Combining Zonotope Abstraction and Constraint Programming for Synthesizing Inductive Invariants

Bibek Kabi[1] Eric Goubault[1] Antoine Miné[2] Sylvie Putot[1]

[1]LIX, Ecole Polytechnique, CNRS, Institut Polytechnique de Paris, France

[2]Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, France

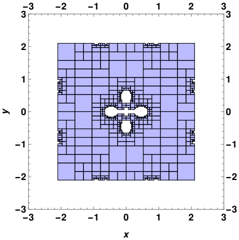
NSV 2020
13th International Workshop on Numerical Software Verification
20-21 July 2020 Los Angeles, CA, USA
https://nsv2020.github.io/

Objective

To verify if programs satisfy safety properties

Example

$x, y :=$ input $[0.9, 1.1]$
while true do
$\quad xnew := \frac{2x}{0.2 + x^2 + y^2 + 1.53 * x^2 * y^2}$
$\quad ynew := \frac{2y}{0.2 + x^2 + y^2 + 1.53 * x^2 * y^2}$
$\quad x := xnew \quad y := ynew$
done



What is an invariant?

An invariable property even after operations or transformations applied

Have they already been computed? and how?

Fixed-point computation [Cousot and Cousot (1977); Bradley (2011)];
Constraint-based techniques [Colón et al. (2003); Gulwani and Tiwari (2008)];

Introduction

Overview

- Motivating example
- Search algorithm
- Invariants of programs
- Conclusion

Motivating example: verification of safety property of a computer program

Motivating example: finding an invariant

Corner example [Martin (2014); Miné et al. (2016)]

$x :=$ input $[0.9, 1.1]$
$y :=$ input $[0.9, 1.1]$
while true do
  $xnew := \frac{2x}{0.2+x^2+y^2+1.53*x^2*y^2}$
  $ynew := \frac{2y}{0.2+x^2+y^2+1.53*x^2*y^2}$
  $x := xnew \quad y := ynew$
done

- initial values of (x, y) or entry states: $I \stackrel{\text{def}}{=} [0.9, 1.1] \times [0.9, 1.1]$
- loop effect on (x, y):

$$F : \mathcal{P}(\mathbb{R}^2) \to \mathcal{P}(\mathbb{R}^2)$$
$$F(X) \stackrel{\text{def}}{=} \{(\frac{2x}{0.2+x^2+y^2+1.53*x^2*y^2}, \frac{2y}{0.2+x^2+y^2+1.53*x^2*y^2})|(x, y) \in X\}$$

Method

To prove an invariant, look for an inductive invariant

Why is it so?

Given:

- the entry states, $I \subseteq \mathbb{R}^n$
- the transfer function, $F : \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$

Then:

- $G \subseteq \mathbb{R}^n$ is an inductive invariant if $I \subseteq G \wedge F(G) \subseteq G$
- $\mathrm{lfp}_I F$ being the smallest one (Tarski's theorem ensures the existence of a least fixpoint)

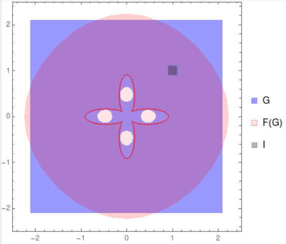
  $\mathrm{lfp}_I F$ is the least fixpoint of a functional F over a (sufficiently structured) partially-ordered domain of program states, defining the program semantics

- any $G \supseteq \mathrm{lfp}_I F$ is an invariant

Motivational example: absence of an inductive invariant

Corner example [Martin (2014); Miné et al. (2016)]

```
x=[0.9,1.1];
y=[0.9,1.1];
while (True) {
  xnew=2x/(0.2 + x^2 +
     y^2 + 1.53x^2y^2);
  ynew=2y/(0.2 + x^2 +
     y^2 + 1.53x^2y^2);
  x=xnew; y=ynew;}
```



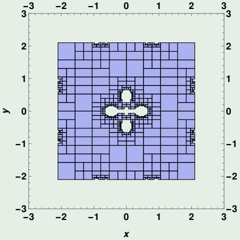
Lack of an inductive invariant

- $G = [-2.1, 2.1] \times [-2.1, 2.1]$ is an invariant
  all executions satisfy $(x,y) \in G$ at loop head, i.e., $\bigcup_{n \in \mathbb{N}} F^n(I) \subseteq G$
- $G = [-2.1, 2.1] \times [-2.1, 2.1]$ is not an inductive invariant
  $F(G) \not\subseteq G$, the problem!

Motivational example: the solution

Corner example [Martin (2014); Miné et al. (2016)]

```
x=[0.9,1.1];
y=[0.9,1.1];
while (True) {
  xnew=2x/(0.2 + x^2 +
    y^2 + 1.53x^2y^2);
  ynew=2y/(0.2 + x^2 +
    y^2 + 1.53x^2y^2);
  x=xnew; y=ynew;}
```



The solution

- search for a disjunction of boxes which is inductive
- search algorithm inspired by constraint programming

Search algorithm

Search algorithm

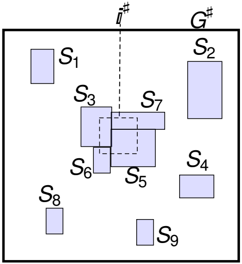
Developed by Miné et al. (2016) and inspired by continuous constraint solving [Rueher (2005); Pelleau et al. (2013)] it infers inductive invariants of numeric programs

Aim

To find a collection of abstract elements $\mathcal{S} = \{S_1, \cdots, S_n\}$ such that

- $I \subseteq \bigcup_i S_i$
- $\forall k : F^\sharp(S_k) \subseteq \bigcup_i S_i$
- $\forall i : S_i \subseteq T$, where T is target invariant or $G^\sharp$

$\implies \bigcup_i S_i$ is an inductive invariant



Prior work

Already combined with interval and octagon abstraction

Current work

- we want to adapt the combination of constraint solving with abstract interpretation for abstract domains which are not complete lattices
- here we deal with affine forms (zonotopes)
  - very good balance between complexity and precision
  - very interesting combinatorial structure that we will exploit
- operations we will revisit:
  - splitting (tiling)
  - inclusion test (improved the complexity)
  - meet (a geometrical meet taking into account all the faces at once)

Implementation

Prior work

- prototype analyzer
  - front end is the OCaml code implementing the algorithm
  - core mathematical functions are computed in Apron

Current work

- Implemented the operations in Taylor1+ [Ghorbal et al. (2009)] zonotope abstract domain in the APRON library `https://github.com/bibekkabi/taylor1plus`
- Implemented the OCaml binding for the splitting operator in Apron
- Adapted the prototype analyzer extending it to zonotope abstract domain and also to polyhedra

  `https://://github.com/bibekkabi/Prototype_analyzerwithApron`

Invariants of programs

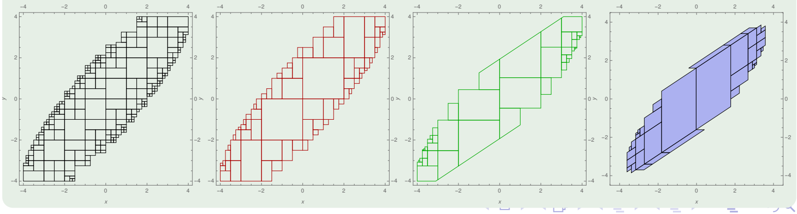
Invariants in Programs

Filter

$x :=$ input $[-0.1, 0.1]$
$y :=$ input $[-0.1, 0.1]$
while true do
  $r := 1.5x - 0.7y + [-0.1, 0.1]$
  $y := x \quad x := r$
done

- 238 boxes 1310 iterations, 0.1029 s
- 74 octagons, 736 iterations, 0.2105 s
- 42 polyhedra, 312 iterations, 0.2554 s
- 38 zonotopes, 222 iterations, 0.5020 s

Inductive invariant obtained for goal $x, y = [-4, 4]$

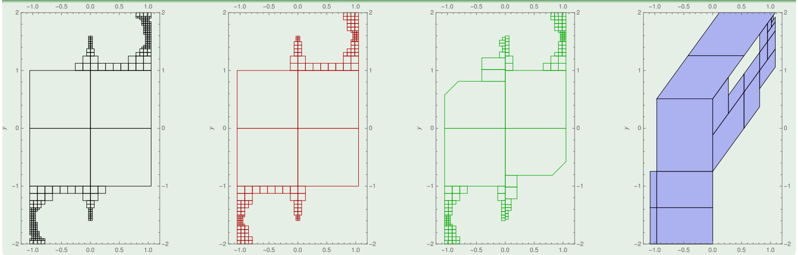


Sine

$x :=$ input $[-1.57079632679, 1.57079632679]$
$y :=$ input $[0, 0]$
while true do
  $y := x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040}$
done

- 240 boxes 1448 iterations, 0.4395 s
- 154 octagons, 348 iterations, 0.1102 s
- 136 polyhedra, 286 iterations, 1.1145 s
- 21 zonotopes, 33 iterations, 0.0547 s

Inductive invariant obtained for goal $x = [-2, 2]$, $y = [-1.05, 1.05]$



Benchmarks

Program	Boxes			Octagons			Zonotopes			Polyhedras		
	#elems.	#iters.	time(s)	#elems.	#iters.	time(s)	#elems.	#iters.	time(s)	#elems.	#iters.	time(s)
Octagon	752	2621	0.1042	752	2756	0.6115	**1**	**1**	**0.0001**	**1**	**1**	**0.0001**
Filter	238	1310	**0.1029**	74	736	0.2105	**38**	**222**	0.5020	42	312	0.2554
Arrow-Hurwicz	1784	1643	0.4033	369	931	0.5147	**15**	**38**	**0.0235**	134	484	1.0059
Filter2	14	58	0.0034	7	13	0.0013	8	16	0.0045	**1**	**1**	**0.0009**
Harm	87	438	**0.0112**	88	448	0.0647	60	254	0.5143	**53**	**243**	0.2442
Harm-reset	87	438	**0.0204**	88	446	0.1478	60	268	0.9717	**53**	**253**	0.3867
Harm-saturated	23	15	**0.0011**	24	16	0.0112	9	14	0.0157	**5**	**9**	0.0124
Lead-lag	-	-	-	-	-	-	-	-	-	-	-	-
Lead-lag-reset	-	-	-	-	-	-	-	-	-	-	-	-
Lead-lag-saturated	-	-	-	-	-	-	-	-	-	-	-	-
Sine	240	1448	0.4395	154	348	0.1102	**21**	**33**	**0.0547**	136	286	1.1145
Square root	7	10	0.0005	4	4	0.0016	**1**	**1**	**0.0001**	4	4	0.0066
Newton	200	102	0.1097	158	76	0.1785	**11**	**17**	**0.0197**	64	26	2.0660
Newton2	1806	499	6.6861	709	430	2.2207	**8**	**6**	**0.0193**	12	12	2.7498
Corner	129781	1847	646.8494	129767	1847	8850.8766	**488**	**999**	**35.6245**	2368	4248	126.7980

- zonotopes provide a good trade off in particular on non-linear programs
- they remain the most effective in showing that the initial invariant is correct

Conclusion

Conclusion

- Explored in detail the CP based AI approaches
- Extended an existing CP framework using zonotopes
- Tested it on non-linear programs

Thank you!

References I

Althoff, M. and Krogh, B. H. (2011). Zonotope bundles for the efficient computation of reachable sets. In *2011 50th IEEE conference on decision and control and European control conference*, pages 6814–6821. IEEE.

Bradley, A. R. (2011). Sat-based model checking without unrolling. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 70–87. Springer.

Colón, M. A., Sankaranarayanan, S., and Sipma, H. B. (2003). Linear invariant generation using non-linear constraint solving. In *Proceedings of CAV*, pages 420–432. Springer.

Cousot, P. and Cousot, R. (1977). Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of POPL*, pages 238–252. ACM.

References II

Ferrez, J.-A., Fukuda, K., and Liebling, T. M. (2005). Solving the fixed rank convex quadratic maximization in binary variables by a parallel zonotope construction algorithm. *European Journal of Operational Research*, 166(1):35–50.

Ghorbal, K., Goubault, E., and Putot, S. (2009). The zonotope abstract domain taylor1+. In *International Conference on Computer Aided Verification*, pages 627–633. Springer.

Ghorbal, K., Goubault, E., and Putot, S. (2010). A logical product approach to zonotope intersection. In *Proceedings of CAV*, pages 212–226.

Girard, A. and Le Guernic, C. (2008). Zonotope/hyperplane intersection for hybrid systems reachability analysis. In *International Workshop on Hybrid Systems: Computation and Control*, pages 215–228. Springer.

References III

Goubault, E. and Putot, S. (2015). A zonotopic framework for functional abstractions. *Formal Methods in System Design*, 47(3):302–360.

Guibas, L. J., Nguyen, A., and Zhang, L. (2003). Zonotopes as bounding volumes. In *Proceedings of the ACM-SIAM symposium on Discrete algorithms*, pages 803–812.

Gulwani, S. and Tiwari, A. (2008). Constraint-based approach for analysis of hybrid systems. In *International Conference on Computer Aided Verification*, pages 190–203. Springer.

Martin, B. (2014). *Rigorous algorithms for nonlinear biobjective optimization*. PhD thesis, Université de Nantes.

Miné, A., Breck, J., and Reps, T. (2016). An algorithm inspired by constraint solvers to infer inductive invariants in numeric programs. In *Proceedings of ESOP*, pages 560–588.

References IV

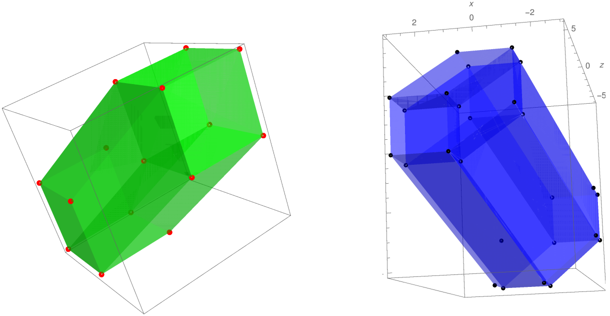
Pelleau, M., Miné, A., Truchet, C., and Benhamou, F. (2013). A constraint solver based on abstract domains. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 434–454. Springer.

Richter-Gebert, J. and Ziegler, G. M. (1994). Zonotopal tilings and the bohne-dress theorem. *Contemporary Mathematics*, 178:211–211.

Rueher, M. (2005). Solving continuous constraint systems. In *International Conference on Computer Graphics and Artificial Intelligence*, volume 1, pages 2–2.

Ziegler, G. M. and Richter-Gebert, J. (2017). 6: Oriented matroids. In *Handbook of Discrete and Computational Geometry, Third Edition*, pages 159–184. Chapman and Hall/CRC.

Appendix



3-dimensional parallelotopic tiles

Affine forms to zonotopes

Example:

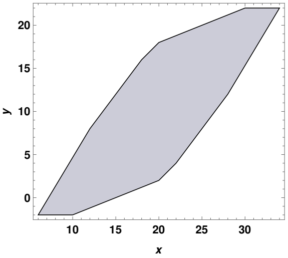
$\hat{x} = 20 - 3\varepsilon_1 + 5\varepsilon_2 + 2\varepsilon_3 + 1\varepsilon_4 + 3\varepsilon_5$

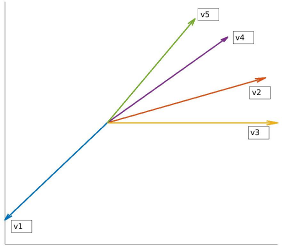
$\hat{y} = 10 - 4\varepsilon_1 + 2\varepsilon_2 \qquad + 1\varepsilon_4 + 5\varepsilon_5$

$$A^T = \begin{pmatrix} 20 & -3 & 5 & 2 & 1 & 3 \\ 10 & -4 & 2 & 0 & 1 & 5 \end{pmatrix}, n = 5, p = 2$$

Geometric concretisation: zonotope

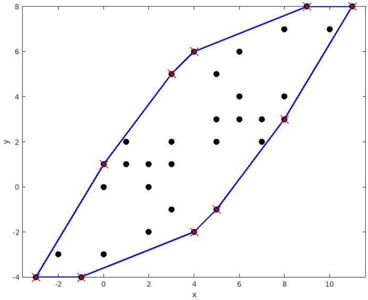
$$\gamma(A) = \left\{ A^T \begin{pmatrix} 1 \\ \varepsilon \end{pmatrix} | \varepsilon \in [-1,1]^n \right\} \subseteq \mathbb{R}^p, A \in \mathcal{M}(n+1, p)$$

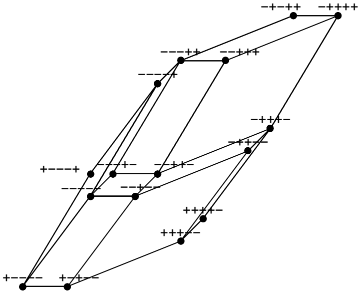




Zonotope tiling: a vertex enumeration problem

Motivation: can project 2^n vertices of n-hypercube with generator matrix





Non-extremal projections become the vertices of the tiles

Challenge

Tiling problem: to find sufficiently many of these not existant vertices

Zonotope: test for inclusion

State-of-the-art (Lemma 4 of Goubault and Putot (2015))

Let two matrices $X \in \mathcal{M}(n_X + 1, p)$ and $Y \in \mathcal{M}(n_Y + 1, p)$, then $\gamma(X) \subseteq \gamma(Y)$ if and only if for all $u \in \mathbb{R}^p$

$$\left| \sum_{i=1}^{p} (y_{(0,i)} - x_{(0,i)}) u_i \right| \leq ||Y_+ u||_1 - ||X_+ u||_1$$

Can this be improved? Yes! Complexity can be reduced to $2\binom{n}{p-1} \times \mathcal{O}(np)$

Lemma

For two zonotopes given by matrices $X \in \mathcal{M}(n_X + 1, p)$ and $Y \in \mathcal{M}(n_Y + 1, p)$, let $u = \{u_1, \cdots, u_k\}$ be vectors in $\mathbb{R}^p$ such that each face in $\gamma(Y)$ has a vector in u that is normal to it. Then $\gamma(X) \subseteq \gamma(Y)$ if and only if

$$\left| \langle u_i, c_x - c_y \rangle \right| \leq ||Y_+ u_i||_1 - ||X_+ u_i||_1, \forall i = 1, \cdots, k$$

where c_x, c_y are the centers of the zonotopes $\gamma(X)$, $\gamma(Y)$ respectively

State-of-the-art

- Sequence of meet of the zonotope with the faces of the other.
  - meet of a zonotope and a linear space geometrically [Girard and Le Guernic (2008)]
  - functional interpretation of the meet of a zonotope with a guard [Ghorbal et al. (2010)]
- Zonotope bundles can be expensive [Althoff and Krogh (2011)]

Solution

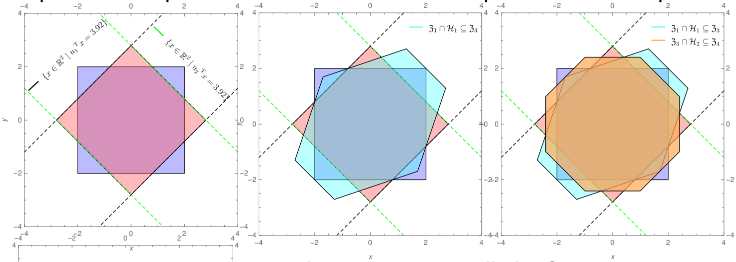
- a geometrical meet taking into account all the faces at once
- $\mathfrak{z}_1 \cap \mathfrak{z}_2 \subseteq \left\{ \alpha M_1^T \begin{pmatrix} 1 \\ e \end{pmatrix} + (1-\alpha) M_2^T \begin{pmatrix} 1 \\ e' \end{pmatrix}, ||e||_\infty \leq 1, ||e'||_\infty \leq 1 \right\}$

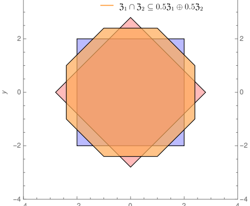
Zonotope: Meet

Example

$S_0 = [2\varepsilon_1, 2\varepsilon_2]^{\mathrm{T}}$ and $F^\sharp(S_0) = [1.4(\varepsilon_1 + \varepsilon_2), 1.4(\varepsilon_1 - \varepsilon_2)]^{\mathrm{T}}$, $S_0 \cap F^\sharp(S_0)$:

Sequential computation of meet of the zonotope and the half-spaces





← Take into account all the faces at once

$$\alpha \begin{pmatrix} 2\varepsilon_1 \\ 2\varepsilon_2 \end{pmatrix} + (1-\alpha) \begin{pmatrix} 1.4 \times (\varepsilon_3 + \varepsilon_4) \\ 1.4 \times (\varepsilon_3 - \varepsilon_4) \end{pmatrix}, \alpha = 0.5$$

Abstract domain

Bottleneck

Computing in $\mathcal{P}(\mathbb{R}^n)$ can be undecidable

Solution

Numerical abstract domain (approximation): : $\mathcal{D}^\sharp \subseteq \mathcal{P}(\mathbb{R}^n)$

- $\mathcal{D}^\sharp$ is a subset of properties of interest with a computer representation
- $F^\sharp : \mathcal{D}^\sharp \to \mathcal{D}^\sharp$ over-approximates the effect of $F : \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$

Numerical abstract domains

- Intervals/Boxes: $x \in [a, b]$
- Polytopes: H-representation (constraints) & V-representation (vertices)
- Octagons: $\pm x \pm y \leq a$
- Affine sets (zonotopes-center symmetric polytopes)

Abstract Interpretation [Cousot and Cousot (1977)]

Traditional Abstract Interpretation Approach

- to look for an abstract post-fixpoint of $F^\sharp$: $F^\sharp(X^\sharp) \subseteq^\sharp X^\sharp$, $I^\sharp \subseteq^\sharp X^\sharp$
- by iterating $F^\sharp$: $X^0 = I, \forall k. X^{k+1} = X^k \cup^\sharp F^\sharp(X^k)$ (Kleene iteration)

Disjunctive completion

- use $\mathcal{P}(\mathcal{D}^\sharp)$ instead of $\mathcal{D}^\sharp$
- synthesize finite collections $G^\sharp \subseteq \mathcal{D}^\sharp$ of abstract elements, $G^\sharp = \{S_1, \ldots, S_n\}$ that satisfies:

$$I \subseteq \bigcup_i S_i$$

$$\forall k : F^\sharp(S_k) \subseteq \bigcup_i S_i$$

$$\bigcup_i S_i \subseteq T$$

Search algorithm: overview

Given $F^\sharp$, $I^\sharp$, $G^\sharp$ such that $I^\sharp \subseteq G^\sharp$

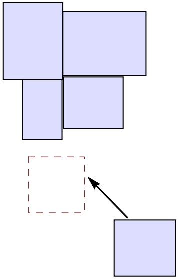
Algorithm [Miné et al. (2016)]

- begin with $\mathcal{S} \stackrel{\text{def}}{=} \{G^\sharp\}$
- while $\exists k : F^\sharp(S_k) \not\subseteq \cup_i S_i$

  either keep S_k, split S_k or discard S_k
- always, $I^\sharp \subseteq \cup_i S_i$
- stopping criteria: $\forall k : F^\sharp(S_k) \subseteq \cup_i S_i$

Doomed

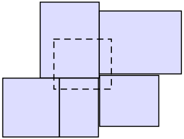
- $F^\sharp(S_k) \cap (\cup_i S_i) = \emptyset$
- such an abstract element is always discarded
- deciding whether S_k is doomed requires an intersection test



Such an element will always prevent inductiveness

Necessary

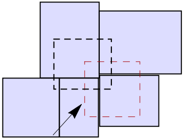
- $S_k \cap I \neq \emptyset$
- deciding whether S_k is necessary requires an intersection test



Such an element keeps ensuring that $I \subseteq \bigcup_i S_i$ always holds

Benign

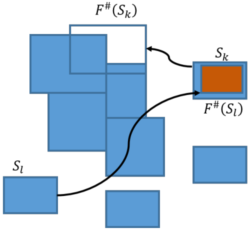
- $F^\sharp(S_k) \subseteq \bigcup_i S_i$
- deciding whether S_k is benign requires an inclusion checking



Such an element does not prevent inductiveness

Useful

- $S_k \cap (\bigcup_i F^\sharp(S_i)) \neq \emptyset$, i.e., an element of $G^\sharp$ relies on S_k to be benign
- deciding whether S_k is useful requires an intersection test

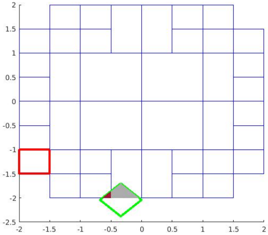


Coverage

- pick the S_k with the least coverage

$$\text{coverage}(S_k) := \frac{\sum_i vol(F^\sharp(S_k) \cap S_i)}{vol(F^\sharp(S_k))}$$

- ultimate aim is to have $\forall k$: coverage(S_k)$= 1$
  - $\forall k$: coverage(S_k)$= 1 \iff F^\sharp(S_k) \subseteq \bigcup_i S_i$
  - Note: S_i do not overlap



Search algorithm: prior work and current work?

New operations defined for zonotopes

- splitting
- meet

Operations improved for zonotopes

inclusion test

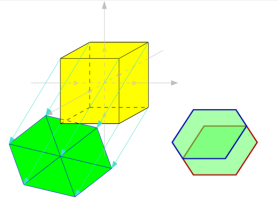
Operations improved for all domains

coverage metric

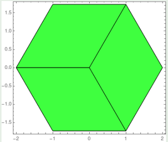
Zonotope: split

Splitting a zonotope

- by splitting the box which the zonotope is a projection of



- by tiling



Zonotope: splitting with overlap

Motivation

- splitting the j^{th} generator such that $\mathfrak{Z}_1 \cup \mathfrak{Z}_2 = \mathfrak{Z}$:
  $\mathfrak{Z}_1 = (c - \frac{g_j}{2}, < g_1, \cdots, \frac{g_j}{2}, \cdots, g_k >)$,
  $\mathfrak{Z}_2 = (c + \frac{g_j}{2}, < g_1, \cdots, \frac{g_j}{2}, \cdots, g_k >)$

Pros

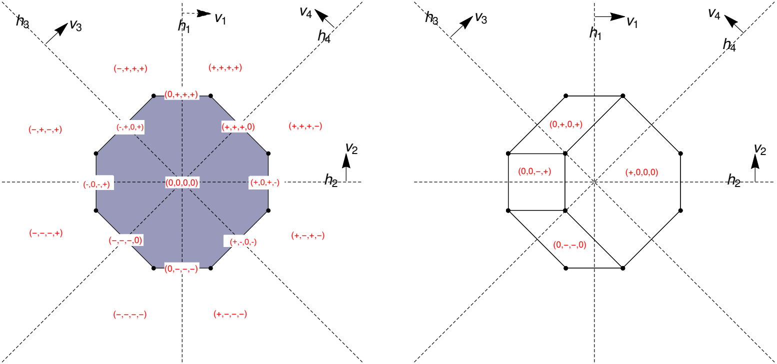
- simple, close to that on boxes
- keeps the same kind of shape
- keeps the direction of the faces fixed

Cons

- produces overlapping zonotopes
  - Note: the data structure of the algorithm requires that the S_i must not overlap
  - Note: needs a minor change in the algorithm; experimented it but not so efficient

Zonotope: polar dual of hyperplane arrangement [Ziegler and Richter-Gebert (2017); Richter-Gebert and Ziegler (1994)]

Any hyperplane partitions the space $\mathbb{R}^p$ into three sets:
$h_j^+ = \{x \mid v_j^{\mathrm{T}} x > b_j\}$, $h_j^0 = \{x \mid v_j^{\mathrm{T}} x = b_j\}$ and $h_j^- = \{x \mid v_j^{\mathrm{T}} x < b_j\}$



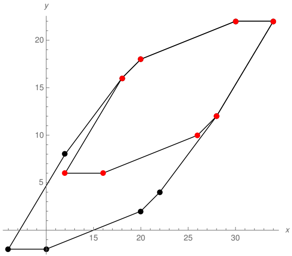
Characterizing a tile

The zero entries of the sign vectors (p generators) characterize the shape
The non-zero entries ($n - p$ generators) will decide the position

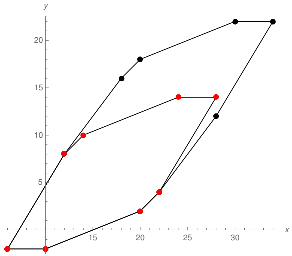
Definition

Let $\{+,-,0\}^n$ be a collection of sign vectors. A (single-element) fixing defines a sub-zonotope

$$\mathfrak{Z}(V \setminus j^{\{+,-\}}) := \sum_{i \in \{0\}^{(n-1)}} [-v_i, +v_i] + \sum_{i \in \{+,-\}} v_i - \sum_{i \in \{+,-\}} v_i$$

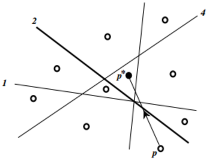


Fixing the first generator to '-'

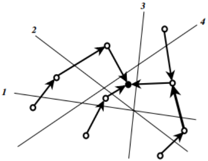


Fixing the first generator to '+'

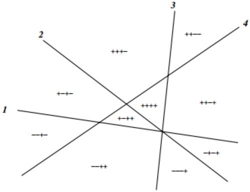
Zonotope: sign vector enumeration



Local search



Neighbor search



Sign vectors

Reverse search algorithm [Ferrez et al. (2005)]

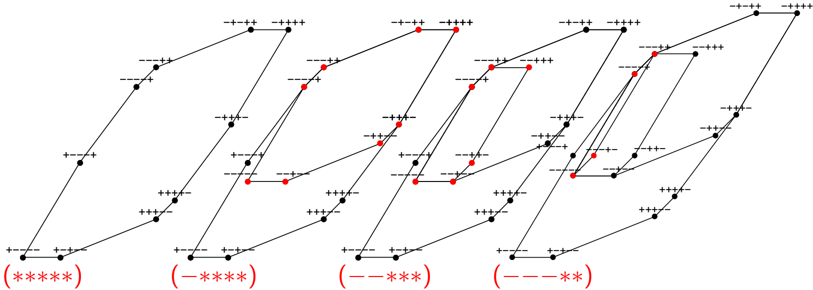
- A local search (f) to map any cell to an adjacent cell
- An Adjacency oracle (Adj) to return the set of neighbor cells of any given cell
- Visit all members by tracing the tree from the root
- Time complexity of $\mathcal{O}(n\ p\ LP(n, p)\ |\Sigma|)$ to compute $\Sigma{=}\Sigma(V)$

Algorithm

- While no. of generators is not equal to 2

  Keep fixing the sign of first generator

- Then finding all the adjacent p-parallelotopic tiles

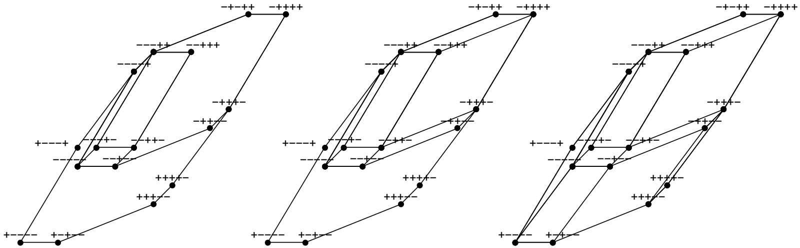


Zonotope: Tiling algorithm

Algorithm

- While no. of generators is not equal to 2
  - Keep fixing the sign of first generator

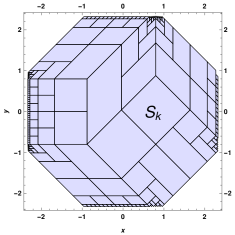
Then finding all the adjacent p-parallelotopic tiles



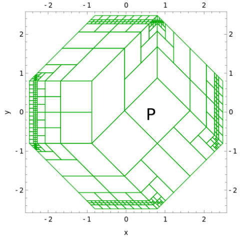
Tiles adjacent to (−−−∗∗) adjacent to (−−∗∗∗) adjacent to (−∗∗∗∗)

Partitioning data structure

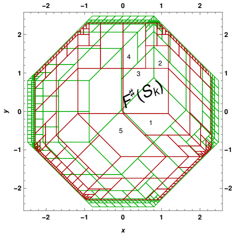
Perform updates efficiently without scanning $G^\sharp$ entirely after each operation



$G^\sharp$: inductive invariant



Partitions $B^\sharp$



Map post

$$\text{coverage}(S_k) := \frac{\#\{P \mid \text{cnt}(P) \neq \emptyset, P \in \text{post}(S_k)\}}{\#\{P \mid P \in \text{post}(S_k)\}}$$

$$S_k \text{is benign} \iff \forall P \in \text{post}(S_k) : \text{cnt}(P) \neq \emptyset \wedge F^\sharp(S_k) \subseteq T$$

Constraint programming

a method to solve Constraint Satisfaction Problems

CSP

- a set of variables $\mathcal{V} \stackrel{\text{def}}{=} \{v_1, \ldots, v_n\}$
- a set of initial domains $\mathcal{D} \stackrel{\text{def}}{=} \{D_1, \ldots, D_n\}$
  $\forall i : D_i \in \mathbb{R}$ or $\forall i : D_i \in \mathbb{Z}$
- and a set of constraints $\mathcal{C} \stackrel{\text{def}}{=} \{C_1, \ldots, C_p\}$ on $\mathcal{V}$

Solution to CSP

$$\mathcal{S} \stackrel{\text{def}}{=} \{x \in \mathcal{D} \mid \forall i : x \models \mathcal{C}_i\}$$

Zonotope: computing volume

Definition

The volume of a zonotope $\mathfrak{Z}(V)$ defined by a set of n vectors $V = \{v_1, \dots, v_n\}$ in p-dimension is given by $2^p \cdot \sum |\det(v_{i_1}, \dots, v_{i_p})|$

Example

Consider a zonotope with the set of vectors $V = ((-3,4),(5,2),(2,0),(1,1),(3,5))$.

$$vol(\mathfrak{Z}(V)) = \det\begin{pmatrix} -3 & 5 \\ -4 & 2 \end{pmatrix} + \det\begin{pmatrix} -3 & 2 \\ -4 & 0 \end{pmatrix} + \det\begin{pmatrix} -3 & 1 \\ -4 & 1 \end{pmatrix} + \det\begin{pmatrix} -3 & 3 \\ -4 & 5 \end{pmatrix} + \det\begin{pmatrix} 5 & 2 \\ 2 & 0 \end{pmatrix} + \det\begin{pmatrix} 5 & 1 \\ 2 & 1 \end{pmatrix} + \det\begin{pmatrix} 5 & 3 \\ 2 & 5 \end{pmatrix} + \det\begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix} + \det\begin{pmatrix} 2 & 3 \\ 0 & 5 \end{pmatrix} + \det\begin{pmatrix} 1 & 3 \\ 1 & 5 \end{pmatrix}$$

Problem

Calculating the coverage by computing the volume can be fairly expensive

Time complexity

$$\mathcal{O}(n\ p\ LP(n,p)\ |\Sigma|) + (n-2)\left[\mathcal{O}(p) + \mathcal{O}\left(2\binom{n}{p-1}\right)\right] + (n-2)\left[\binom{n}{p-1}\left\{\mathcal{O}\left(2\binom{n}{p-1}\right) + \mathcal{O}(p)\right\}\right]$$

- Finding the sign vectors of the original zonotope to be tiled using the reverse search

  $\mathcal{O}(n\ p\ LP(n,p)\ |\Sigma|)$

- Fixing the sign of the generators until the parallelotopic tile
  - We compute the sign vectors of each sub-zonotope and their centers
  - $(n-2)\left[\mathcal{O}(p) + \mathcal{O}\left(2\binom{n}{p-1}\right)\right]$
- Computing the parallelotopic tiles for each sub-zonotope

  $(n-2)\left[\binom{n}{p-1}\left\{\mathcal{O}\left(2\binom{n}{p-1}\right) + \mathcal{O}(p)\right\}\right]$

Test

- $\mathfrak{Z}_1 \cap \mathfrak{Z}_2 \neq \emptyset$ iff $c_1 - c_2 \in (0, \langle g_1, \cdots, g_k, h_1, \cdots, h_m \rangle)$ [Guibas et al. (2003)]
- Finding the values of the noise symbols by

$$\begin{aligned} \min \quad & c^{\mathrm{T}} x \\ \text{subject to} \quad & Ax = b \end{aligned}$$