A Logical Product Approach to Zonotope Intersection

Khalil Ghorbal, Eric Goubault, Sylvie Putot

Laboratory for the Modelling and Analysis of Interacting Systems
CEA, LIST, Boîte 94, Gif-sur-Yvette, F-91191 France.
firstname.lastname@cea.fr

Abstract. We define and study a new abstract domain which is a fine-grained combination of zonotopes with (sub-)polyhedric domains such as the interval, octagon, linear template or polyhedron domains. While abstract transfer functions are still rather inexpensive and accurate even for interpreting non-linear computations, we are able to also interpret tests (i.e. intersections) efficiently. This fixes a known drawback of zonotopic methods, as used for reachability analysis for hybrid systems as well as for invariant generation in abstract interpretation: intersection of zonotopes are not always zonotopes, and there is not even a best zonotopic over-approximation of the intersection. We describe some examples and an implementation of our method in the APRON library, and discuss some further interesting combinations of zonotopes with non-linear or non-convex domains such as quadratic templates and maxplus polyhedra.

1 Introduction

Zonotopic abstractions are known to give fast and accurate over-approximations in invariant synthesis for static analysis of programs, as introduced by the authors [10, 11, 7], as well as in reachability analysis of hybrid systems [8]. The main reason for this is that the interpretation of linear assignments is exact and done in linear time in terms of the "complexity" of the zonotopes, and non-linear expressions are dynamically linearized in a rather inexpensive way, unlike for most of other sub-polyhedric domains (zones [19], linear templates [21], even polyhedra [6]). But unions, at the exception of recent work [14], and more particularly intersections [9] are not canonical operations, and are generally computed using approximate and costly methods, contrarily to the other domains we mentioned. We present in this article a way to combine the best of the two worlds: by constructing a form of logical product [15] of zonotopes with any of these sub-polyhedric domains, we still get accurate and inexpensive methods to deal with the interpretation of linear and non-linear assignments, while intersections in particular, come clean thanks to the sub-polyhedric component of the domain.

Consider for instance the following program (loosely based on non-linear interpolation methods in e.g. embedded systems), which will be our running example:

```
real x = [0,10];
real y = x*x - x;
if (y >= 0) y = x/10; /* (x=0 or x >= 1) and y in [0,1] */
else y = x*x+2;       /* (x>0 and x<1) and y in [2,3] */
```

As indicated in the comments of the program, the `if` branch is taken when we have $x = 0$ or $x \geq 1$, so that y at the end of the program, is always in $[0, 3]$. Although this program looks quite simple, it is difficult to analyze, and the invariants found for y at the end of the program by classical domains[1] are disappointing: intervals, octagons, polyhedra, or zonotopes without constraint all find a range of values for y larger or equal than $[0, 102]$: even those which interpret quite accurately non-linear operations are not able to derive a constraint on x from the constraint on y. Whereas by the method proposed here, a logical product of zonotopes with intervals, in its APRON implementation, we find the much better range $[0, 9.72]$ (comparable to the exact result $[0, 3]$).

Contents of the paper We first introduce in Section 2 affine sets, a zonotopic abstract domain for abstract interpretation, that abstracts input/output relations in a program. We then introduce the problem of computing intersections in Section 3: starting with the running example, we define constrained affine sets as the combination of zonotopes with polyhedric domains and show they are well suited for the interpretation of tests. We then generalize the order on affine sets to constrained affine sets and define monotonic abstract transfer functions for arithmetic operators, that over-approximate the concrete semantics. Section 4 completes the definition of this new abstract domain: starting with the easier "one-variable" problem, we then give an algorithm for computing a join operator. We demonstrate the interest of the domain by describing in Section 5 the results on some examples, based on an implementation of our method in the library APRON. We conclude by a discussion of future work, including some further interesting combinations of zonotopes with non-linear or non-convex domains such as quadratic templates and maxplus polyhedra.

Related work In [17], the authors propose an approach based on a reduced product [5], to get more tractable and efficient methods for deriving sub-polyhedric invariants. But, still, the reduction algorithm of [17] is fairly expensive, and this domain also suffers from the drawbacks of polyhedra, in the sense that it is not well suited for efficiently and precisely deriving invariants for non-linear computations. Logical products in abstract interpretation are defined in [15]. The authors use the Nelson-Oppen combination method for logical theories, in the convex case, to get polynomial time abstractions on a much finer (than classical reduced products) combination of two abstract domains. As explained in Section 3.2, this approach does not directly carry over our case, because the theories we want to combine do not satisfy all the hypotheses of [15]. We thus choose in this paper a direct approach to the logical product of zonotopes with other classical abstract domains.

2 Affine sets: main definitions and properties

2.1 Affine arithmetic and zonotopes

Affine arithmetic is an extension of interval arithmetic on affine forms, first introduced in [4], that takes into account affine correlations between variables. An *affine form* is a

[1] The experiments were carried out using the domains interfaced within APRON [20].

formal sum over a set of *noise symbols* ε_i

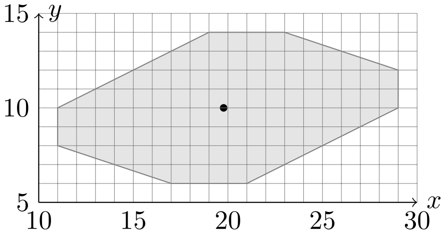
$$\hat{x} \stackrel{\text{def}}{=} \alpha_0^x + \sum_{i=1}^{n} \alpha_i^x \varepsilon_i,$$

with $\alpha_i^x \in \mathbb{R}$ for all i. Each noise symbol ε_i stands for an independent component of the total uncertainty on the quantity $\hat{x}$, its value is unknown but bounded in [-1,1]; the corresponding coefficient α_i^x is a known real value, which gives the magnitude of that component. The same noise symbol can be shared by several quantities, indicating correlations among them. These noise symbols can not only model uncertainty in data or parameters, but also uncertainty coming from computation. The semantics of affine operations is straightforward, non affine operations are linearized and introduce a new noise symbol: we refer the reader to [11, 13] for more details.

In what follows, we introduce matrix notations to handle tuples of affine forms. We note $\mathcal{M}(n,p)$ the space of matrices with n lines and p columns of real coefficients. A tuple of affine forms expressing the set of values taken by p variables over n noise symbols $\varepsilon_i,\ 1 \leq i \leq n$, can be represented by a matrix $A \in \mathcal{M}(n+1,p)$. We formally define the zonotopic concretization of such tuples by :

Definition 1. *Let a tuple of affine forms with* p *variables over* n *noise symbols, defined by a matrix* $A \in \mathcal{M}(n+1,p)$*. Its concretization is the zonotope*

$$\gamma(A) = \left\{ {}^tA^te \mid e \in \mathbb{R}^{n+1}, e_0 = 1, \|e\|_\infty = 1 \right\} \subseteq \mathbb{R}^p \ .$$



For example, for $n = 4$ and $p = 2$, the gray zonotope is the concretisation of the affine set $X = (\hat{x}, \hat{y})$, with $\hat{x} = 20 - 4\varepsilon_1 + 2\varepsilon_3 + 3\varepsilon_4$, $\hat{y} = 10 - 2\varepsilon_1 + \varepsilon_2 - \varepsilon_4$, and ${}^tA = \begin{pmatrix} 20 & -4 & 0 & 2 & 3 \\ 10 & -2 & 1 & 0 & -1 \end{pmatrix}$.

2.2 An ordered structure: affine sets

In order to construct an ordered structure preserving abstract input/output relations [14], we now define affine sets X as Minkowski sums of a *central* zonotope, $\gamma(C^X)$ and of a *perturbation* zonotope centered on 0, $\gamma(P^X)$. Central zonotopes depend on central noise symbols ε_i, whose interpretation is fixed once and for all in the whole program: they represent the uncertainty on input values to the program, with which we want to keep as many relations as possible. Perturbation zonotopes depend on perturbation symbols η_j which are created along the interpretation of the program and represent the uncertainty of values due to the control-flow abstraction, for instance while computing the join of two abstract values.

Definition 2. *We define an affine set* X *by the pair of matrices* $(C^X, P^X) \in \mathcal{M}(n+1,p) \times \mathcal{M}(m,p)$*. The affine form* $\pi_k(X) = c_{0k}^X + \sum_{i=1}^n c_{ik}^X \varepsilon_i + \sum_{j=1}^m p_{jk}^X \eta_j$*, where the* ε_i *are the central noise symbols and the* η_j *the perturbation or union noise symbols, describes the kth variable of* X*.*

We define an order on affine sets [7, 14] which is slightly more strict than concretization inclusion: it formalizes the fact that the central symbols have a specific interpretation as parametrizing the initial values of input arguments to the analyzed program:

Definition 3. *Let $X = (C^X, P^X)$, $Y = (C^Y, P^Y)$ be two affine sets in $\mathcal{M}(n+1, p) \times \mathcal{M}(m, p)$. We say that $X \leq Y$ iff*

$$\forall u \in \mathbb{R}^p, \; \|(C^Y - C^X)u\|_1 \leq \|P^Y u\|_1 - \|P^X u\|_1 \enspace .$$

It expresses that the norm of the difference $(C^Y - C^X)u$ for all $u \in \mathbb{R}^p$ is less than what the perturbation terms P^X and P^Y allow, that is the difference of the norms of $P^Y u$ with $P^X u$.

Classically, input/output functional abstractions are handled by adding slack variables corresponding to the initial values of the inputs. Here, we want relations between the variables of the program and the uncertain inputs, that is the inputs that create noise symbols. It can be proved that the relation of Definition 3 is equivalent to the geometric order on the larger zonotopes obtained by adding these slack variables to the zonotopes represented by our affine sets.

The binary relation $\leq$ of Definition 3 is a preorder, that we identify in the sequel with the partial order, quotient of this preorder by the equivalence relation[2] $X \sim Y$ iff by definition $X \leq Y$ and $Y \leq X$. Note also that this partial order is decidable, with a complexity bounded by a polynomial in p and an exponential in $n + m$. In practice, see [14], we do not need to use this costly general decision procedure.

3 *Constrained* affine sets for intersection

We now introduce the logical product of the domain $\mathcal{A}_1$ of Section 2 with any lattice, $(\mathcal{A}_2, \leq_2, \cup_2, \cap_2)$, used to abstract the values of the noise symbols ε_i and η_j. Formally, supposing that we have $n + 1$ noise symbols ε_i and m noise symbols η_j as in Section 2.2, we are given a concretization function: $\gamma_2 : \mathcal{A}_2 \to \mathcal{P}(\{1\} \times \mathbb{R}^n \times \mathbb{R}^m)$ and pseudo-inverse α_2. We now define constrained affine sets:

Definition 4. *A constrained affine set U is a pair $U = (X, \Phi^X)$ where $X = (C^X, P^X)$ is an affine set, and Φ^X is an element of $\mathcal{A}_2$. Equivalently, we write $U = (C^X, P^X, \Phi^X)$.*

Classical abstractions of "constraints" on the ε_i we will be using throughout this text are $\mathcal{A}$ consisting of products of $1 + n + m$ intervals (with the first one always being equal to 1), zones, octagons, and polyhedra (in the hyperplane $\varepsilon_0 = 1$).

3.1 Interpretation of tests

Equality tests on variables We first consider the case of the interpretation of equality test of two variables within an abstract state. Let us begin by a motivating example, which will make clear what the general interpretation of Definition 5 should be.

[2] It can be characterized by $C^X = C^Y$ and same concretizations for P^X and P^Y.

Example 1. Consider, with an interval domain for the noise symbols, $Z = [\![x_1 == x_2]\!]X$ where

$$\begin{cases} \Phi^X &= 1 \times [-1,1] \times [-1,1] \times [-1,1] \\ \hat{x}_1^X &= 4 + \varepsilon_1 + \varepsilon_2 + \eta_1, \quad \gamma(\hat{x}_1) = [1,7] \\ \hat{x}_2^X &= -\varepsilon_1 + 3\varepsilon_2, \quad \gamma(\hat{x}_2) = [-4,4] \end{cases}$$

We look for $\hat{z} = \hat{x}_1 = \hat{x}_2$, with $\hat{z} = z_0 + z_1\varepsilon_1 + z_2\varepsilon_2 + z_3\eta_1$. Using $\hat{x}_1 - \hat{x}_2 = 0$, i.e.

$$4 + 2\varepsilon_1 - 2\varepsilon_2 + \eta_1 = 0, \tag{1}$$

and substituting η_1 in $\hat{z} - \hat{x}_1 = 0$, we deduce $z_0 = 4z_3$, $z_1 = 2z_3 - 1$, $z_2 = -2z_3 + 3$. The abstraction in intervals of constraint (1) yields tighter bounds on the noise symbols: $\Phi^Z = 1 \times [-1,-0.5] \times [0.5,1] \times [-1,0]$. We now look for z_3 that minimizes the width of the concretization of z, that is $0.5|2z_3 - 1| + 0.5|3 - 2z_3| + |z_3|$. A straightforward $O((m+n)^2)$ method to solve the problem evaluates this expression for z_3 successively equal to 0, 0.5 and 1.5: the minimum is reached for $z_3 = 0.5$. We then have

$$\begin{cases} \Phi^Z = 1 \times [-1,-0.5] \times [0.5,1] \times [-1,0] \\ \hat{x}_1^Z = \hat{x}_2^Z (= \hat{z}) = 2 + 2\varepsilon_2 + 0.5\eta_1, \quad \gamma(\hat{x}_1^Z) = \gamma(\hat{x}_2^Z) = [2.5,4] \end{cases}$$

Note that the concretization $\gamma(\hat{x}_1^Z) = \gamma(\hat{x}_2^Z)$ is not only better than the intersection of the concretizations $\gamma(\hat{x}_1^X)$ and $\gamma(\hat{x}_2^X)$ which is $[1,4]$, but also better than the intersection of the concretization of affine forms $(\hat{x}_1^X)$ and $(\hat{x}_2^X)$ for noise symbols in Φ^Z. Note that there is not always a unique solution minimizing the width of the concretization.

In the following, we use bold letters to denote intervals, and for an interval $\boldsymbol{u} = [\underline{u}, \overline{u}]$, we note $\mathrm{dev}(\boldsymbol{u}) = \overline{u} - \underline{u}$.

Definition 5. *Let $X = (C^X, P^X, \Phi^X)$ a constrained affine set with $(C^X, P^X) \in \mathcal{M}(n+1,p) \times \mathcal{M}(m,p)$. We define $Z = [\![x_j == x_i]\!]X$ by:*
- $\Phi^Z = \Phi^X \bigcap \alpha_2 \left(\{(\varepsilon_1,\ldots,\varepsilon_n,\eta_1,\ldots,\eta_m) \mid (c^X_{0j} - c^X_{0i}) + \sum_{r=1}^n (c^X_{rj} - c^X_{ri})\varepsilon_r + \sum_{r=1}^m (p^X_{rj} - p^X_{ri})\eta_r = 0\}\right)$,
- $c^Z_{rl} = c^X_{rl}, \forall r \in \{0,\ldots,n\}$, and $\forall l \in \{1,\ldots,p\}, l \neq i,j$,
- $p^Z_{rl} = p^X_{rl}, \forall r \in \{1,\ldots,m\}$ and $\forall l \in \{1,\ldots,p\}, l \neq i,j$.
Let k such that $c^X_{kj} - c^X_{ki} \neq 0$, we define

$$c^Z_{li} = c^Z_{lj} = c^X_{li} + \frac{c^X_{lj} - c^X_{li}}{c^X_{kj} - c^X_{ki}}(c^Z_{ki} - c^X_{kj}) \ \forall l \in \{0,\ldots,n\},\ l \neq k, \tag{2}$$

$$p^Z_{li} = p^Z_{lj} = p^X_{li} + \frac{p^X_{lj} - p^X_{li}}{c^X_{kj} - c^X_{ki}}(c^Z_{ki} - c^X_{kj}) \ \forall l \in \{1,\ldots,m\}, \tag{3}$$

with c^Z_{ki} that minimizes $\sum_{l=1}^n |c^Z_{li}|\,\mathrm{dev}(\boldsymbol{\varepsilon_l^Z}) + \sum_{l=1}^m |p^Z_{li}|\,\mathrm{dev}(\boldsymbol{\eta_l^Z})$.
If for all k, $c^X_{kj} = c^X_{ki}$, then we look for r such that $p^X_{rj} - p^X_{ri} \neq 0$; if for all r, $p^X_{rj} = p^X_{ri}$ then $x_i = x_j$ and $Z = X$.

This expresses that the abstraction of the constraint on the noise symbols induced by the test is added to the domain of constraints, and the exact constraint is used to define an affine form z satisfying $z = x_j^Z = x_i^Z$, and such that $\gamma(z)$ is minimal. Indeed, let k such that $c_{kj}^X - c_{ki}^X \neq 0$, then $x_j == x_i$ allows to express ε_k as

$$\varepsilon_k = c_{0j}^X - c_{0i}^X + \sum_{1 \leq l \leq n, l \neq k} \frac{(c_{lj}^X - c_{li}^X)}{c_{ki}^X - c_{kj}^X} \varepsilon_l + \sum_{1 \leq l \leq m} \frac{(p_{lj}^X - p_{li}^X)}{c_{ki}^X - c_{kj}^X} \eta_l \quad . \tag{4}$$

We now look for $\pi_i(Z) = \pi_j(Z)$ equal to $\pi_i(X)$ and $\pi_j(X)$ under condition (4) on the noise symbols (where $\pi_k(X)$ describes the kth variable of X, as introduced in Definition 2). Substituting ε_k in for example $\pi_i(Z) = \pi_i(X)$, we can express, for all l, c_{li}^Z and p_{li}^Z as functions of c_{ki}^Z and get possibly an infinite number of solutions defined by (2) and (3) that are all equivalent when (4) holds. When condition (4) will be abstracted in a noise symbols abstract domain such as intervals, these abstract solutions will no longer be equivalent, we choose the one that minimizes the width of $\gamma(\pi_i(Z))$ which is given by $\sum_{l=1}^{n} |c_{li}^Z| \operatorname{dev}(\boldsymbol{\varepsilon_l^Z}) + \sum_{l=1}^{m} |p_{li}^Z| \operatorname{dev}(\boldsymbol{\eta_l^Z})$. This sum is of the form $\sum_{l=1}^{m+n} |a_l + b_l c_{ki}^Z|$, with known constants a_l and b_l. The minimization problem can be efficiently solved in $O((m+n)log(m+n))$ time, $m+n$ being the number of noise symbols appearing in the expressions of x_i and x_j, by noting that the minimum is reached for $c_{ki}^Z = -\frac{a_{l_0}}{b_{l_0}}$ for a $l_0 \in \{1, \ldots, m+n\}$. When it is reached for two indexes l_p and l_q, it is reached for all c_{ki}^Z in $\left[-\frac{a_{l_p}}{b_{l_p}}, -\frac{a_{l_p}}{b_{l_p}}\right]$, but we choose one of the bounds of this intervals because it corresponds to the substitution in x_i^Z of one of the noise symbols, and is in the interest for the interpretation of tests on expressions.

Equality tests on expressions Now, in the case of an equality test between arithmetic expressions, new constraints on the noise symbols can be added, corresponding to the equality of the two expressions interpreted as affine forms. We also choose new affine forms for variables appearing in the equality test: let $X = (C^X, P^X, \Phi^X)$ a constrained affine set with $(C^X, P^X) \in \mathcal{M}(n+1, p) \times \mathcal{M}(m, p)$. We define $Z = [\![exp1 == exp2]\!]X$ by: $Y_1 = [\![x_{p+1} = exp1]\!][\![x_{p+2} = exp2]\!]X$ using the semantics for arithmetic operations, as defined in section 3.3, then $Y_2 = [\![x_{p+1} == x_{p+2}]\!]Y_1$. Noting that one of the noise symbols appearing in the constraint introduced by the equality test, does not appear in $x_{p+1}^{Y_2} = x_{p+2}^{Y_2}$ as computed by Definition 5, using this constraint we substitute this noise symbol in the other variables in Y_2. We then eliminate the added variables x_{p+1} and x_{p+2} to obtain Z, in which $exp1 == exp2$ is thus algebraically satisfied.

Example 2. Consider $Z = [\![x_1 + x_2 == x_3]\!]X$ where

$$\begin{cases} \Phi^X = 1 \times [-1,1] \times [-1,1] \times [-1,1] \\ \hat{x}_1^X = 2 + \varepsilon_1, \quad \gamma(\hat{x}_1) = [1,3] \\ \hat{x}_2^X = 2 + \varepsilon_2 + \eta_1, \quad \gamma(\hat{x}_2) = [0,4] \\ \hat{x}_3^X = -\varepsilon_1 + 3\varepsilon_2, \quad \gamma(\hat{x}_3) = [-4,4] \end{cases}$$

We first compute $x_4 := x_1 + x_2$ in affine arithmetic: here, problem $x_4 == x_3$ is then the test we solved in example 1. The abstraction in intervals of constraint (1) yields $\Phi^Z = 1 \times [-1, -0.5] \times [0.5, 1] \times [-1, 0]$, and an affine form x_3^Z optimal in the sense of the width of its concretization, $x_3^Z = 2 + 2\varepsilon_2 + 0.5\eta_1$. Now, $\hat{x}_1^X + \hat{x}_2^X = \hat{x}_3^Z$ is satisfied when constraint (1) holds exactly, but not in its interval abstraction Φ^Z. But substituting ε_1 which does not appear in x_3^Z by $-2+\varepsilon_2-0.5\eta_1$ in $\hat{x}_1^X$ and $\hat{x}_2^X$, we obtain forms $\hat{x}_1^Z$ and $\hat{x}_2^Z$ that satisfy $x_1+x_2 == x_3$ in the abstract domain:

$$\begin{cases} \Phi^Z = 1 \times [-1,-0.5] \times [0.5,1] \times [-1,0] \\ \hat{x}_1^Z = \varepsilon_2 - 0.5\eta_1, \quad \gamma(\hat{x}_1) = [0.5,1.5] \\ \hat{x}_2^Z = 2+\varepsilon_2+\eta_1, \quad \gamma(\hat{x}_2) = [1.5,3] \\ \hat{x}_3^Z = 2+2\varepsilon_2+0.5\eta_1, \quad \gamma(\hat{x}_1^Z) = \gamma(\hat{x}_2^Z) = [2.5,4] \end{cases}$$

Inequality tests In the case of inequality tests, we only add constraints on noise symbols, for example for strict inequality:

Definition 6. *Let* $X = (C^X, P^X, \Phi^X)$ *a constrained affine set with* $(C^X, P^X) \in \mathcal{M}(n+1,p) \times \mathcal{M}(m,p)$. *We define* $Z = [\![exp1 < exp2]\!]X$ *by* $Z = (C^X, P^X, \Phi^Z)$*:*
$\Phi^Z = \Phi^X \bigcap \alpha_2 \left(\left\{ (\varepsilon_1, \ldots, \varepsilon_n, \eta_1, \ldots, \eta_m) \mid (c^Y_{0p+2} - c^Y_{0p+1}) + \right.\right.$
$\left.\left. \sum_{k=1}^n (c^Y_{kp+2} - c^Y_{kp+1})\varepsilon_k + \sum_{k=1}^m (p^Y_{kp+2} - p^Y_{kp+1})\eta_k < 0 \right\} \right)$,
where $Y = [\![x_{p+1} = exp1]\!][\![x_{p+2} = exp2]\!]X$.

3.2 Order relation

In a standard reduced product [5] of $\mathcal{A}_1$ with $\mathcal{A}_2$, the order relation would naturally be based on the component-wise ordering. But in such products, we cannot possibly reduce the abstract values so that to gain as much collaboration as needed between $\mathcal{A}_1$ and $\mathcal{A}_2$ for giving formal grounds to the reasoning of Example 1 for instance. What we really need is to combine the *logical theories* of affine sets, $Th(\mathcal{A}_1)$[3] with the one of quantifier-free linear arithmetic [18] over the reals, , $Th(\mathcal{A}_2)$[4], including all the domains we have in mind in this paper (intervals, zones, octagons, linear and non-linear templates, polyhedra). Look back at Example 1: we found a solution to the constraint $x_1 == x_2$ via a fine-grained interaction between the two theories $Th(\mathcal{A}_1)$ and $Th(\mathcal{A}_2)$. Unfortunately, the methods of [15] are not directly applicable; in particular $\mathcal{A}_1$ is not naturally expressible as a logical lattice - it is not even a lattice in general. Also, the signatures $\Sigma_{\mathcal{A}_1}$ and $\Sigma_{\mathcal{A}_2}$ share common symbols, which is not allowed in the approach of [15].

In order to compute the abstract transfer functions in the logical product $Th(\mathcal{A}_1) \cup Th(\mathcal{A}_2)$, we first define an order relation on the product domain $\mathcal{A}_1 \times \mathcal{A}_2$, that allows a fine interaction between the two domains. First, $X = (C^X, P^X, \Phi^X) \leq Y = (C^Y, P^Y, \Phi^Y)$ should imply that $\Phi^X \leq_2 \Phi^Y$, i.e. the range of values that noise symbols can take in form X is smaller than for Y. Then, we mean to adapt Definition 3 for noise symbols no longer defined in $[-1,1]$ as in the unconstrained case, but in the range of values Φ^X common to X and Y. Noting that:

$$\|C^X u\|_1 = \sup_{\epsilon_i \in [-1,1]} |\langle \epsilon, C^X u \rangle|,$$

where $\langle ., . \rangle$ is the standard scalar product of vectors in $\mathbb{R}^{n+1}$, we set:

[3] Signature $\Sigma_{\mathcal{A}_1}$ comprises equality, addition, multiplication by real numbers and real numbers.

[4] Signature $\Sigma_{\mathcal{A}_2}$ comprises $\Sigma_{\mathcal{A}_1}$ plus inequality and negation

Definition 7. *Let X and Y be two constrained affine sets. We say that $X \leq Y$ iff $\Phi^X \leq_2 \Phi^Y$ and, for all $t \in \mathbb{R}^p$,*

$$\sup_{(\epsilon,-)\in\gamma_2(\Phi^X)} |\langle (C^Y - C^X)t, \epsilon\rangle| \leq \sup_{(-,\eta)\in\gamma_2(\Phi^Y)} |\langle P^Y t, \eta\rangle| - \sup_{(-,\eta)\in\gamma_2(\Phi^X)} |\langle P^X t, \eta\rangle| \enspace .$$

The binary relation defined in Definition 7 is a preorder on constrained affine sets which coincides with Definition 3 in the "unconstrained" case when $\Phi^X = \Phi^Y = \{1\} \times [-1, 1]^{n+m}$. We use in the sequel its quotient by its equivalence relation, i.e. the partial order generated by it.

Definition 8. *Let X be a constrained affine set. Its concretization in $\mathcal{P}(\mathbb{R}^p)$ is*

$$\gamma(X) = \left\{ {}^t C^X \epsilon + {}^t P^X \eta \mid \epsilon, \eta \in \gamma_2(\Phi^X) \right\} \enspace .$$

For Φ^X such that $\gamma_2(\Phi^X) = \{1\} \times [-1, 1]^{n+m}$, this is equivalent to the concretization of the affine set (C^X, P^X) as defined in Section 2.2. As for affine sets [14], the order relation of Definition 7 is stronger than the geometric order: if $X \leq Y$ then $\gamma(X) \subseteq \gamma(Y)$. This allows for expressing functional dependencies between the input and current values of each variables as discussed in [14].

Note that γ is in general computable when $\mathcal{A}$ is a subpolyhedric domain (intervals, zones, octagons, linear templates and general polyhedra), as a linear transformation applied to a polyhedron. In the same case, the interval concretisation of X can be computed using any (guaranteed) solver for linear programs such as LURUPA [16], since it involves $2p$ (for p variables) linear programs:

$$\sup_{\epsilon,\eta\in\gamma(\Phi^X)} {}^t C^X \epsilon + {}^t P^X \eta, \text{ and } \inf_{\epsilon,\eta\in\gamma(\Phi^X)} {}^t C^X \epsilon + {}^t P^X \eta \enspace .$$

Of course, when $\mathcal{A}$ is the domain of intervals, this is done by a direct and easy calculation.

3.3 Semantics of arithmetic operations

Operations are not different than the ones generally defined on zonotopes, or on affine forms, see [4, 14], the only difference is in the multiplication where we use the constraints on ε_i and η_j to derive bounds for the non-linear part.

We note $[\![\text{new } \varepsilon_{n+1}]\!]_{\mathcal{A}_2} \Phi^X$ the creation of a new noise symbol ε_{n+1} with (concrete) values in $[-1, 1]$. We first define the assignment of a new variable x_{p+1} with a range of value $[a, b]$:

Definition 9. *Let $X = (C^X, P^X, \Phi^X)$ be a constrained affine set with $(C^X, P^X) \in \mathcal{M}(n+1, p) \times \mathcal{M}(m, p)$ and $a, b \in \mathbb{R}$. We define $Z = [\![x_{p+1} = [a, b]]\!] X$ where $(C^Z, P^Z) \in \mathcal{M}(n+2, p+1) \times \mathcal{M}(m, p+1)$ with : $\Phi^Z = [\![\text{new } \varepsilon_{n+1}]\!]_{\mathcal{A}_2} \Phi^X$, $C^Z =$*

$$\left(\begin{array}{c|c} & \frac{a+b}{2} \\ & 0 \\ C^X & \dots \\ & 0 \\ \hline 0 & \frac{|a-b|}{2} \end{array} \right), \; P^Z = \left(\begin{array}{c|c} & 0 \\ P^X & \dots \\ & 0 \end{array} \right) \enspace .$$

We carry on by addition, or more precisely, the operation interpreting the assignment $x_{p+1} := x_i + x_j$ and adding new variable x_{p+1} to the affine set:

Definition 10. *Let $X = (C^X, P^X, \Phi^X)$ be a constrained affine set where (C^X, P^X) is in $\mathcal{M}(n+1,p) \times \mathcal{M}(m,p)$. We define $Z = [\![x_{p+1} = x_i + x_j]\!]X = (C^Z, P^Z, \Phi^Z)$ where $(C^Z, P^Z) \in \mathcal{M}(n+1,p+1) \times \mathcal{M}(m,p+1)$ by $\Phi^Z = \Phi^X$ and*

$$C^Z = \left(C^X \middle| \begin{array}{c} c^X_{0,i} + c^X_{0,j} \\ \dots \\ c^X_{n,i} + c^X_{n,j} \end{array} \right) \text{ and } P^Z = \left(P^X \middle| \begin{array}{c} p^X_{1,i} + p^X_{1,j} \\ \dots \\ p^X_{m,i} + p^X_{m,j} \end{array} \right) .$$

The following operation defines the multiplication of variables x_i and x_j, appending the result to the constrained affine set X. All polynomial assignments can be defined using this and the previous operations.

Definition 11. *Let $X = (C^X, P^X, \Phi^X)$ be a constrained affine set where (C^X, P^X) is in $\mathcal{M}(n+1,p) \times \mathcal{M}(m,p)$. We define $Z = (C^Z, P^Z, \Phi^Z) = [\![x_{p+1} = x_i \times x_j]\!]X$ where $(C^Z, P^Z) \in \mathcal{M}(n+2,p+1) \times \mathcal{M}(m+1,p+1)$ by :*

- $\Phi^Z = [\![\mathit{new}\ \varepsilon_{n+1}]\!]_{\mathcal{A}_2} \circ [\![\mathit{new}\ \eta_{m+1}]\!]_{\mathcal{A}_2} \Phi^X$
- $c^z_{l,k} = c^x_{l,k}$ *and* $c^z_{n+1,k} = 0$ *for all* $l = 0, \dots, n$ *and* $k = 1, \dots, p$
- *Let* m_r *(resp.* μ_r*) be the* $(r+1)$*th coordinate (i.e. corresponding to* ε_r*) of* $\mathrm{mid}(\gamma(\Phi^X))$ *(resp. of* $\mathrm{dev}(\gamma(\Phi^X))$*), where* mid *(resp.* dev *) denotes the middle (resp. the radius) of an interval,* q_l *(resp.* χ_l*) be the* $(l+n+1)$*th coordinate (i.e. corresponding to* η_l*) of* $\mathrm{mid}(\gamma(\Phi^X))$ *(resp. of* $\mathrm{dev}(\gamma(\Phi^X))$*). Write* $d^x_i = c^x_{0,i} + \sum_{1 \le r \le n} c^x_{r,i} m_r + \sum_{1 \le l \le m} p^x_{l,i} q_l$*:*
  $c^z_{0,p+1} = d^x_i d^x_j - \sum_{1 \le r \le n} (d^x_i c^x_{r,j} + d^x_j c^x_{r,i}) m_r - \sum_{1 \le l \le m} (d^x_i p^x_{l,j} + d^x_i p^x_{l,i}) q_l + \sum_{1 \le r \le n} \frac{1}{2} c^x_{r,i} c^x_{r,j} \mu^2_r + \sum_{1 \le l \le m} \frac{1}{2} p^x_{l,i} p^x_{l,j} \chi^2_l$
- $c^z_{l,p+1} = d^x_i c^x_{l,j} + c^x_{l,i} d^x_j$ *for all* $l = 1, \dots, n$
- $c^z_{n+1,p+1} = \sum_{1 \le r \le n} \frac{1}{2} |c^x_{r,i} c^x_{r,j}| \mu^2_r + \sum_{1 \le r \ne l \le n} |c^x_{r,i} c^y_{l,j}| \mu_r \mu_l$
- $p^z_{l,k} = p^x_{l,k}$*,* $p^z_{m+1,k} = 0$ *and* $p^z_{l,p+1} = 0$*, for all* $l = 1, \dots, m$ *and* $k = 1, \dots, p$
- $p^z_{m+1,p+1} = \sum_{1 \le l \le m} |p^x_{l,i} p^x_{l,j}| \chi^2_l + \sum_{1 \le r \ne l \le m} |p^x_{r,i} p^x_{l,j}| \chi_r \chi_l + \sum_{0 \le r \le n}^{1 \le l \le m} (|c^x_{r,i} p^x_{l,j}| + |p^x_{l,i} c^x_{r,j}|) \mu_r \chi_l$.

The correctness of this abstract semantics stems from the fact that these operations are increasing functions over the set of constrained affine sets. For sub-polyhedric domains $\mathcal{A}_2$, m_r, q_l, μ_r and χ_l are easily computable, solving with a guaranteed linear solver the four linear programming problems $\sup_{\epsilon,\eta \in \gamma(\Phi^X)} \varepsilon_r$ (resp. inf) and $\sup_{\epsilon,\eta \in \gamma(\Phi^X)} \eta_l$ (resp. inf) - for an interval domain for $\mathcal{A}_2$, no such computation is needed of course.

Getting back to the running example of Section 1, in the `false` branch of the `if (y>=0)` test, we have to compute $y = x * x + 2$ with $x = 5 + 5\varepsilon_1$ and $\varepsilon_1 \in [-1, -0.444]$. Using Definition 11 which takes advantage of the bounds on ε_1 to get a better bound on the non-linear part (typically not possible if we had constructed a reduced product), we get $y = 14.93 + 13.9\varepsilon_1 + 0.96\varepsilon_3$ with $\varepsilon_3 \in [-1, 1]$. This gives $\gamma(y) = [0.07, 9.72]$, which is very precise since $\gamma(x) = [0, 2.77]$, hence we should ideally find $\gamma(y)$ in $\gamma(x) * \gamma(x) + 2 = [2, 9.72]$. Note that the multiplication given in Definition 11 and used here, is not the direct adaptation of the multiplication in the unconstrained case, that would give the much less accurate form $y = 41.97 + 50\varepsilon_1 + 10.03\varepsilon_3$: the better formulation is obtained by choosing an affine form that is a linearization of $x_i \times x_j$ no longer at 0, but at the center of the range of the constrained noise symbols.

4 Join operator on constrained affine sets

We first examine the easier case of finding a join operator for affine sets with just one variable, and $\mathcal{A}_2$ being the lattice of intervals. We then use the characterisations we find in this case to give efficient formulas for a precise (although over-approximated) join operator in the general case. We do not study here maximal lower bounds of affine sets, although they are naturally linked to the interpretation of tests, Section 3.1, this is outside the scope of this paper.

4.1 The one-dimensional case

In dimension one, constrained affine sets are simply *constrained affine forms*:

$$\hat{a} = \Big(\hat{a}(\epsilon) = \alpha_0^a + \sum_1^n \alpha_i^a \varepsilon_i,\ \beta^a,\ \Phi^a\Big),$$

where $\epsilon = (\varepsilon_1, \ldots, \varepsilon_n)^t$ belongs to Φ^a, and β^a is non negative. We use the bold face notation, $\boldsymbol{\varepsilon_i^a}$, to denote the interval concretization of ε_i. Let $\hat{a}$ and $\hat{b}$ be two constrained affine forms. Then $\hat{a} \leq \hat{b}$ in the sense of Definition 7 if and only if

$$\begin{cases} \Phi^a \subseteq \Phi^b \\ \sup_{\epsilon \in \Phi^a} |\hat{a}(\epsilon) - \hat{b}(\epsilon)| \leq \beta^b - \beta^a \end{cases}$$

In general, there is no least upper bound for two constrained affine forms, but rather, as already noted in the unconstrained case [13, 14], *minimal upper bounds*. A sufficient condition for $\hat{c}$ to be a minimal upper bound is to enforce a minimal concretization, that is, $\gamma(\hat{c}) = \gamma(\hat{a}) \cup \gamma(\hat{b})$, and then minimize β^c among upper bounds with this concretization.

Algorithm 1 computes this particular mub in some cases (when the first `return` branch is taken), and else an upper bound with minimal interval concretisation. Let us introduce the following notion used in the algorithm: let $\boldsymbol{i}$ and $\boldsymbol{j}$ be two intervals; $\boldsymbol{i}$ and $\boldsymbol{j}$ are said to be in generic position if ($\boldsymbol{i} \subseteq \boldsymbol{j}$ or $\boldsymbol{j} \subseteq \boldsymbol{i}$) imply ($\sup(\boldsymbol{i}) = \sup(\boldsymbol{j})$ or $\inf(\boldsymbol{i}) = \inf(\boldsymbol{j})$). We say by extension that two affine forms are in generic position if their interval concretizations are in generic position. The join algorithm is similar to the formula in the unconstrained case described in [14] except we have to be cautious about the relative position of the ranges of noise symbols.

Example 3. To complete the analysis of the running example of Section 1, the join of the abstract values for y on the two branches must be computed:

$$\begin{cases} \Phi^a = 1 \times [-1,1] \times [-1,1] \times [-1,1] \\ \hat{a} = 0.5 + 0.5\varepsilon_1 \\ \gamma(\hat{a}) = [0,1] \end{cases} \quad \begin{cases} \Phi^b = 1 \times [-1,-0.444] \times [-1,1] \times [-1,1] \\ \hat{b} = 14.93395 + 13.9\varepsilon_1 + 0.96605\varepsilon_3 \\ \gamma(\hat{b}) = [0.0679, 9.7284] \end{cases}$$

Algorithm 1: Join of two constrained affine forms

```
if $\hat{a}$ and $\hat{b}$ are in generic position then
    if $\mathrm{mid}(\gamma(\hat{b})) \leq \mathrm{mid}(\gamma(\hat{a}))$ then swap $\hat{a}$ and $\hat{b}$.
    for $i \geq 1$ do
        $\alpha_i^c \longleftarrow 0$
        if $\boldsymbol{\varepsilon_i^a}$ and $\boldsymbol{\varepsilon_i^b}$ are in generic position then
            if $\alpha_i^a \geq 0$ and $\alpha_i^b \geq 0$ then
                if $\mathrm{mid}(\boldsymbol{\varepsilon_i^a}) \leq \mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$ and $\mathrm{mid}(\boldsymbol{\varepsilon_i^b}) \geq \mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$ then
                    $\alpha_i^c \longleftarrow \min(\alpha_i^a, \alpha_i^b)$
            if $\alpha_i^a \leq 0$ and $\alpha_i^b \leq 0$ then
                if $\mathrm{mid}(\boldsymbol{\varepsilon_i^a}) \geq \mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$ and $\mathrm{mid}(\boldsymbol{\varepsilon_i^b}) \leq \mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$ then
                    $\alpha_i^c \longleftarrow \max(\alpha_i^a, \alpha_i^b)$
    if $0 \leq \sum_{i=1}^n \alpha_i^c(\mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b}) - \mathrm{mid}(\boldsymbol{\varepsilon_i^a})) \leq \mathrm{mid}(\gamma(\hat{a}) \cup \gamma(\hat{b})) - \mathrm{mid}(\gamma(\hat{a}))$ and
    $\mathrm{mid}(\gamma(\hat{a}) \cup \gamma(\hat{b})) - \mathrm{mid}(\gamma(\hat{b})) \leq \sum_{i=1}^n \alpha_i^c(\mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b}) - \mathrm{mid}(\boldsymbol{\varepsilon_i^b})) \leq 0$ then
        $\beta^c \longleftarrow \mathrm{dev}(\gamma(\hat{a}) \cup \gamma(\hat{b})) - \sum_{i=1}^n |\alpha_i^c| \,\mathrm{dev}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$
        $\alpha_0^c \longleftarrow \mathrm{mid}(\gamma(\hat{a}) \cup \gamma(\hat{b})) - \sum_{i=1}^n \alpha_i^c \,\mathrm{mid}(\boldsymbol{\varepsilon_i^a} \cup \boldsymbol{\varepsilon_i^b})$
        return $(\alpha_0^c, \alpha_1^c, \ldots, \alpha_n^c, \beta^c)$                    /* MUB */
$\beta^c \longleftarrow \mathrm{dev}(\gamma(\hat{a}) \cup \gamma(\hat{b}))$, $\alpha_0^c \longleftarrow \mathrm{mid}(\gamma(\hat{a}) \cup \gamma(\hat{b}))$, return $(\alpha_0^c, \beta^c)$        /* UB */
```

$\hat{a}$ and $\hat{b}$ are in generic positions, and so are ε_1^a and ε_1^b, but condition $\mathrm{mid}(\boldsymbol{\varepsilon_1^b}) \geq \mathrm{mid}(\boldsymbol{\varepsilon_1^a} \cup \boldsymbol{\varepsilon_1^b})$ is not satisfied, so that the join gives the following minimal upper bound:

$$\begin{cases} \Phi^c = 1 \times [-1,1] \times [-1,1] \times [-1,1] \times [-1,1] \\ \hat{c} = 4.8642 + 4.8642\eta_1, \ \gamma(\hat{c}) = [0, 9.7284] \end{cases}$$

Example 4. Let us now consider a second example:

$$\begin{cases} \Phi^a = 1 \times [-1,0] \times [-1,1] \\ \hat{a} = 1 + 2\varepsilon_1 - \varepsilon_2, \ \gamma(\hat{a}) = [-2,2] \end{cases} \quad \begin{cases} \Phi^b = 1 \times [-1,1] \times [0,0.5] \\ \hat{b} = 4 + 3\varepsilon_1 - \varepsilon_2, \ \gamma(\hat{b}) = [-2,7] \end{cases}$$

$\hat{a}$ and $\hat{b}$ are in generic positions, as well as ε_1^a and ε_1^b, while ε_2^a and ε_2^b are not; the join gives the following minimal upper bound:

$$\begin{cases} \Phi^c = 1 \times [-1,1] \times [-1,1] \times [-1,1] \\ \hat{c} = \frac{5}{2} + 2\varepsilon_1 + \frac{5}{2}\eta_1, \ \gamma(\hat{c}) = [-2,7] \end{cases}$$

4.2 Join operator in the general case

As in the unconstrained case [14], mubs for the global order on *constrained affine sets* are difficult to characterize. Instead of doing so, we choose in this paper to describe a simple yet efficient way of computing a good over-approximation of such mubs, relying on Algorithm 1 for mubs with minimal concretisation for *constrained affine forms*.

We first project the constrained affine forms defining each variable of the environment (the $\pi_k(X)$, for all k) by considering all noise symbols as if they were central noise symbols. We then use Algorithm 1 to independently compute a minimal upper bound for the constrained affine form defining each variable of the environment (on $\pi_k(X)$, for all k), and introduce a new noise symbol for each variable to handle the perturbation term computed in this Algorithm. We thus obtain an upper bound of the constrained affine set.

Example 5. Consider, for all noise symbols in $[-1, 1]$, constrained affine sets X and Y defined by $x_1 = 1 + \varepsilon_1,\ x_2 = 1 + \varepsilon_2$, and $y_1 = 1 + \eta_1,\ y_2 = 1 + \eta_1$. Considering first the 1D cases, we have $x_1 \leq y_1$ and $x_2 \leq y_2$. However we do not have $X \leq Y$ for the global order of Definition 7. Applying the join operator defined here on X and Y, we construct Z, defined by $z_1 = 1 + \eta_2$ and $z_2 = 1 + \eta_3$. We now have $X \leq Z$ and $Y \leq Z$.

5 Experiments

In this section, we compare results[5] we obtain with our new domain, called constrained T1+, in its APRON implementation, with the octagon and polyhedron APRON domains and the unconstrained T1+[7]. Our constrained T1+ implementation allows to choose as a parameter of the analysis, the APRON domain we want to use to abstract the constraints on noise symbols. However, at this stage, conditionals are interpreted only for the interval domain, we thus present results for this domain only.

Table 1 shows the numerical range of a variable of interest of each test case and for each domain, after giving the exact range we would hope to find. It can be noted that on these examples, constrained T1+ is always more accurate than octagons, and is also more accurate than polyhedra on non affine problems.

Table 1. Comparison of Constrained T1+ with APRON's abstract domains

	Exact	Octagons	Polyhedra	T1+	Cons. T1+
InterQ1	$[0, 1875]$	$[-3750, 6093]$	$[-2578, 4687]$	$[0, 2500]$	$[0, 1875]$
Cosine	$[-1, 1]$	$[-1.50, 1.0]$	$[-1.50, 1.0]$	$[-1.073, 1]$	$[-1, 1]$
SinCos	$\{1\}$	$[0.84, 1.15]$	$[0.91, 1.07]$	$[0.86, 1.15]$	$[0.99, 1.00]$
InterL2	$\{0.1\}$	$[-1, 1]$	$[0.1, 0.4]$	$[-1, 1]$	$[0.1, 1]$
InterQ2	$\{0.36\}$	$[-1, 1]$	$[-0.8, 1]$	$[-1, 1]$	$[-0.4, 1]$

In Table 1, `InterQ1` combines linear tests with quadratic expressions, only constrained T1+ finds the right upper bound of the invariant. `Cosine` is a piecewise 3rd order polynomial interpolation of the cosine function: once again, only constrained T1+ finds the exact invariant. The program `SinCos` computes the sum of the squares of the sine and cosine functions (real result is 1). `InterL2` (resp. `InterQ2`) computes a piecewise affine (resp. quadratic) function of the input, then focuses on the inverse image of 1 by this function.

[5] sources of the examples are available online `http://www.lix.polytechnique.fr/Labo/Khalil.Ghorbal/CAV2010`

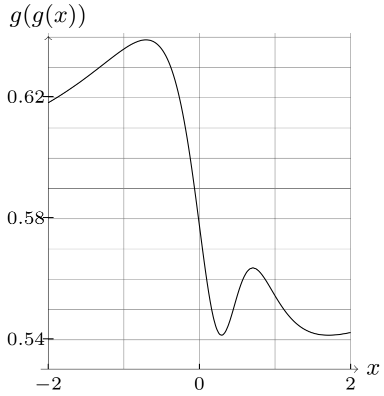
We now consider the computation of $g(g(x))$ on the range $x = [-2, 2]$, where

$$g(x) = \frac{\sqrt{x^2 - x + 0.5}}{\sqrt{x^2 + 0.5}}.$$

We parametrize the program that computes $g(g(x))$ by a number of tests that subdivide the domain of the input variable (see Figure 1 left for a parametrization by n subdivisions), in order to compare the relative costs and precisions of the different domains when the size of the program grows.

Fig. 1. Implementation of g(g(x)) for x in [-2,2] (left) and plot of g(g(x)) (right)

```
g(x) = sqrt(x*x-x+0.5)/sqrt(x*x+0.5);
x = [-2,2];
/* for n subdivisions */
h = 4/n;
if (-x<=h-2)
  y = g(x); z = g(y);
...
else if (-x<=i*h-2) /* i in {2,...,n-1}*/
  y = g(x); z = g(y);
...
else
  y = g(x); z = g(y);
```

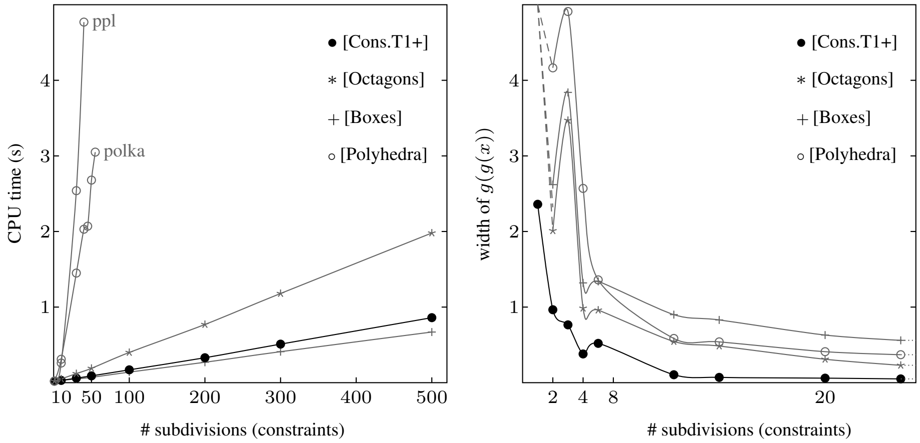


It can be noted (Figure 2 left) that our domain scales up well while giving here more accurate results (Figure 2 right) than the other domains. As a matter of fact, with an interval domain for the noise symbols, all abstract transfer functions are linear or at worst quadratic in the number of noise symbols appearing in the affine forms. Notice also that our implementations detects the squares of variables, which allows constrained T1+ to give $[0, 4.72]$ without subdivisions while all other domains end with $[-\infty, +\infty]$ (noted by the dotted lines on Figure 2 right). The fact that the results observed for 3 and 5 subdivisions (Figure 2 right) are less accurate respectively than those observed for 2 and 4 subdivisions, is related to the behaviour of $g(g(x))$ on $[-2, 2]$ (see Figure 1 right): for example when a change of monotony appears near the center of a subdivision, the approximations will be less accurate than when it appears at the border.

6 Conclusion, and future work

In this paper, we studied the logical product of the domain of affine sets with sub-polyhedric domains on noise symbols, although the framework as described here is much more general. We concentrated on such abstract domains for $\mathcal{A}$ for practical reasons, in order to have actual algorithms to compute the abstract transfer functions.

Fig. 2. Comparing analysis time and results of the different APRON domains.



However, in some embedded control systems, quadratic constraints appear already on the set of initial values to be treated by the control program, or as a necessary condition for behaving well, numerically speaking. For example in [3], as in a large class of navigation systems, the control program manipulates normalized quaternions, that describe the current position in 3D, of an aircraft, missile, rocket etc. We think that a combination of zonotopes with quadratic templates [1] in the lines of this article would be of interest to analyze these programs.

Also, as noticed in [2], maxplus polyhedra encompass a large subclass of disjunctions of zones; hence, by combining it with affine sets, we get another rather inexpensive way to derive a partially disjunctive analysis from affine forms (another with respect to the ideas presented in [13]).

Another future line of work is to combine the ideas of this paper with the ones of [12] to get better under-approximation methods in static analysis.

Acknowledgments. This work was partially funded by the French national research agency (ANR) projects ASOPT (Analyse Statique et OPTimisation) and Eva-Flo (Evaluation et Validation Automatique pour le Calcul Flottant) as well as by DIGITEO project PASO.

References

[1] A. Adje, S. Gaubert, and E. Goubault. Coupling policy iteration with semi-definite relaxation to compute accurate numerical invariants in static analysis. In *Proceedings of European Symposium On Programming - to appear*, 2010.

[2] X. Allamigeon, S. Gaubert, and E. Goubault. Inferring Min and Max Invariants Using Max-plus Polyhedra. In *Proceedings of the 15th International Static Analysis Symposium, LNCS 5079*, pages 189–204. Springer Verlag, 2008.

[3] O. Bouissou, E. Conquet, P. Cousot, R. Cousot, J. Feret, K. Ghorbal, E. Goubault, D. Lesens, L. Mauborgne, A. Mine, S. Putot, and X. Rival. Space software validation using abstract interpretation. In *Proceedings of the Int. Space System Engineering Conference, Data Systems in Aerospace DASIA'09*, 2009.

[4] J. L. D. Comba and J. Stolfi. Affine arithmetic and its applications to computer graphics. *Proceedings of SIBGRAPI*, 1993.

[5] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proceedings of Principles Of Programming Languages*, pages 269–282. ACM Press, 1979.

[6] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Proceedings of Principles Of Programming Languages*, pages 84–96. ACM Press, 1978.

[7] K. Ghorbal, E. Goubault, and S. Putot. The zonotope abstract domain Taylor1+. In *Proceedings of Computer Aided Verification, LNCS 5643*, pages 627–633. Springer-Verlag, 2009.

[8] A. Girard. Reachability of uncertain linear systems using zonotopes. In *Proceedings of Hybrid Systems: Computation and Control, LNCS 3414*, pages 291–305. Springer-Verlag, 2005.

[9] A. Girard and C. Le Guernic. Zonotope/hyperplane intersection for hybrid systems reachability analysis. In *Proceedings of the 11th international workshop on Hybrid Systems: Control and Computation, LNCS 4981*, pages 215–228. Springer-Verlag, 2008.

[10] E. Goubault and S. Putot. Weakly relational domains for floating-point computation analysis. In *Presented at the second international workshop on Numerical and Symbolic Abstract Domains, also available at http://www.di.ens.fr/˜goubault/papers/NSAD05.pdf*, 2005.

[11] E. Goubault and S. Putot. Static analysis of numerical algorithms. In *Proceedings of Static Analysis Symposium, LNCS 4134*, pages 18–34. Springer-Verlag, 2006.

[12] E. Goubault and S. Putot. Under-approximations of computations in real numbers based on generalized affine arithmetic. In *Proceedings of Static Analysis Symposium, LNCS 4634*, pages 137–152. Springer-Verlarg, 2007.

[13] E. Goubault and S. Putot. Perturbed affine arithmetic for invariant computation in numerical program analysis. *CoRR*, abs/0807.2961, available at http://arxiv.org/abs/0807.2961, 2008.

[14] E. Goubault and S. Putot. A zonotopic framework for functional abstractions. *CoRR*, abs/0910.1763, available at http://arxiv.org/abs/0910.1763, 2009.

[15] S. Gulwani and A. Tiwari. Combining abstract interpreters. In *Proceedings of the ACM SIGPLAN conference on Programming language design and implementation*, pages 376–386. ACM Press, 2006.

[16] C. Keil. Lurupa - rigorous error bounds in linear programming. In *Algebraic and Numerical Algorithms and Computer-assisted Proofs, Dagstuhl Seminar 5391*, 2005.

[17] V. Laviron and F. Logozzo. Subpolyhedra: A (more) scalable approach to infer linear inequalities. In *Proceedings of Verification Model-Checking and Abstract Interpretation, LNCS 5403*, pages 229–244. Springer-Verlag, 2009.

[18] Z. Manna and A. R. Bradley. *The Calculus of Computation; Decision procedures with applications to verification*. Springer-Verlag, 2007.

[19] A. Miné. A new numerical abstract domain based on difference-bound matrices. In *Proceedings of the Second Symposium on Programs as Data Objects, LNCS 2053*, pages 155–172. Springer-Verlag, 2001.

[20] APRON Project. Numerical abstract domain library, 2007. `http://apron.cri.ensmp.fr`.

[21] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Scalable analysis of linear systems using mathematical programming. In *Proceedings of Verification Model-Checking an Abstract Interpretation, LNCS 3385*, pages 25–41, 2005.