Automatic Generation of Polynomial Symmetry Breaking Constraints

Mădălina Erașcu* and Johannes Middeke**

*West University of Timișoara, Romania
**Temple University, Japan Campus, Tokyo, Japan

madalina.erascu@e-uvt.ro
johannes.middeke@tuj.temple.edu

July 15, 2026

Outline

Contents

Motivation

Symmetries appear in *domains* like:

- symbolic computation [1]
- optimization and constraints programming [2, 3]
- mixed-integer linear programming [4, 5, 6]
- satisfiability [7, 8]
- satisfiability modulo theory [9]

Why symmetry is problematic?

- Symmetries are seen as redundant branches of the search tree that must be pruned since they lead to many equivalent solutions.

What is symmetry breaking?

- Symmetry breaking means adding extra constraints which eliminate as many symmetric solutions as possible.
- In *static symmetry breaking* constraints are added a priori, before solving the problem.

Motivation

Symmetries appear in *domains* like:

- symbolic computation [1]
- optimization and constraints programming [2, 3]
- mixed-integer linear programming [4, 5, 6]
- satisfiability [7, 8]
- satisfiability modulo theory [9]

Why symmetry is problematic?

- Symmetries are seen as redundant branches of the search tree that must be pruned since they lead to many equivalent solutions.

What is symmetry breaking?

- Symmetry breaking means adding extra constraints which eliminate as many symmetric solutions as possible.
- In *static symmetry breaking* constraints are added a priori, before solving the problem.

Motivation

Symmetries appear in *domains* like:

- symbolic computation [1]
- optimization and constraints programming [2, 3]
- mixed-integer linear programming [4, 5, 6]
- satisfiability [7, 8]
- satisfiability modulo theory [9]

Why symmetry is problematic?

- Symmetries are seen as redundant branches of the search tree that must be pruned since they lead to many equivalent solutions.

What is symmetry breaking?

- Symmetry breaking means adding extra constraints which eliminate as many symmetric solutions as possible.
- In *static symmetry breaking* constraints are added a priori, before solving the problem.

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Related Work

1. dynamic symmetry breaking: adding constraints during search so that partial assignments symmetric to those already considered will not be visited.

2. static symmetry breaking: adding constraints *a priori* to restrict feasibility to orbit representatives.
   - in *constraints programming (CP)*, the methods are based on lex-leader and related ordering constraints [3, 2].
   - in *mixed-integer programming (MIP)*, the most well-known approaches are polyhedral:
     - for assignment-matrix formulations with interchangeable rows/columns, orbitopes encode the convex hull of lex-sorted 0-1 matrices and yield strong linear descriptions and compact extended formulations [5, 10]
     - a more general approach, static inequalities can also be derived from computational group theory (stabilizer chains, Schreier-Sims tables) to break symmetry in arbitrary formulations [11, 6].
   - in *satisfiability modulo theory (SMT)* there are fewer approaches; symmetry breaking is typically imposed on the Boolean abstraction [9].

Contents

Problem Specification

We consider *integer linear problems*

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & Ax \geq b, \\ & x \in U^n \end{aligned}$$

where

- $f : \mathbb{R} \to \mathbb{R}$ is a function,
- $x = (x_1, \ldots, x_n)$ are the decision variables,
- $A \in \mathbb{R}^{m \times n}$ is a real matrix,
- $b \in \mathbb{R}^m$ is a real vector,
- $U \subset \mathbb{R}$ is a *finite set*.

We call f the *objective function* and $\{Ax \geq b : x \in U^n\}$ the *feasible set*.

Symmetries & Symmetry Breakers

An integer linear problem has the form

$$\text{minimize } f(x) \text{ subject to } Ax \geq b, x \in U^n.$$

A *symmetry* is a permutation matrix $P \in \Pi_n$ such that

$$\forall x \in U^n: \quad f(Px) = f(x) \quad \text{and} \quad APx \geq b \iff Ax \geq b.$$

A *symmetry breaker* is a predicate $\psi: U^n \to \{true, false\}$ such that

$$\{Ax \geq b : x \in U^n\} \cap \psi^{-1}(true)$$

contains an optimal solution.

It is possible that $\psi_1, \ldots, \psi_\ell$ are symmetry breakers but $\psi_1 \wedge \ldots \wedge \psi_\ell$ is not.

Main Theorem

Assume that G is a (subgroup of the) *group of symmetries* for

$$\text{minimize } f(x) \text{ subject to } Ax \geq b, x \in U^n.$$

Let $h \in \mathbb{R}[x]$ be a polynomial.

Then

$$\{h(Px) - h(x) \leq 0 : P \in G\}$$

is a family of *compatible symmetry breakers*.

Example

Consider the (toy) problem

$$\begin{aligned} \text{minimize} \quad & x + y \\ \text{subject to} \quad & x + y \geq 1 \text{ and } x, y \in \{0, 1\}. \end{aligned}$$

Optimal solutions are $x = 1, y = 0$ and $x = 0, y = 1$.

The symmetry group is $\Pi_2 = \{\left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)\}$.

Let $h(x, y) = x^2$. Then

$$h(y, x) - h(x, y) = y^2 - x^2 = (y - x)(y + x) \leq 0$$

is a symmetry breaker.

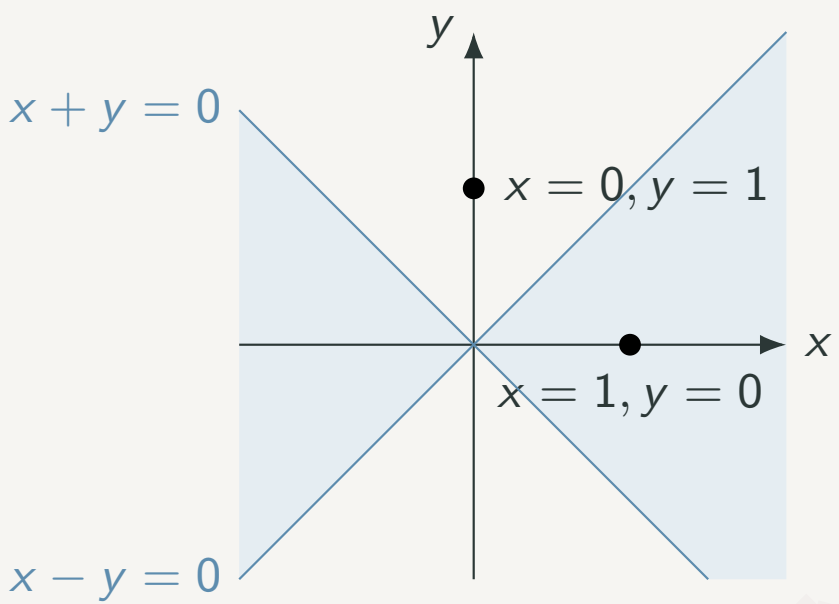
Example

For the problem

$$\begin{aligned} \text{minimize} \quad & x + y \\ \text{subject to} \quad & x + y \geq 1 \text{ and } x, y \in \{0, 1\}. \end{aligned}$$

with solutions $x = 0, y = 1$ and $x = 1, y = 0$ we have found the symmetry breaker

$$(y - x)(y + x) \leq 0.$$



Sketch of the solutions.

The shaded area is

$$(y - x)(y + x) \leq 0.$$

Contents

Case Study: Bin Packing

Bin packing problem [12, Chapter 8]: place m *items* into the smallest number of up to n *bins*.

$$\begin{aligned} \text{minimise} \quad & y_1 + \ldots + y_n \\ \text{subject to} \quad & \forall k : s_1 x_{1k} + \ldots + s_m x_{mk} \leq B y_k \\ & \forall i : x_{i1} + \ldots + x_{in} = 1 \\ & \forall i, k : x_{ik} \in \{0, 1\} \\ & \forall k : y_k \in \{0, 1\} \end{aligned}$$

where

- s_i is the *size* of item i,
- $x_{ik} = 1$ means item i is placed in bin k,
- $y_k = 1$ means bin k is used,
- B is the maximum *bin capacity*.

Symmetries

Assume the item sizes are ordered

$$s_1 = \ldots = s_{i_1} < s_{i_1+1} = \ldots = s_{i_2} < s_{i_2+1} = \ldots < s_{i_\ell+1} = \ldots = s_m.$$

Arrange the variables in a block matrix

$$V = \begin{array}{c} \\ s_1 \\ \vdots \\ s_{i_1} \\ \vdots \\ s_{i_\ell+1} \\ \vdots \\ s_m \end{array} \begin{pmatrix} y_1 & y_2 & \cdots & y_n \\ \hline x_{11} & x_{12} & \cdots & x_{1n} \\ \vdots & \vdots & & \vdots \\ x_{11} & x_{12} & \cdots & x_{1n} \\ \hline \vdots & \vdots & & \vdots \\ \hline x_{i_\ell+1,1} & x_{i_\ell+1,2} & \cdots & x_{i_\ell+1,n} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

Then every row and column permutation which respects the block structure is a symmetry.

Symmetries

Using *vectorization* and *Kronecker products*, we can write the system as

$$\begin{aligned} \text{minimize} \quad & (e \otimes e_1)\,\mathrm{vec}(V) \\ \text{subject to} \quad & \begin{pmatrix} \mathbf{1}_n \otimes u \\ e \otimes (0\ \mathbf{1}_m) \\ -e \otimes (0\ \mathbf{1}_m) \end{pmatrix} \mathrm{vec}(V) \geq \begin{pmatrix} 0 \\ e \\ e \end{pmatrix}, \quad \mathrm{vec}(V) \in \{0,1\}^{(m+1)n} \end{aligned}$$

where

- V is the matrix from the previous slide,
- $u = (B, -s_1, \ldots, -s_m)$,
- $\mathbf{1}_\ell$ is the $\ell \times \ell$ identity matrix,
- $e = (1, \ldots, 1)$,
- and $e_1, e_2, \ldots$ are the canonical basis vectors.

Symmetries are of the form

$$C \otimes \mathrm{diag}(1, R) \qquad \text{where} \quad C \in \Pi_n \text{ and } R \in \Pi_m.$$

Symmetry Breaker Generation

Symmetry breakers are generated by the following method:

1. Pick a random polynomial $h \in \mathbb{R}[\mathrm{vec}(V)]$ according to a *template*.
2. Let C be the product of 50 random transposition matrices in Π_n.
3. Let R be the product of 50 random transposition matrices in Π_m.
4. Set $P = C \otimes \mathrm{diag}(1, R)$.
5. Return $h(P\,\mathrm{vec}(V)) - h(\mathrm{vec}(V)) \leq 0$.

The templates are determined by their *shape*

$$x, \quad y, \quad x+y, \quad x^2, \quad y^2, \quad xy, \quad x^2+y^2, \quad x+y^2, \quad \text{or} \quad x^2+y$$

and their *size* (number of variables).

We generated families of breakers from the same polynomial h by generating multiple *permutations*.

Contents

Experimental Results

- Benchmark Generation: Near Half-Capacity Families inspired by [13].
- We generated 1440 problem instances using the classes 3, 5, 7, 9, augmented with different symmetry breaking templates and sizes for the #vars and #perms; $B = 100$

Class	3	5	7	9
n	2000	2000	1024	1000
intervals	[49,51]	[48,52]	[47,53]	[46,54]

Category	**Templates**
Linear	x, y, $x + y$
Quadratic	x^2, y^2, $x\,y$, $x^2 + y^2$
Mixed	$x + y^2$, $x^2 + y$

Size	**#vars**	**#perms**
Few variables, few permutations	≈ 10	50
Few variables, many permutations	≈ 10	500
Many variables, few permutations	≈ 1000	50
Numerous variables, few permutations	≈ 4000	50

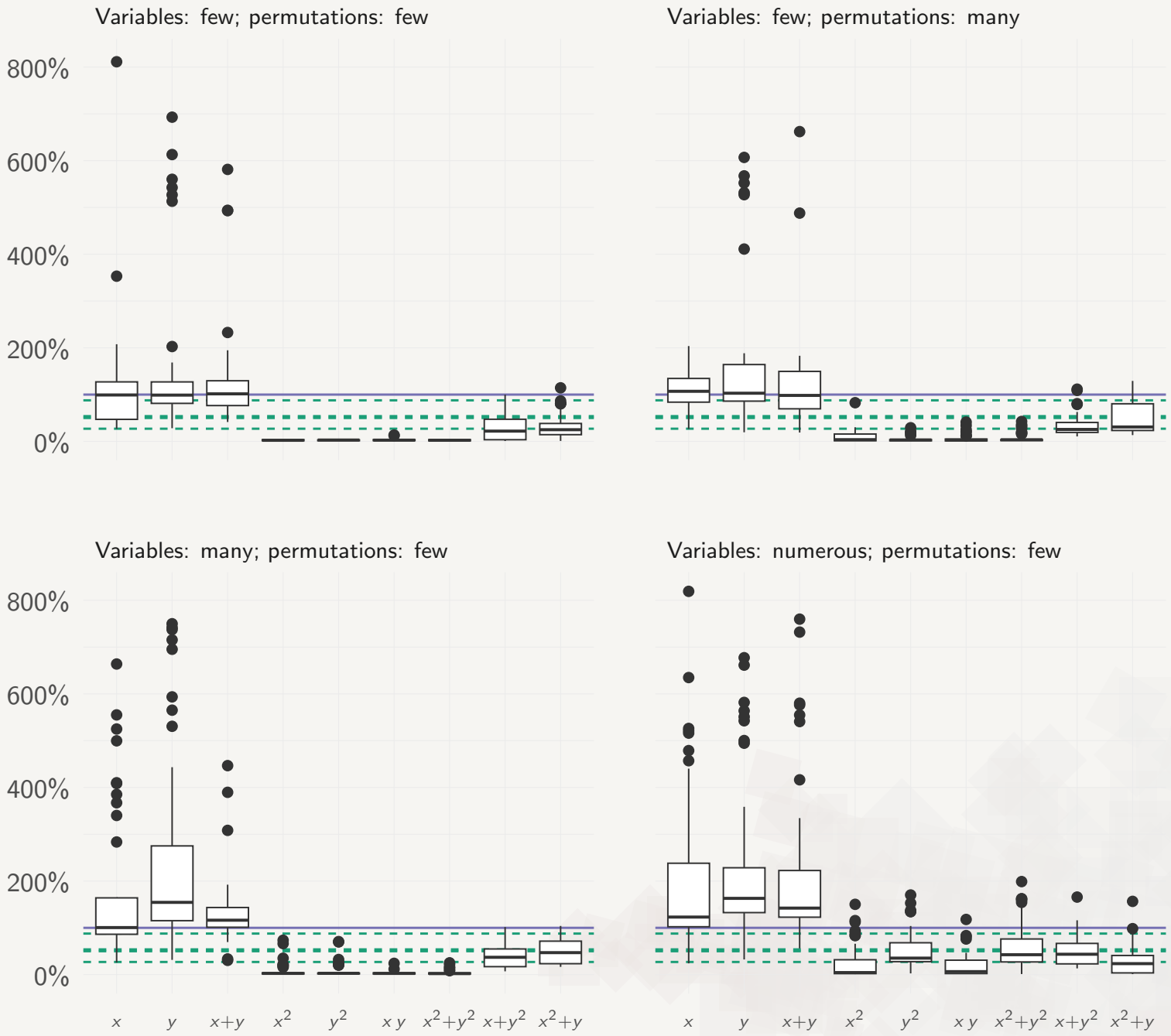
- The generated benchmarks have around 4 million variables and around 4000 constraints for the 3 and 5 classes cases.

Experimental Results (cont'd)

- The experiments were run using Gurobi Optimizer 12.0.3 [14] (academic license) on an Apple M2 Pro (10 cores).
- We used a deadline of 1800 work units for each problem instance.

Parameter	Baseline	Default	Conservative	Aggressive
`Nonconvex`	2	2	2	2
`Presolve`	0	-1	-1	-1
`Symmetry`	0	-1	1	2

Experimental Results (cont'd)



Contents

Conclusions and Future Work

Conclusions

- Quadratic breakers consistently outperform linear ones as well as Gurobi's built-in symmetry handling; in particular, the small-scale (i. e., few variables, few permutations) *xy* template.
- Small-scale symmetry breaker sets, both linear and quadratic, are the most beneficial across all templates.
- Purely linear breakers tend to be less efficient, especially when the number of variables or permutations grows.
- As the scale of the symmetry breakers increases, either by involving many variables or by sampling many permutations, the performance becomes more variable.
- Only approx. 0.7% problem instances reached the time limit mainly with linear symmetry breakers.

Future Work

1. *Explanation of why the non-linear symmetry breakers are better than the linear ones.*
2. *Explanation of how/why the choice of polynomials affects performance.*
3. Evaluate the approach also to cubic symmetry breakers.
4. Evaluate our method on arbitrary functions (e.g., algebraic or transcendental functions).
5. Investigate other mathematical programming solvers (e.g., CPLEX [15]), SMT (e.g., Z3 [16]) and constraint programming (e.g., OR-Tools [17]) approaches that can handle non-linear constraints.
6. Apply, and extend, our symmetry breaking generation approach to other case studies.

Conclusions and Future Work

Conclusions

- Quadratic breakers consistently outperform linear ones as well as Gurobi's built-in symmetry handling; in particular, the small-scale (i. e., few variables, few permutations) *xy* template.
- Small-scale symmetry breaker sets, both linear and quadratic, are the most beneficial across all templates.
- Purely linear breakers tend to be less efficient, especially when the number of variables or permutations grows.
- As the scale of the symmetry breakers increases, either by involving many variables or by sampling many permutations, the performance becomes more variable.
- Only approx. 0.7% problem instances reached the time limit mainly with linear symmetry breakers.

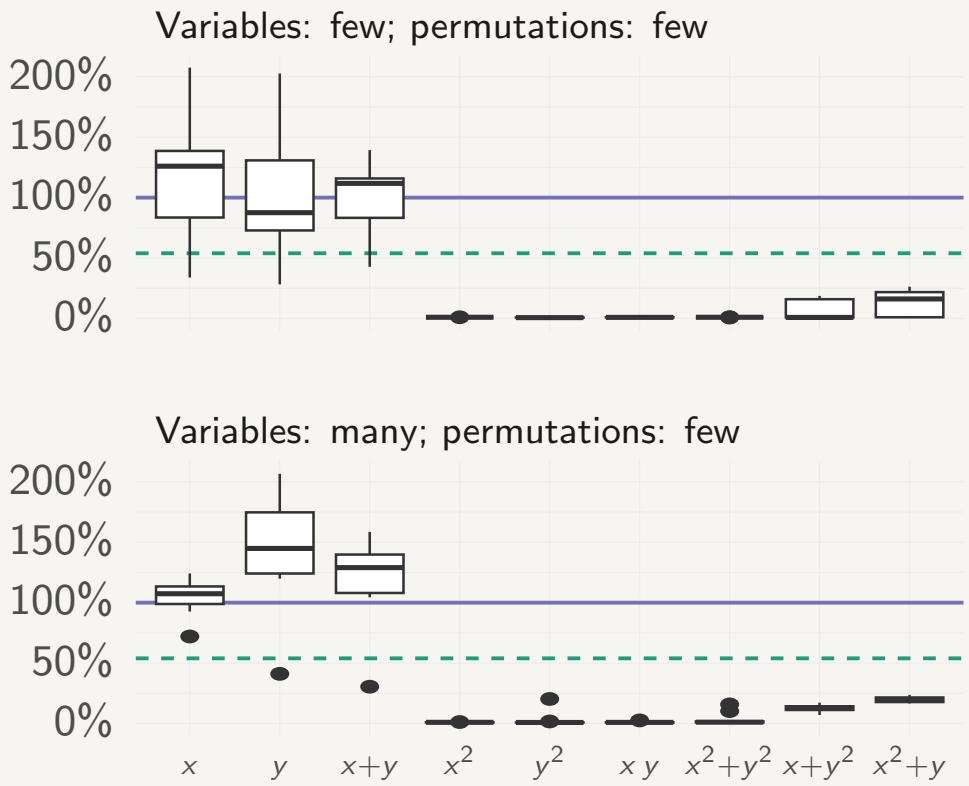
Future Work

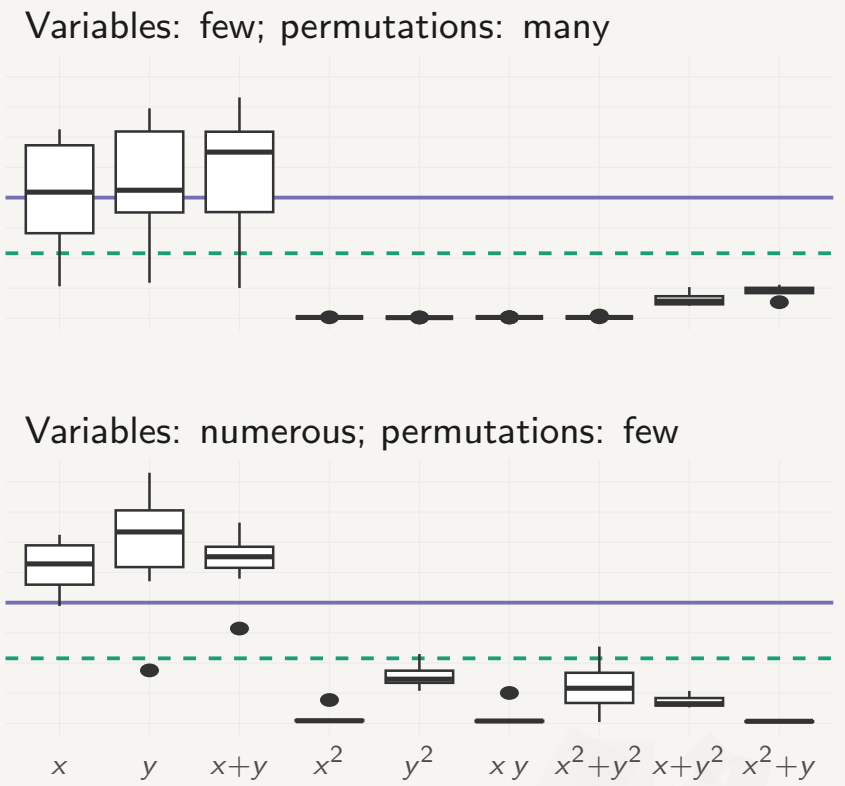
1. *Explanation of why the non-linear symmetry breakers are better than the linear ones.*
2. *Explanation of how/why the choice of polynomials affects performance.*
3. Evaluate the approach also to cubic symmetry breakers.
4. Evaluate our method on arbitrary functions (e.g., algebraic or transcendental functions).
5. Investigate other mathematical programming solvers (e.g., CPLEX [15]), SMT (e.g., Z3 [16]) and constraint programming (e.g., OR-Tools [17]) approaches that can handle non-linear constraints.
6. Apply, and extend, our symmetry breaking generation approach to other case studies.

Contents

Experimental Results

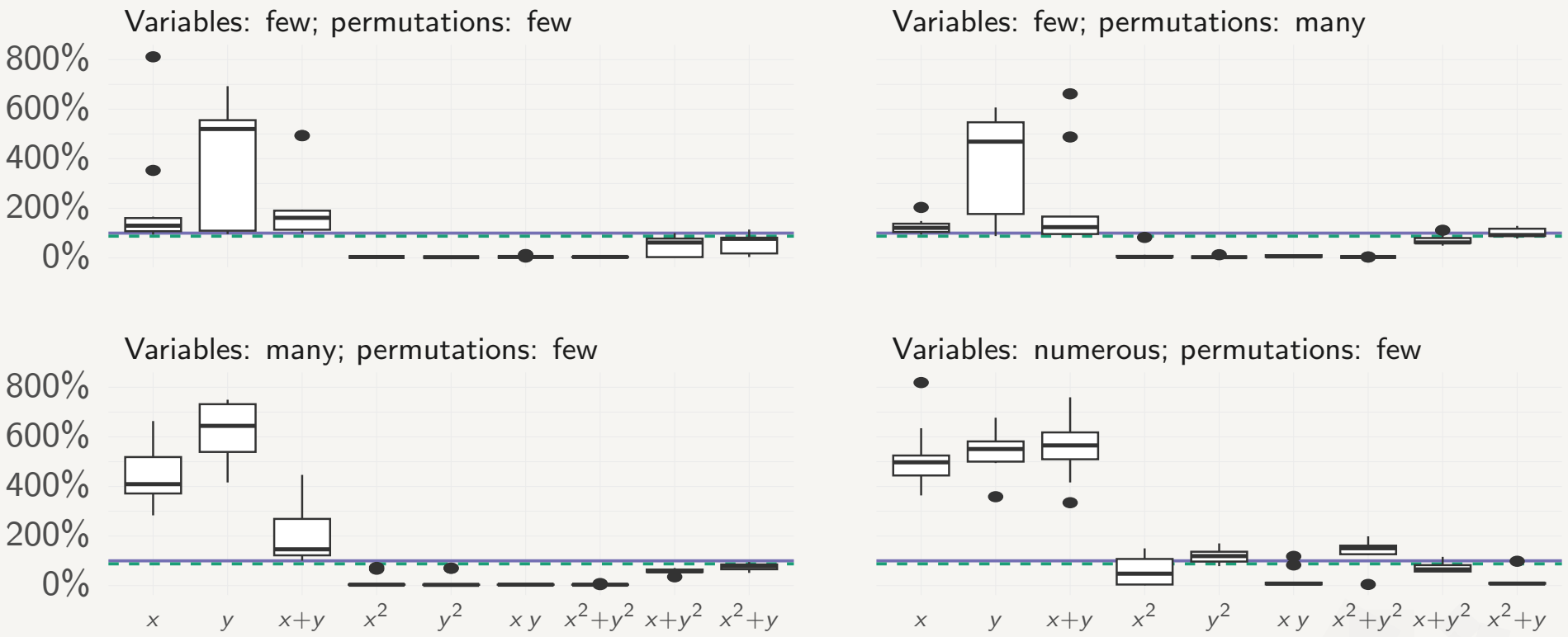
Results for the 3 classes case only.





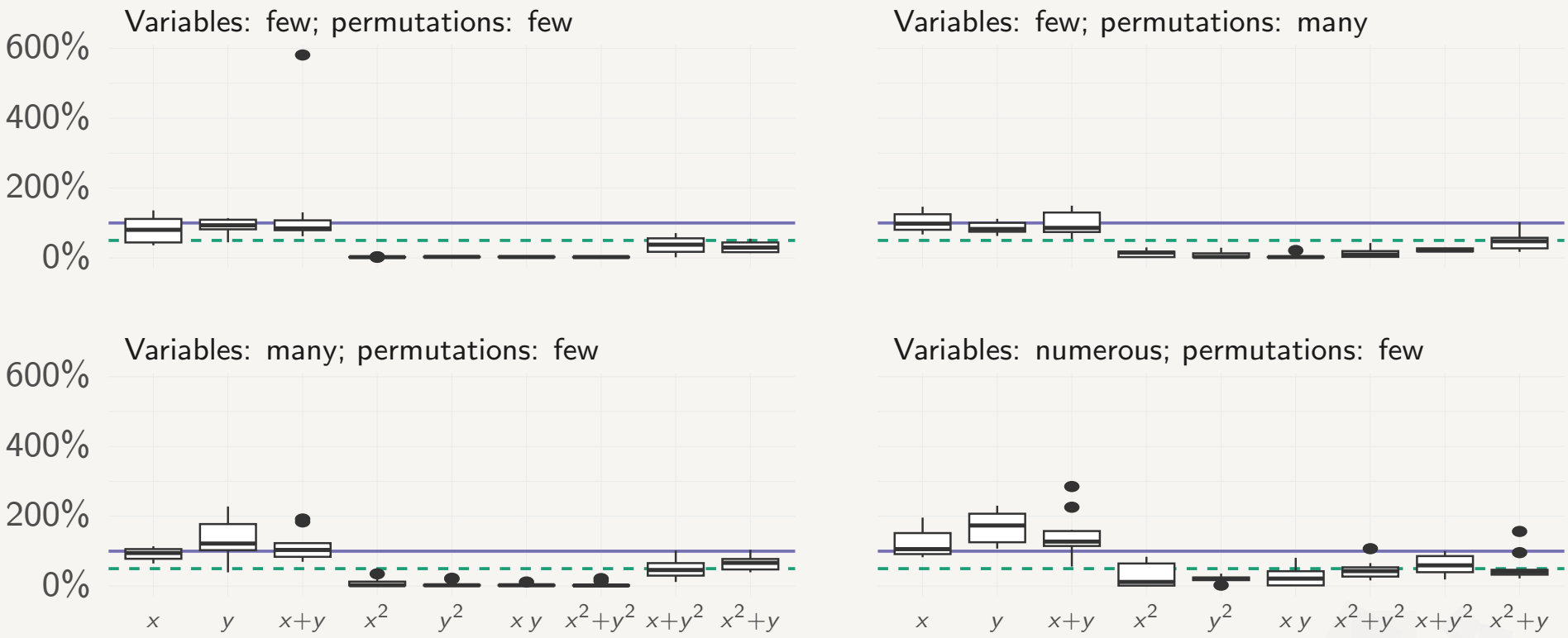
Experimental Results

Results for the 5 classes case only.



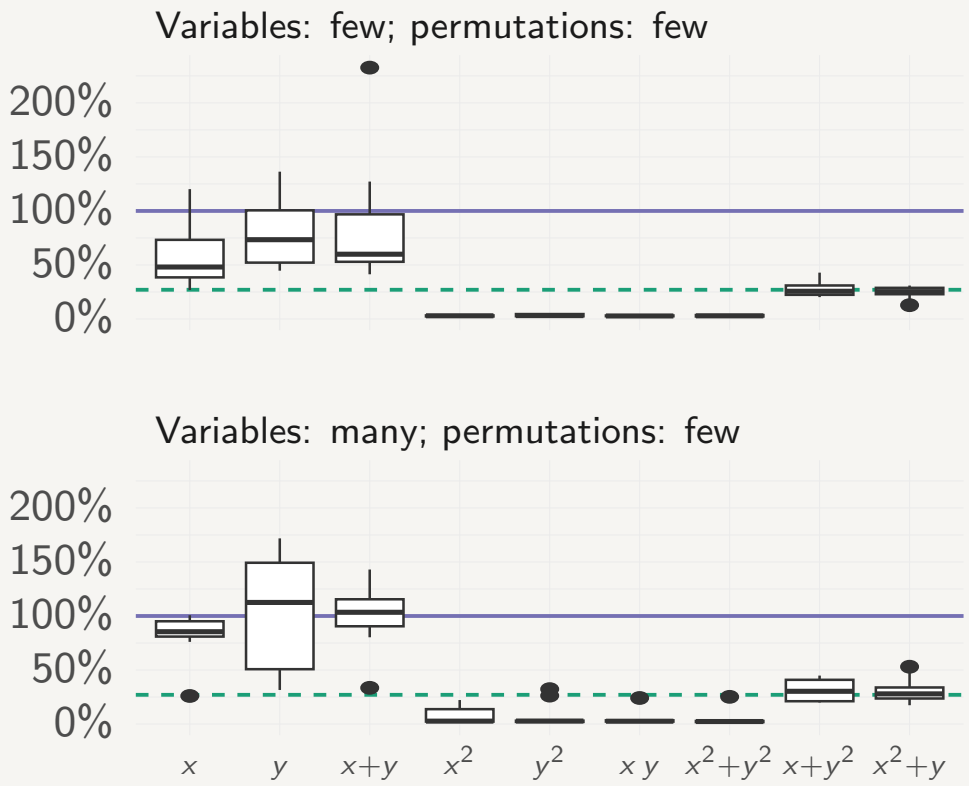
Experimental Results

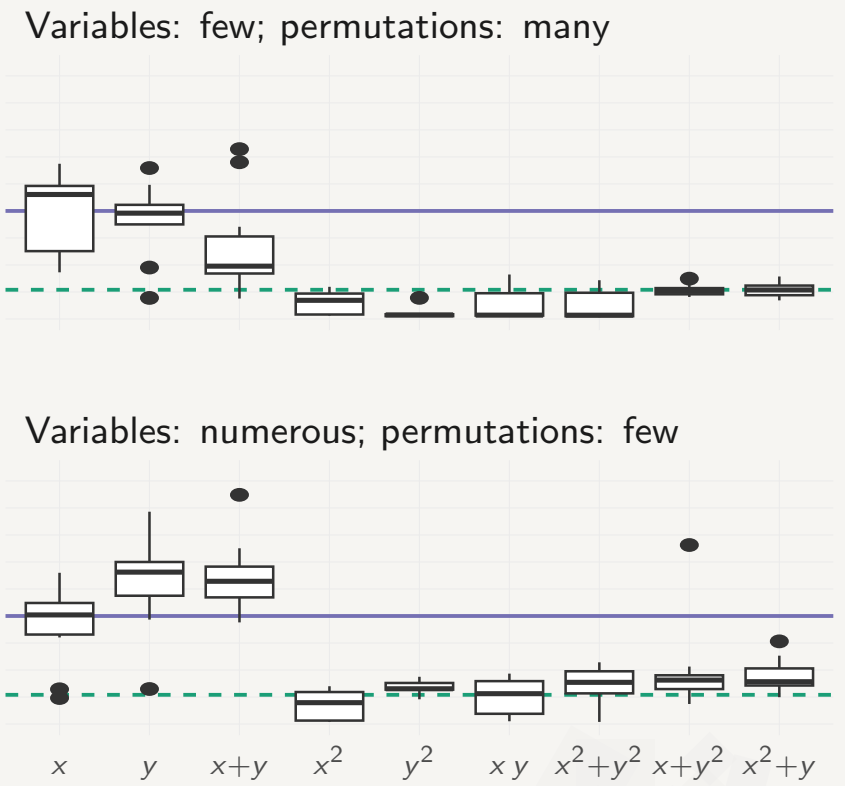
Results for the 7 classes case only.



Experimental Results

Results for the 9 classes case only.





Sketch of Proof for Main Theorem

Assume that x^* is an *optimal solution* for mimimizing $f(x)$ subject to $Ax \geq b$ and $x \in U^n$.

Let $G \subseteq \Pi_n$ be a (subgroup of the group of symmetries). Note that G is *finite*.

Let $h \in \mathbb{R}[x]$ and consider $H = \{h(Px^*) : P \in G\}$.

Since G contains symmetries, $APx^* \geq b$ and $f(Px^*) = f(x^*)$ for all $P \in G$. That is, H contains optimal solutions.

Since H is finite, it has a maximum $y = P^*x^* \in H$. For this maximum we have

$$\forall P \in G\colon h(Py) \leq h(y).$$

Thus, y is an optimal solution which fulfils $h(Py) - h(y) \leq 0$ for all $P \in G$.

References I

[1] E. Hubert, "Preserving and exploiting symmetry in algebraic computation," in *Proceedings of the 2024 International Symposium on Symbolic and Algebraic Computation*, 2024, pp. 13–13.

[2] I. P. Gent, K. E. Petrie, and J.-F. Puget, "Symmetry in constraint programming," in *Handbook of Constraint Programming*, F. Rossi, P. van Beek, and T. Walsh, Eds. Elsevier, 2006, pp. 329–376.

[3] T. Walsh, "Symmetry breaking constraints: Recent results," *arXiv preprint arXiv:1204.3348*, 2012.

[4] F. Margot, "Pruning by isomorphism in branch-and-cut," *Mathematical Programming*, vol. 94, no. 1, pp. 71–90, 2002.

[5] V. Kaibel and M. Pfetsch, "Packing and partitioning orbitopes," *Mathematical Programming*, vol. 114, no. 1, pp. 1–36, 2008.

[6] D. Salvagnin, "Symmetry breaking inequalities from the schreier-sims table," in *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Springer, 2018, pp. 521–529.

References II

[7] F. A. Aloul, I. L. Markov, and K. A. Sakallah, "Shatter: Efficient symmetry-breaking for boolean satisfiability," in *Proceedings of the 40th Design Automation Conference (DAC)*. ACM, 2003, pp. 836–839.

[8] J. M. Crawford, M. L. Ginsberg, E. M. Luks, and A. Roy, "Symmetry-breaking predicates for search problems," in *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR'96)*. Morgan Kaufmann, 1996, pp. 148–159.

[9] S. Dingliwal, R. Agarwal, H. Mittal, and P. Singla, "Advances in symmetry breaking for sat modulo theories," *arXiv preprint arXiv:1908.00860*, 2019. [Online]. Available: https://arxiv.org/abs/1908.00860

[10] Y. Faenza and V. Kaibel, "Extended formulations for packing and partitioning orbitopes," *Mathematics of Operations Research*, vol. 34, no. 3, pp. 686–697, 2009.

[11] L. Liberti and J. Ostrowski, "Stabilizer-based symmetry breaking constraints for mathematical programs," *Journal of Global Optimization*, vol. 60, pp. 183–194, 2014.

[12] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., 1990.

References III

[13] C. Muir, L. Marshall, and A. Toriello, "Temporal bin packing with half-capacity jobs," *INFORMS Journal on Optimization*, vol. 6, no. 1, pp. 46–62, 2024.

[14] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com

[15] Manual, CPLEX User's, "Ibm ilog cplex optimization studio," *Version*, vol. 12, no. 1987-2018, p. 1, 1987.

[16] L. De Moura and N. Bjørner, "Z3: An efficient SMT solver," in *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2008, pp. 337–340.

[17] L. Perron and V. Furnon, "OR-Tools," Google. [Online]. Available: https://developers.google.com/optimization/