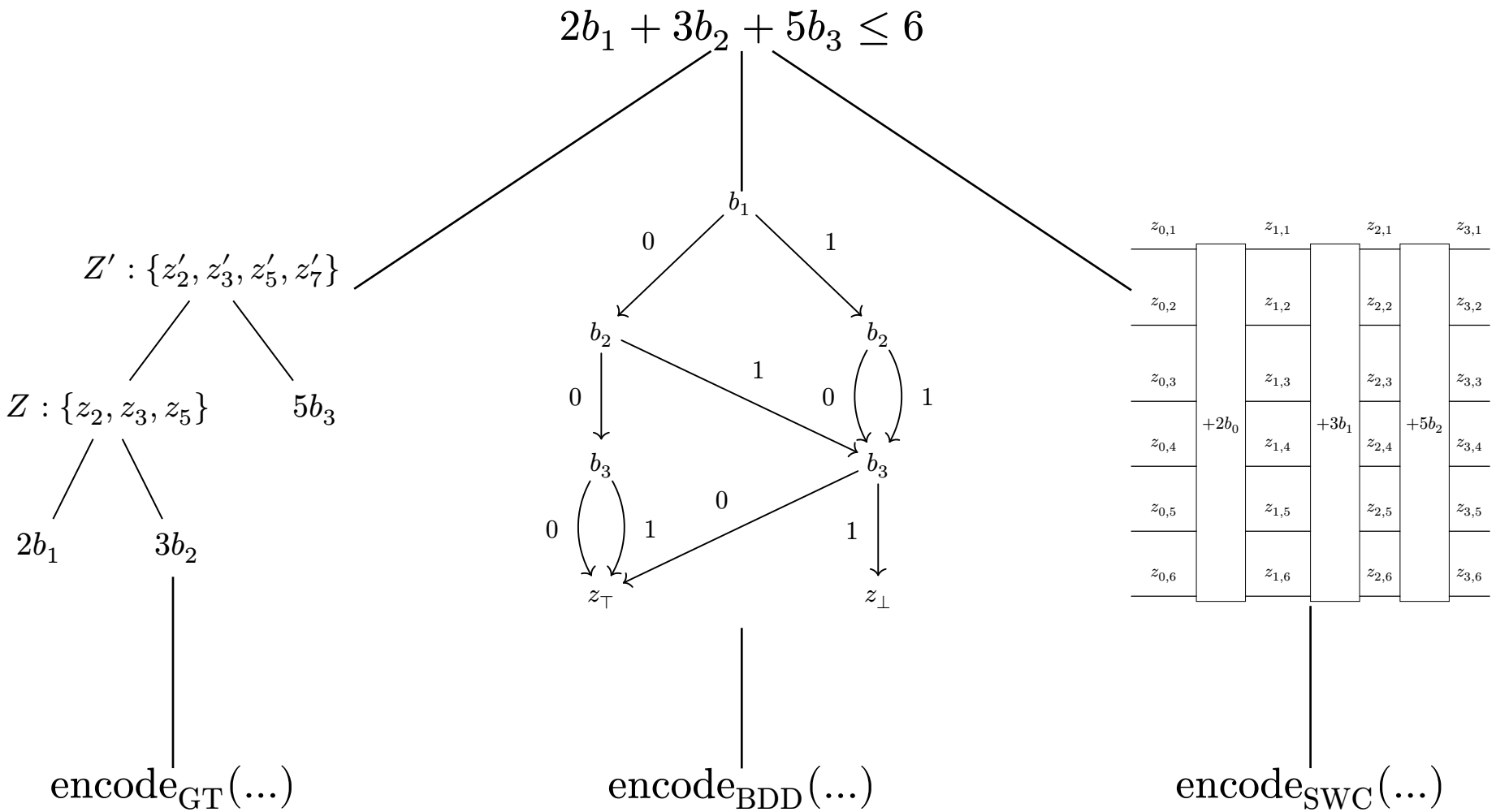
- A PB constraint can be converted to an IL constraint:

$$\begin{aligned} 2b_1 + 3b_2 \leq 4 &\equiv 2x'_1 \in \{0,1\} + 3x'_2 \in \{0,1\} \leq 4 \\ &\equiv x_1 \in \{0,2\} + x_2 \in \{0,3\} \leq 4 \end{aligned}$$

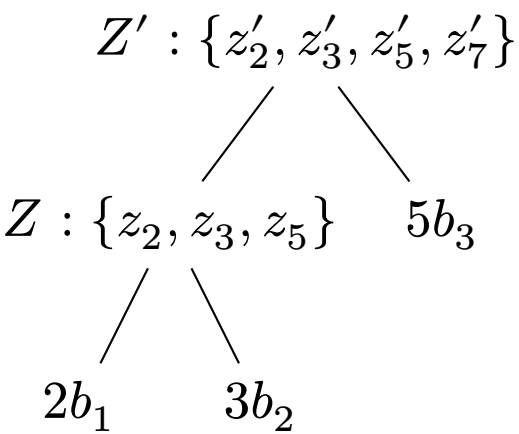
- The *ternary inequality* IL constraint $x + y \leq z$ can be order encoded:

$$\text{encode}_{\leq}(x + y \leq z) \equiv \bigwedge_{v \in D(x), w \in D(y)} (\llbracket x \geq v \rrbracket \wedge \llbracket y \geq w \rrbracket) \rightarrow \llbracket z \geq v + w \rrbracket$$

- We **decompose-and-encode** PB constraints into ternary inequalities


$2b_1 + 3b_2 + 5b_3 \leq 6$
$Z' : \{z'_2, z'_3, z'_5, z'_7\}$
$Z : \{z_2, z_3, z_5\}$
$5b_3$
$2b_1$
$3b_2$
b_1
b_2
b_2
b_3
b_3
$z_\top$
$z_\bot$
$z_{0,1}$ $z_{1,1}$ $z_{2,1}$ $z_{3,1}$
$z_{0,2}$ $z_{1,2}$ $z_{2,2}$ $z_{3,2}$
$z_{0,3}$ $z_{1,3}$ $z_{2,3}$ $z_{3,3}$
$z_{0,4}$ $z_{1,4}$ $z_{2,4}$ $z_{3,4}$
$z_{0,5}$ $z_{1,5}$ $z_{2,5}$ $z_{3,5}$
$z_{0,6}$ $z_{1,6}$ $z_{2,6}$ $z_{3,6}$
$+2b_0$
$+3b_1$
$+5b_2$
$\text{encode}_{\text{GT}}(\ldots)$
$\text{encode}_{\text{BDD}}(\ldots)$
$\text{encode}_{\text{SWC}}(\ldots)$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$

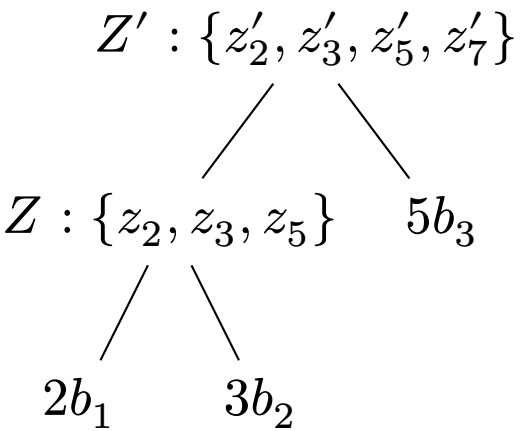


Abstraction A binary tree [1].

Each node Z contains auxiliary variables $z_{\min(w,k+1)}$ for every possible sum w of indices/coefficients of the variables in child nodes L and R.

Each z_v is true if the indices/coefficients of the variables in the sub-tree rooted at Z sums up to at least v.

$$2b_1 + 3b_2 + 5b_3 \leq 6$$



$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge \ldots$$

$\mathbf{encode_{GT}(\ldots)}$ For each internal node Z with child nodes L and R:

$$\bigwedge_{l_v \in L} l_v \rightarrow z_v$$

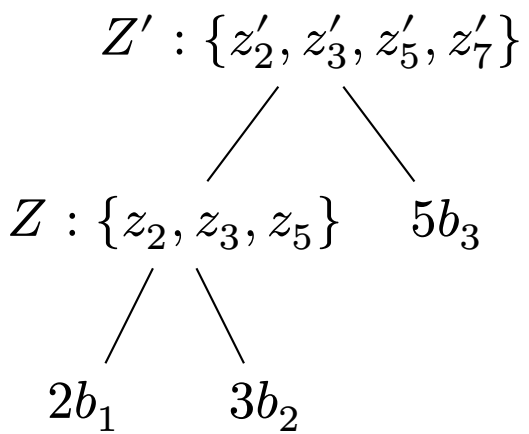
$$\wedge \bigwedge_{r_w \in R} r_w \rightarrow z_w$$

$$\wedge \bigwedge_{l_v \in L, r_w \in R} (l_v \wedge r_w) \rightarrow z_{\min(v+w, k+1)}$$

..and finally add $\neg z'_{k+1}$ in the root.

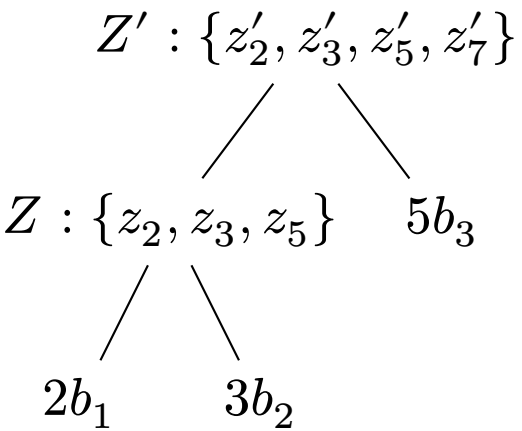
$$2b_1 + 3b_2 + 5b_3 \leq 6$$

$$x_1 \in \{0, 2\} + x_2 \in \{0, 3\} + x_5 \in \{0, 5\} \leq 6$$



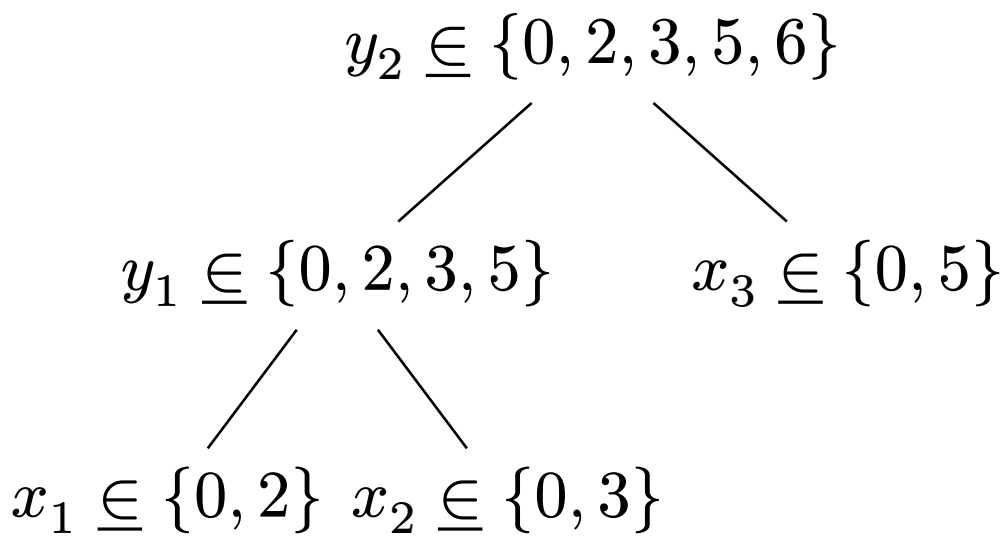
$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge ...$$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$

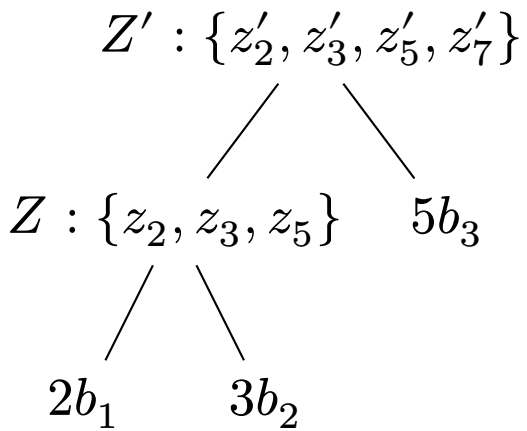


$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge ...$$

$$x_1 \sqsubseteq \{0, 2\} + x_2 \sqsubseteq \{0, 3\} + x_5 \sqsubseteq \{0, 5\} \leq 6$$

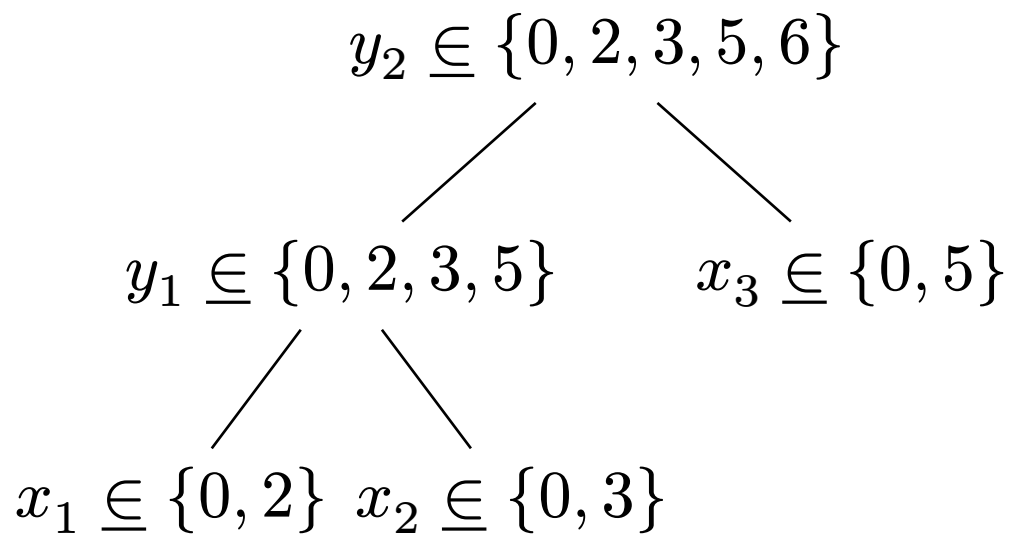


$$2b_1 + 3b_2 + 5b_3 \leq 6$$



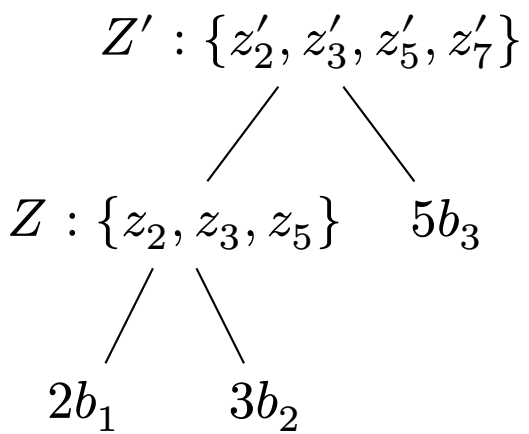
$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge \ldots$$

$$x_1 \subseteq \{0,2\} + x_2 \subseteq \{0,3\} + x_5 \subseteq \{0,5\} \leq 6$$



$$x_1 \subseteq \{0,2\} + x_2 \subseteq \{0,3\} \leq y_1 \subseteq \{0,2,3,5\}$$
$$\wedge\, y_1 + x_3 \subseteq \{0,5\} \leq y_2 \subseteq \{0,2,3,5,7\} \wedge y_2 \leq 6$$

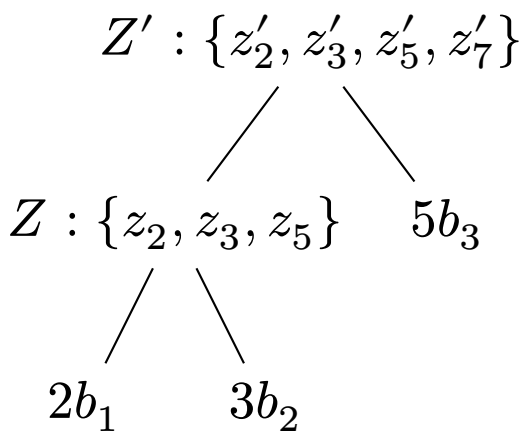
$$2b_1 + 3b_2 + 5b_3 \leq 6$$



$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge ...$$

$$x_1 \in \{0,2\} + x_2 \in \{0,3\} + x_5 \in \{0,5\} \leq 6$$
$$\equiv x_1 \in \{0,2\} + x_2 \in \{0,3\} \leq y_1 \in \{0,2,3,5\} \wedge$$
$$y_1 + x_3 \in \{0,5\} \leq y_2 \in \{0,2,3,5,7\} \wedge y_2 \leq 6$$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$

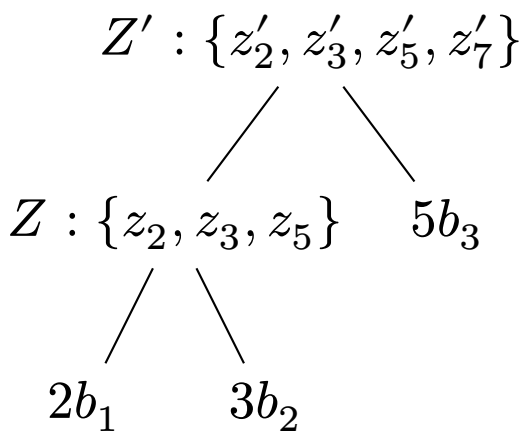


$$\Rightarrow (b_1 \rightarrow z_2) \wedge$$
$$(b_2 \rightarrow z_3) \wedge$$
$$((b_1 \wedge b_2) \rightarrow z_5) \wedge \ldots$$

$$x_1 \in \{0,2\} + x_2 \in \{0,3\} + x_5 \in \{0,5\} \leq 6$$
$$\equiv x_1 \in \{0,2\} + x_2 \in \{0,3\} \leq y_1 \in \{0,2,3,5\} \wedge$$
$$y_1 + x_3 \in \{0,5\} \leq y_2 \in \{0,2,3,5,7\} \wedge y_2 \leq 6$$

$$\Rightarrow ((\llbracket x_1 \geq 0 \rrbracket \wedge \llbracket x_2 \geq 0 \rrbracket) \rightarrow \llbracket y_1 \geq 0 \rrbracket) \wedge$$
$$((\llbracket x_1 \geq 0 \rrbracket \wedge \llbracket x_2 \geq 3 \rrbracket) \rightarrow \llbracket y_1 \geq 3 \rrbracket) \wedge$$
$$((\llbracket x_1 \geq 2 \rrbracket \wedge \llbracket x_2 \geq 0 \rrbracket) \rightarrow \llbracket y_1 \geq 2 \rrbracket) \wedge$$
$$((\llbracket x_1 \geq 2 \rrbracket \wedge \llbracket x_2 \geq 3 \rrbracket) \rightarrow \llbracket y_1 \geq 5 \rrbracket) \wedge \ldots$$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$



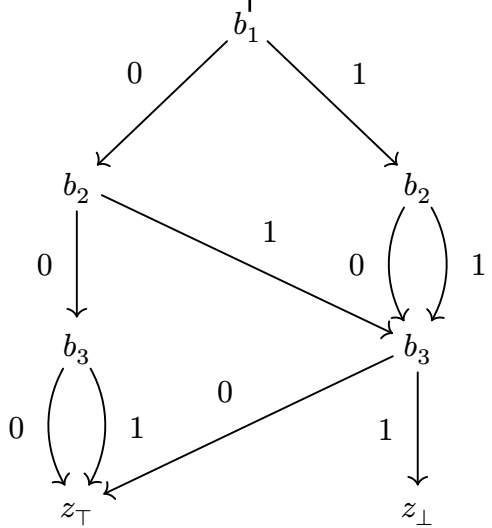
$$\begin{aligned}
& x_1 \Subset \{0,2\} + x_2 \Subset \{0,3\} + x_5 \Subset \{0,5\} \leq 6 \\
\equiv\ & x_1 \Subset \{0,2\} + x_2 \Subset \{0,3\} \leq y_1 \Subset \{0,2,3,5\} \wedge \\
& y_1 + x_3 \Subset \{0,5\} \leq y_2 \Subset \{0,2,3,5,7\} \wedge y_2 \leq 6
\end{aligned}$$

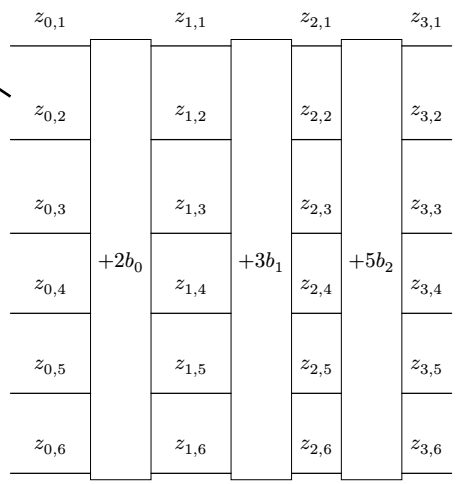
$$\begin{aligned}
\Rightarrow\ & (b_1 \rightarrow z_2) \wedge \\
& (b_2 \rightarrow z_3) \wedge \\
& ((b_1 \wedge b_2) \rightarrow z_5) \wedge ...
\end{aligned}$$

$$\begin{aligned}
\Rightarrow\ & ([\![x_1 \geq 2]\!] \rightarrow [\![y_1 \geq 2]\!]) \wedge \\
& ([\![x_2 \geq 3]\!] \rightarrow [\![y_1 \geq 3]\!]) \wedge \\
& (([\![x_1 \geq 2]\!] \wedge [\![x_2 \geq 3]\!]) \rightarrow [\![y_1 \geq 5]\!]) \wedge ...
\end{aligned}$$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$

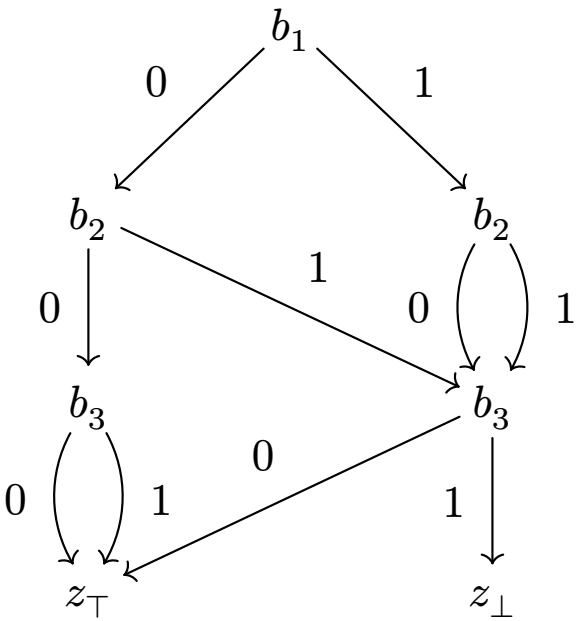
$$x_1 + x_2 \leq y_1 \in \{0, 2, 3, 5\}$$
$$\wedge\ y_1 + x_3 \leq 6$$





$\text{encode}_{\leq}(\ldots)$ $\text{encode}_{\text{BDD}}(\ldots)$ $\text{encode}_{\text{SWC}}(\ldots)$

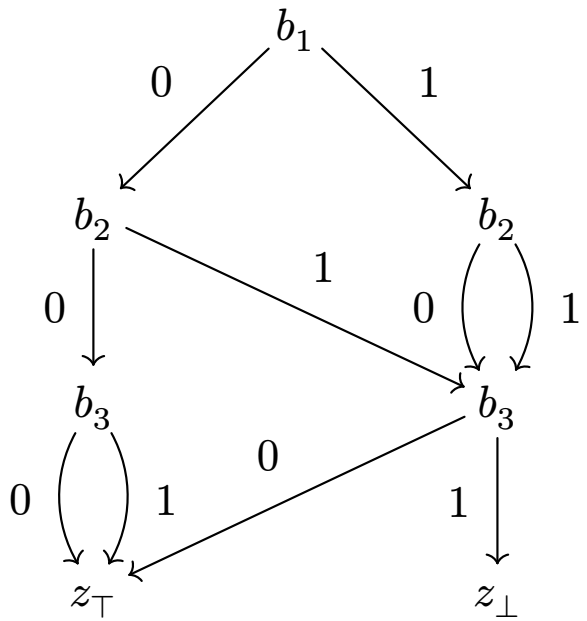
$$2b_1 + 3b_2 + 5b_3 \leq 6$$



Abstraction A binary DAG constructed to represent a PB constraint C [2]

Each decision b_i determines whether a path follows the 0- or 1-edge at node i. Paths leading to terminal $z_\top$ satisfy C.

$$2b_1 + 3b_2 + 5b_3 \leq 6$$



$$(z_1 \to z_1') \wedge$$
$$((z_1 \wedge b_1) \to z_1'') \wedge \ldots$$

Abstraction A binary DAG constructed to represent a PB constraint C [2]

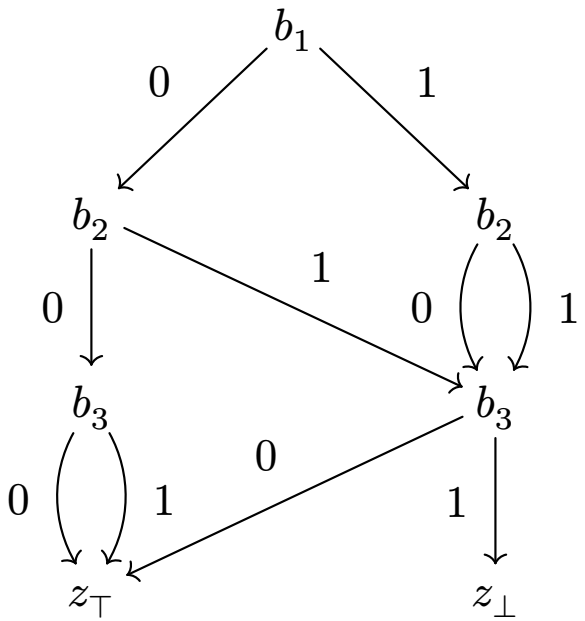
Each decision b_i determines whether a path follows the 0- or 1-edge at node i. Paths leading to terminal $z_\top$ satisfy C.

$\mathbf{encode_{BDD}(\ldots)}$ Associate node i with z_i, and for 0-child i' and 1-child i'':

$$(z_i \to z_{i'}) \wedge ((z_i \wedge b_i) \to z_{i''})$$

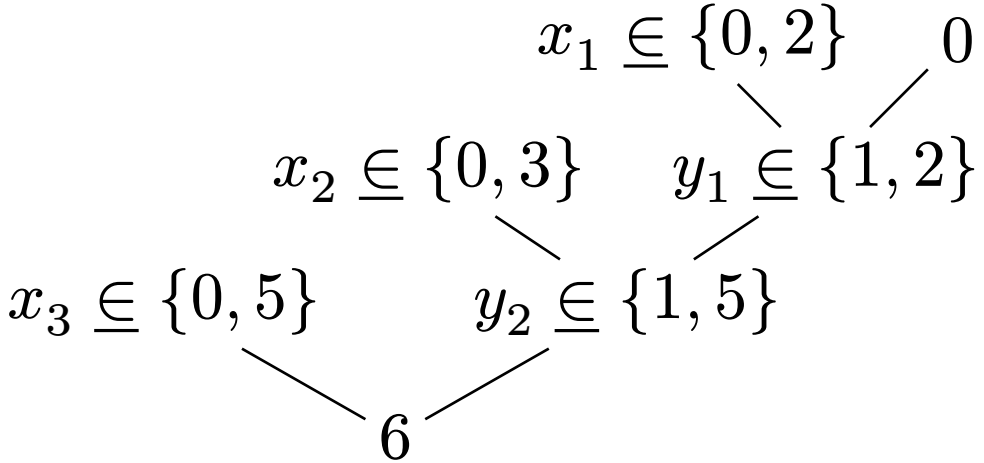
..and finally add $z_1 \wedge z_\top \wedge \neg z_\bot$

$$2b_1 + 3b_2 + 5b_3 \leq 6$$

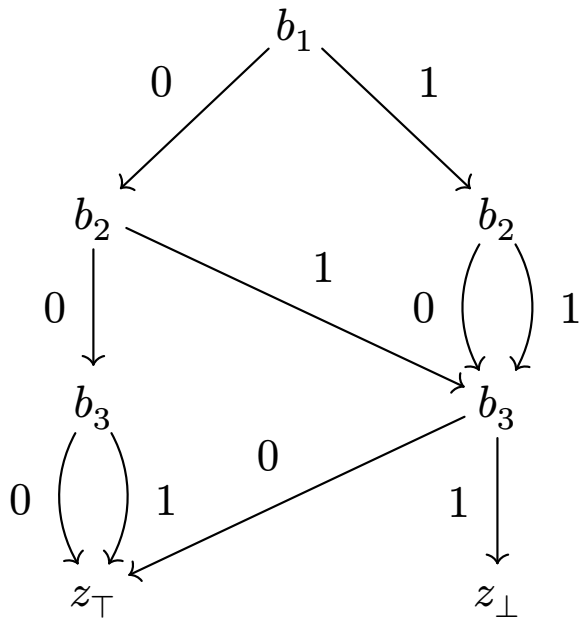


$$(z_1 \to z_1') \wedge$$
$$((z_1 \wedge b_1) \to z_1'') \wedge \ldots$$

$$x_1 \sqsubseteq \{0,2\} + x_2 \sqsubseteq \{0,3\} + x_5 \sqsubseteq \{0,5\} \leq 6$$

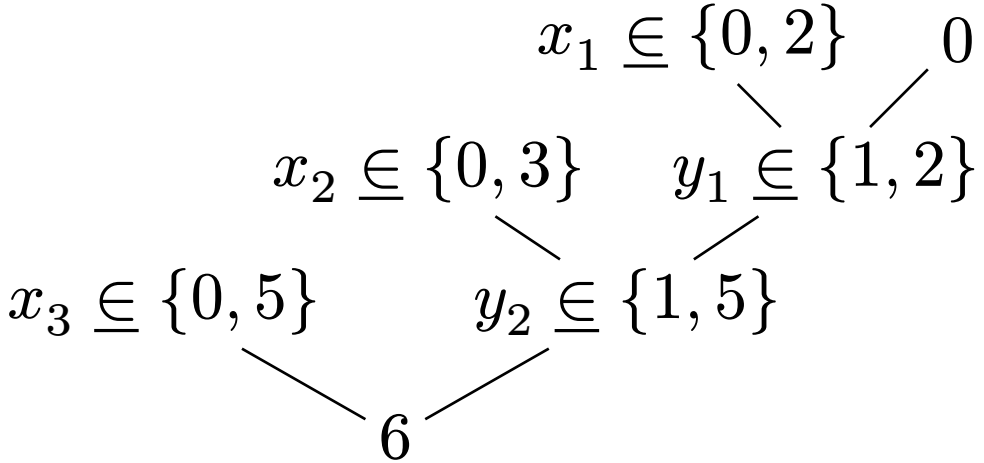


$$2b_1 + 3b_2 + 5b_3 \leq 6$$

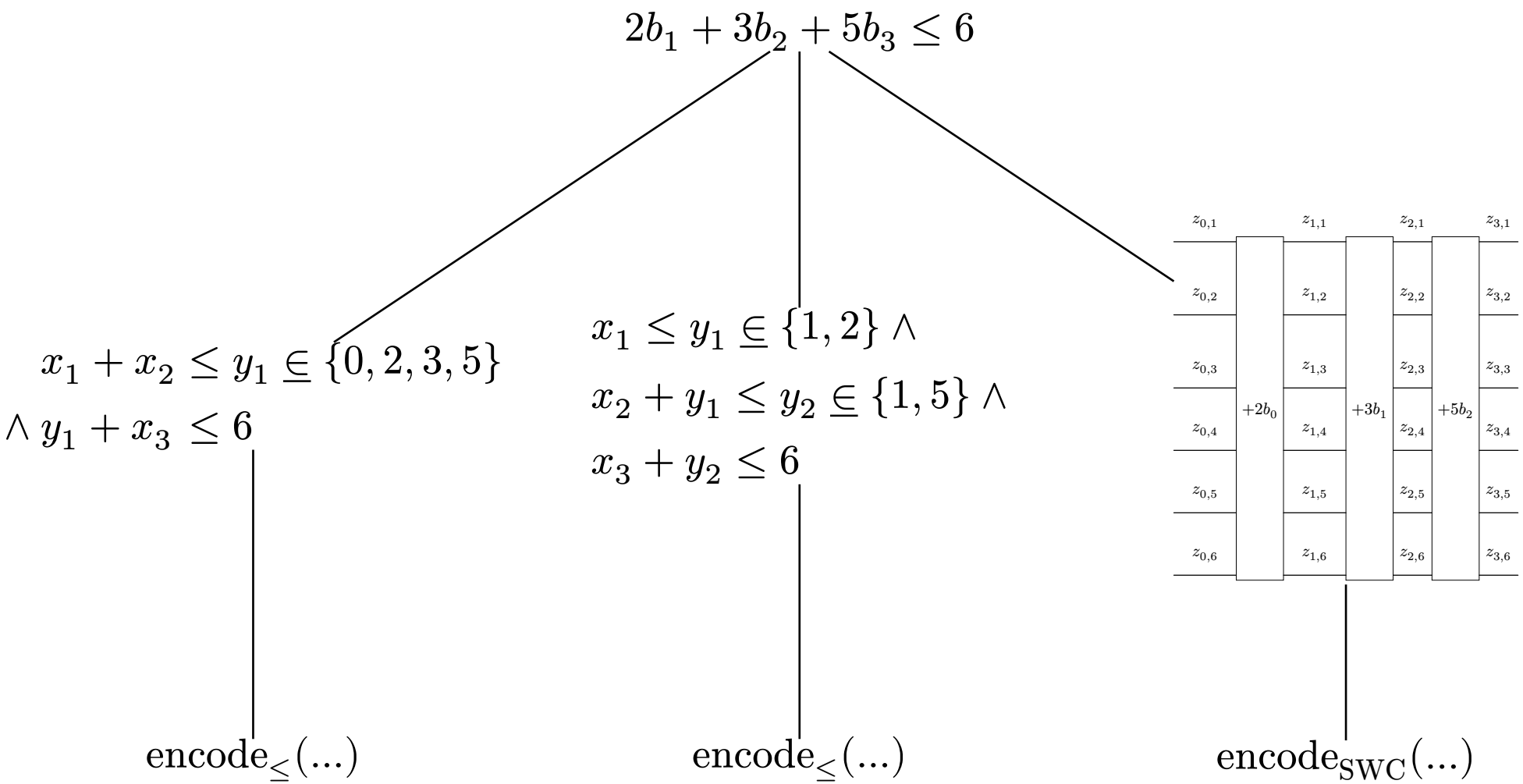


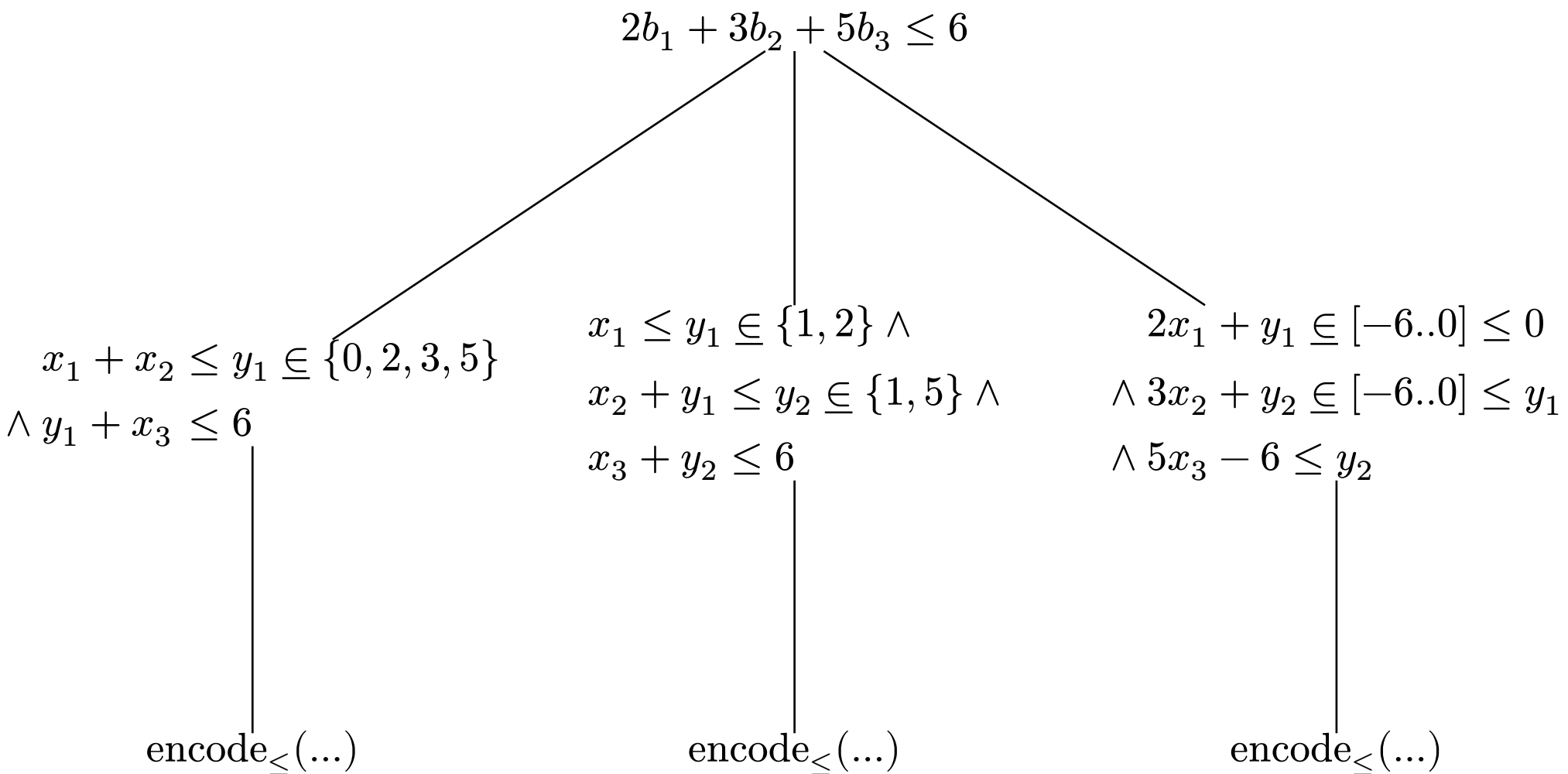
$$(z_1 \to z_1') \wedge$$
$$((z_1 \wedge b_1) \to z_1'') \wedge \ldots$$

$$x_1 \subseteq \{0,2\} + x_2 \subseteq \{0,3\} + x_5 \subseteq \{0,5\} \leq 6$$



$$x_1 \subseteq \{0,2\} \leq y_1 \subseteq \{1,2\} \wedge$$
$$x_2 \subseteq \{0,3\} + y_1 \leq y_2 \subseteq \{1,5\} \wedge$$
$$x_3 \subseteq \{0,5\} + y_2 \leq 6$$


$2b_1 + 3b_2 + 5b_3 \leq 6$
$x_1 + x_2 \leq y_1 \in \{0, 2, 3, 5\}$
$\wedge\ y_1 + x_3 \leq 6$
$x_1 \leq y_1 \in \{1, 2\} \wedge$
$x_2 + y_1 \leq y_2 \in \{1, 5\} \wedge$
$x_3 + y_2 \leq 6$
$z_{0,1}$ $z_{1,1}$ $z_{2,1}$ $z_{3,1}$
$z_{0,2}$ $z_{1,2}$ $z_{2,2}$ $z_{3,2}$
$z_{0,3}$ $z_{1,3}$ $z_{2,3}$ $z_{3,3}$
$z_{0,4}$ $z_{1,4}$ $z_{2,4}$ $z_{3,4}$
$z_{0,5}$ $z_{1,5}$ $z_{2,5}$ $z_{3,5}$
$z_{0,6}$ $z_{1,6}$ $z_{2,6}$ $z_{3,6}$
$+2b_0$
$+3b_1$
$+5b_2$
$\text{encode}_{\leq}(\ldots)$
$\text{encode}_{\leq}(\ldots)$
$\text{encode}_{\text{SWC}}(\ldots)$


$2b_1 + 3b_2 + 5b_3 \leq 6$
$x_1 + x_2 \leq y_1 \in \{0, 2, 3, 5\}$
$\wedge\ y_1 + x_3 \leq 6$
$x_1 \leq y_1 \in \{1, 2\} \wedge$
$x_2 + y_1 \leq y_2 \in \{1, 5\} \wedge$
$x_3 + y_2 \leq 6$
$2x_1 + y_1 \in [-6..0] \leq 0$
$\wedge\ 3x_2 + y_2 \in [-6..0] \leq y_1$
$\wedge\ 5x_3 - 6 \leq y_2$
$\text{encode}_{\leq}(...)$
$\text{encode}_{\leq}(...)$
$\text{encode}_{\leq}(...)$

Advantages and extensions

We now have integer representations of three encoding methods.

Consistency proofs

Each encoding has been proven to be GAC in their respective papers. We show this once for all encodings by proving GAC on the encoded decomposition.

Summary. While the order encoding of $x + y \leq z$ is *not* GAC without implication chains on all variables, implication chains are implicitly enforced by the tree structure present in all encodings.

Advantages and extensions

Support for IL constraints

For a PB constraint over $q_i b_i$, the leaves of the decomposition are integer variables with $\{0, q_i\}$ domains. For IL constraints, these leaves have integer domains, like the internal nodes.

Advantages and extensions

Support for IL constraints

For a PB constraint over $q_i b_i$, the leaves of the decomposition are integer variables with $\{0, q_i\}$ domains. For IL constraints, these leaves have integer domains, like the internal nodes.

If At-Most-One [3] or IC [4] **side-constraints** are present on a set of Boolean variables, then this set can also be seen as direct or order encoded integer variables.

Advantages and extensions

Support for equality constraints

An *equality* PB constraint is often encoded by two inequalities:

$$\sum_{i=1}^{n} q_i b_i = k \equiv \sum_{i=1}^{n} q_i b_i \leq k \wedge \sum_{i=1}^{n} q_i b_i \geq k$$

In the worst case, this *doubles* the number of auxiliary variables. Instead, we replace the ternary inequalities in the decomposition for ternary **equalities**:

$$x + y = z \equiv x + y \leq z \wedge x + y \geq z$$

Advantages and extensions

Support for mixed integer encodings

So far, all integer variables and constraints in the decomposition use the order encoding, denoted: $x{:}\mathbb{O} + y{:}\mathbb{O} \leq z{:}\mathbb{O}$. However, the decomposition is **encoding-agnostic**.

If all variables are binary encoded, i.e. $x{:}\mathbb{B} + y{:}\mathbb{B} = z{:}\mathbb{B}$, the ripple-carry adder encoding should be used instead. If there is a mix of encodings, $x{:}\mathbb{B} + y{:}\mathbb{O} \leq z{:}\mathbb{B}$, **coupling** encodings are required.

Integer variables with large domains benefit from the binary encoding, whereas small domains fit the order encoding.

Experimental evaluation

Solve times are compared on:

- Three generated instance sets of two problems:
  MBKP Multi-dimensional bounded knapsack (inequalities)
  MBSSP Multi-dimensional bounded subset-sum (equalities)
- Include three baselines:
  - `picat-sat` [5]
  - `savile-row` [6]
  - `diet-sugar` [7]
- Time-out of 3 minutes

(N, B, M, Q)	$(50, 1, 25, 50)$		$(10, 10, 200, 50)$	
	solved	vars./cl.	solved	vars./cl.
GT + $\mathbb{B}$	84 (6.0)	15/76	85 (11.5)	36/224
GT + $\mathbb{O} \leq 25$	88 (5.7)	14/54	**94 (10.9)**	39/238
GT + $\mathbb{O} \leq 75$	**89 (5.7)**	14/54	93 (11.8)	43/252
GT + $\mathbb{O}$	67 (3.5)	84/7852	0	701/159425
GT	68 (3.9)	84/7852	0	985/164102
BDD + $\mathbb{B}$	78 (3.3)	23/99	78 (10.1)	40/234
BDD + $\mathbb{O} \leq 25$	75 (8.9)	26/113	93 (13.4)	42/244
BDD + $\mathbb{O} \leq 75$	76 (10.2)	29/114	90 (13.8)	43/252
BDD + $\mathbb{O}$	75 (6.4)	351/694	61 (17.4)	775/7983
BDD	74 (5.2)	351/694	0	8313/16588
Savile Row (GMTO)	87 (5.1)	20/43	89 (8.0)	50/179

Tab. 1: Results for 2/3 instance sets of 100 MBKP + virtual best control

Conclusion and future work

Conclusion

- We formalised an **abstraction** that covers three PB encoding methods
- We develop and empirically evaluate four **extensions**

Future work

- Cover additional encoding methods: ripple-carry adders (formally), sorting networks, and other constraint types
- Additional extensions: proof-logging, other CP techniques

Conclusion and future work

[1] S. Joshi, R. Martins, and V. M. Manquinho, "Generalized Totalizer Encoding for Pseudo-Boolean Constraints," in *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, G. Pesant, Ed., in Lecture Notes in Computer Science, vol. 9255. Springer, 2015, pp. 200–209. doi: 10.1007/978-3-319-23219-5_15.

[2] I. Abío, R. Nieuwenhuis, A. Oliveras, E. Rodríguez-Carbonell, and V. Mayer-Eichberger, "A New Look at BDDs for Pseudo-Boolean Constraints," *J. Artif. Intell. Res.*, vol. 45, pp. 443–480, 2012, doi: 10.1613/jair.3653.

[3] M. Bofill, J. Coll, P. Nightingale, J. Suy, F. Ulrich-Oltean, and M. Villaret, "SAT encodings for Pseudo-Boolean constraints together with at-most-one constraints," *Artif. Intell.*, vol. 302, p. 103604, 2022, doi: 10.1016/j.artint.2021.103604.

[4] I. Abío, V. Mayer-Eichberger, and P. J. Stuckey, “Encoding Linear Constraints with Implication Chains to CNF,” in *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, G. Pesant, Ed., in Lecture Notes in Computer Science, vol. 9255. Springer, 2015, pp. 3–11. doi: 10.1007/978-3-319-23219-5_1.

[5] N.-F. Zhou and H. Kjellerstrand, “The Picat-SAT Compiler,” in *Practical Aspects of Declarative Languages - 18th International Symposium, PADL 2016, St. Petersburg, FL, USA, January 18-19, 2016. Proceedings*, M. Gavanelli and J. H. Reppy, Eds., in Lecture Notes in Computer Science, vol. 9585. Springer, 2016, pp. 48–62. doi: 10.1007/978-3-319-28228-2_4.

[6] P. Nightingale, P. Spracklen, and I. Miguel, “Automatically Improving SAT Encoding of Constraint Problems Through Common Subexpression Elimination in Savile Row,” in *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, G. Pesant, Ed., in Lecture Notes in

Computer Science, vol. 9255. Springer, 2015, pp. 330–340. doi: 10.1007/978-3-319-23219-5_23.

[7] T. Soh, D. Le Berre, M. Banbara, and N. Tamura, “sCOP: SAT-based Constraint Programming System,” *Proceedings of XCSP3 Competition 2018 (XCSP18)*, pp. 93–94, 2018.