Acta Cybernetica **26** (2024) 755–774.

A Constraint Programming Approach for Polytopic Simulation of Ordinary Differential Equations — A Collision Detection Application*

Julien Alexandre dit Sandretto[ab], Alexandre Chapoutot[ac], Christophe Garion[de], and Xavier Thirioux[df]

Abstract

This paper presents a constraint-based approach to compute the reachable tube of nonlinear differentiable equations. A set of initial values for the equations is considered and defined by a polytope represented as intersections of zonotopes. Guaranteed numerical integration based on zonotopic computation is used to compute reachable tubes. In order to efficiently build polytopes defined by the intersection of several zonotopes, we use a previously developed abstract domain [27] to represent reachable tubes. The proposed contribution allows to compute more expressive reachable tubes more efficiently than methods based only on boxes, and therefore could improve verification/validation processes in robotics application for example. The approach is evaluated on examples taken from literature and we present two applications of this work.

Keywords: constraint programming, abstract domains, ordinary differential equations, cyber-physical systems, abstract interpretation

1 Context and state of the art

Cyber-physical systems (CPS) are systems in which software and physical parts interoperate deeply. The physical part of these systems is often modeled by differential equations. When properties have to be verified on these systems, for instance the feasibility or the safety of a mission assigned to a robot, the solution of such differential equations is generally required. Even if Ordinary Differential Equations

*This work was supported by the Defense Innovation Agency (AID) of the French Ministry of Defense (research project N° 2018 60 0072 00 470 75 01).

[a]ENSTA Paris, Institut Polytechnique de Paris, Palaiseau, France

[b]E-mail: alexandre@ensta.fr, ORCID: 0000-0002-6185-2480

[c]E-mail: chapoutot@ensta.fr, ORCID: 0000-0002-7230-0710

[d]ISAE-SUPAERO, Université de Toulouse, France

[e]E-mail: garion@isae-supaero.fr, ORCID: 0000-0002-4467-2939

[f]E-mail: thirioux@isae-supaero.fr, ORCID: 0009-0002-1126-6835

DOI: 10.14232/actacyb.300771

(ODE) are mostly considered to model cyber-physical systems, obtaining an analytical solution to this class of equations is a complex issue and approximations obtained with numerical methods are sometimes sufficient to check a given property. However, for some applications, such as [9, 14, 17, 23], an approximation is not enough and an enclosure of the exact solution is required.

Starting from a set of possible initial points, the solution of an ODE can be represented by a *reachable tube* describing the evolution of the system from this initial set. To ease the computation of such a tube, *abstract domains* can be used to enclose it: *boxes*, *zonotopes*, *ellipsoids*, and nonconvex sets such as *Taylor models* (see [6] for a recent review of the use of such abstract domains in set-based simulation). The more accurate an abstract domain is, *i.e.* the smallest the difference between the hull of the abstraction and the abstracted set is, the more accurate the enclosure of the reachable tube will be and therefore will be useful for verification purposes for instance.

However, only few of these set abstractions come with an associated arithmetic which is mandatory to evaluate nonlinear expressions[1]. Among such abstractions, boxes are mainly used because the underlying computation process, *i.e.*, interval arithmetic, is easy to use [19]. Zonotopes are also a good representation choice because they rely on affine arithmetic [13] which is also efficiently computable. Polytope enclosure is a promising approach as it is more precise than boxes and zonotopes, but suffers from the expensiveness of its geometrical computation as many linear programs need to be solved to obtain the set described by the sides of the polytope. Considering a polytope as *an intersection of zonotopes* and therefore benefiting from affine arithmetic is a possible solution to overcome the limitations of polytopes. To do so, two main techniques exist: i) zonotope bundles [7], *i.e.*, a set of zonotopes is used so the intersection is not computed; or ii) the intersection is computed when necessary [4].

In this paper, we propose a novel approach based on constraint programming to compute polytopes-based reachable tubes of ODEs in two phases. First a constraint satisfaction problem (mixing disjunctions and conjunctions) is generated from the tubes computed with zonotopes-based simulation. This CSP is thus a novel way to describe a reachable tube (that may or not be evaluated). Our proposed CSP approach also simplifies the computation of intersection of zonotopes. Second, these CSPs can be combined to represent intersection or union of trajectories, collision with obstacles, or other kind of properties that can be modelled by constraints. Finally, the global CSP is solved using a previously designed abstract domains that take advantage of the specificities of ODEs [27], *i.e.*, their continuous aspect and the fact that their abstractions as tubes can be naturally expressed as disjunctions. This approach allows to solve nonlinear ODEs with uncertain initial conditions, and can address several types of problems such as optimal control or safety verification.

However, while several papers deal with constraint programming combined with differential equations, such as [12, 16, 18] which address parameter identification,

[1] Minkowski sum may be then used but comes with all its limitations: only for linear operations, based on convex decomposition, etc

[22] which proposes to solve more complex differential equations, or [3] which considers problems in robotics, we propose in this paper to exploit the CSP framework to obtain a more expressive representation of sets to enclose reachable tubes.

This paper is organized as follows. Section 2 presents the main mathematical tools used in the remaining of the paper. Section 3 presents how to compute intersection of zonotopes using the Constraint Programming framework. Section 4 details experiments on three small size problems and two applications. Finally, Section 5 gives some perspectives for future work.

2 Prerequisite concepts

2.1 Polytopes and zonotopes

Considering the Euclidean space $\mathbb{R}^n$, a *convex hull* of a finite set $\{v_1, \dots, v_N\}$ of points in $\mathbb{R}^n$ is defined as $\texttt{conv}(v_1, \dots, v_N) = \big\{x \in \mathbb{R}^n : x = \sum_{i=1}^N e_i v_i, e_i \geq 0, \sum_{i=1}^N e_i = 1\big\}$.

A *convex polytope* in $\mathbb{R}^n$ is denoted by $\mathcal{P} = \texttt{conv}(v_1, \dots, v_N)$ where each v_i represents a vertex of the polytope. In the following, polytopes are considered as convex by default. A polytope is a bounded (convex) polyhedron $\mathcal{P} \subset \mathbb{R}^n$, which can be also defined as $\mathcal{P} = \{\mathbf{x} \in \mathbb{R}^n | H\mathbf{x} \leq \mathbf{k}\}$ where H is a matrix of size $m \times n$ and $\mathbf{k}$ is a column vector of dimension m. This latter can be rewritten as a set of constraints $\mathcal{P} = \{x \in \mathbb{R}^n : \langle l_i, x\rangle + a_i \geq 0, i = 1, \dots, m\}$ where $\langle ., . \rangle$ is scalar product in $\mathbb{R}^n$, $l_i \in \mathbb{R}^n$ and $a_i \in \mathbb{R}$. $\mathcal{P}$ has exactly m facets which are the intersections of $\mathcal{P}$ with the hyperplanes defined in the previous definition. These facets (or similarly the vertices v_i) can be computed with a geometrical approach while the set defined by $\mathcal{P}$ can be paved using a classical branch and prune algorithm.

A *zonotope* is a centrally symmetric polytope. The main advantage of zonotopes is that its set computation (operations, such as addition or product, on variables with zonotopic domains) can be implemented using affine arithmetic [13]. A set of values in this domain is represented by an *affine form* $\hat{x}$, which is a formal expression of the form $\hat{x} = \alpha_0 + \sum_{i=1}^n \alpha_i \varepsilon_i$ where the coefficients α_i are real numbers called *noise symbols*, α_0 the *center* of the affine form, and the ε_i are formal variables ranging over the interval $[-1, 1]$. Affine forms encode linear dependencies among variables: if $x \in [a_1, a_2]$ and $y = 2x$, then x will be represented by the affine form $\hat{x}$ above and y will be represented as $\hat{y} = 2\alpha_0 + 2\alpha_1 \varepsilon$.

2.2 Polytopes as intersection of zonotopes

A polytope can be represented exactly by the intersection of some zonotopes as proposed in [4, 24]. Once a polytope $\mathcal{P}$ has been represented exactly by the intersection of zonotopes, i.e., $\mathcal{P} = Z_1 \cap \dots \cap Z_n$, the image of a function f on $\mathcal{P}$ can be computed as $f(\mathcal{P}) = f(Z_1 \cap \dots \cap Z_n) \subseteq f(Z_1) \cap \dots \cap f(Z_n)$ where $f(Z_1), \cdots, f(Z_n)$ can be computed using affine arithmetic.

A general procedure to exactly represent a polytope $\mathcal{P} \subset \mathbb{R}^n$ by the intersection of zonotopes is presented in [24]: randomly select n inequality constraints from the pool of all inequality constraints representing the polytope and then use these n inequality constraints to construct a zonotope with the minimal volume containing the polytope until all inequality constraints for the polytope have been used up.

Notice that an intersection of zonotopes is not a zonotope in general. Such an intersection can be computed, or enclosed, with the help of a box based paver or a polytope based method, but the computation cost is usually expensive and prohibitive [15].

2.3 Set-based simulation of ODEs

An *Initial Value Problem* (IVP) for ODEs is defined by[2]

$$\begin{cases} \dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y}(t)), \\ t \in \mathcal{T} := [0, t_{\text{end}}] \text{ and } \mathbf{y}(0) \in \mathcal{Y}_0, \end{cases} \tag{1}$$

with a nonlinear function $\mathbf{f} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$. More precisely, IVP for ODEs are considered over a finite time horizon $[0, t_{\text{end}}]$. Note that a bounded set $\mathcal{Y}_0$ of initial values is considered in this framework. This implies that solution of Equation (1) is a set of trajectories. We assume classical hypotheses on $\mathbf{f}$ to ensure the existence and uniqueness of the solution of Equation (1). The set $\mathcal{Y}(\mathcal{T}, \mathcal{Y}_0)$ stands for $\mathcal{Y}(\mathcal{T}, \mathcal{Y}_0) = \{\mathbf{y}(t; \mathbf{y}_0) : t \in \mathcal{T}, \mathbf{y}_0 \in \mathcal{Y}_0\}$.[3] Intuitively, $\mathcal{Y}(\mathcal{T}, \mathcal{Y}_0)$ gathers all the points reached by the scalar solution $\mathbf{y}(t; \mathbf{y}_0)$ of Equation (1) starting from all scalar initial values $\mathbf{y}_0$. Note that $\mathcal{Y}(\mathcal{T}, \mathcal{Y}_0)$ is hardly computable in general. The goal of a validated or rigorous numerical integration methods is to characterize the set of functions satisfying (1). These functions are represented by the set of values they can reach with their associated time instants, *i.e.*, $\{\mathbf{y}(t; \mathbf{y}_0) : \mathbf{y}_0 \in \mathcal{Y}_0, t \in [0, t_{\text{end}}]\}$. A convenient and often used way to access these values is to use the abstract domain of intervals which uses interval analysis to compute an over approximation of this set [10, 19, 20]. In the following, we consider Lipschitz and continuous ODEs.

Different set abstractions can be considered to tackle the problem of simulation for IVP (boxes, zonotopes, ellipsoids, etc). The main challenge is to be able to compute arithmetic operations with the chosen abstraction, even for nonlinear operations. Box representation is often preferred because of its very simple arithmetic which is also available in many tools. When considering the set of initial conditions as a box $[\mathbf{y}_0] \supset \mathcal{Y}_0$, the use of the interval framework to solve Equation (1) makes possible the design of an inclusion function $[\mathbf{y}](t; [\mathbf{y}_0])$ for the computation of an over approximation of $\mathbf{y}(t; [\mathbf{y}_0])$. To build it, a sequence of time instants $t_1, \dots, t_s$ such that $t_1 < \dots < t_s$ and a sequence of boxes $[\mathbf{y}_1], \dots, [\mathbf{y}_s]$ such that $[\mathbf{y}](t_{i+1}; [\mathbf{y}_i]) \subseteq [\mathbf{y}_{i+1}]$ for all $i \in [0, s-1]$ are computed using a classic iterative

[2] Notice that the $\mathbf{f}$ function generally takes a $\mathbf{q} \in \mathbb{R}^p$ argument representing parameters. We omit it in the current paper as we do not consider parameterized differential equations.

[3] $\mathbf{y}(t; \mathbf{y}_0)$ is a notation representing the function $\mathbf{y}(t)$ with fixed initial values $\mathbf{y}_0$.

two-steps methods [20] producing a *reachable tube* $\{[\tilde{\mathbf{y}}_0], \ldots, [\tilde{\mathbf{y}}_s]\}$ where each $[\tilde{\mathbf{y}}_i]$ is the state interval containing all the values reachable in the time interval $[t_i, t_{i+1}]$.

Zonotopes are another abstraction used in set based simulation. Indeed, zonotopes being the geometrical concretization of affine arithmetic, simulation computed with affine arithmetic allows to obtain a reachable tube made of zonotopes $\mathcal{T} \equiv \{([t_1], \tilde{\mathcal{Z}}_1), \ldots, ([t_{\text{end}}], \tilde{\mathcal{Z}}_{\text{end}})\}$ which is a time-sorted list of pairs (time interval × zonotope). Tools such as DynIbex [1] or CORA [5] implement this feature. The main advantages of zonotopes are the wrapping effect[4] reduction and a more precise set representation. Considering a closed set $\mathcal{S}$, its optimal box-hull $\mathcal{B}$, its optimal zonotope-hull $\mathcal{Z}$ and its optimal polytope-hull $\mathcal{P}$, the following property holds: $\mathcal{S} \subset \mathcal{P} \subset \mathcal{Z} \subset \mathcal{B}$.

2.4 From reachable tubes to constraint programming

The solution of an IVP-ODE which is given as a set of timed zonotopes in the form $\{([t_1], \tilde{\mathcal{Z}}_1), \ldots, ([t_{\text{end}}], \tilde{\mathcal{Z}}_{\text{end}})\}$ can easily be transcribed as a disjunction of constraints since each pair $([t_i], \tilde{\mathcal{Z}}_i)$ corresponds to a quantified proposition $\forall t \in [t_i]\ \mathbf{y}(t) \in \tilde{\mathcal{Z}}_i$.[5] Eventually, the abstraction of the set of trajectories can be considered as a disjunction of all these constraints since at each time $t \in [t_0, t_{\text{end}}]$, $\mathbf{y}(t)$ verifies exactly one of them[6].

3 CP for polytopic integration

Computing the intersection of zonotopic tubes is a hard task: zonotopes are not closed under intersection or union, and thus computation of the intersection of tubes of zonotopes can be very expensive using a geometrical approach. We propose in this section to tackle this problem from the constraint solving point of view, as the intersection of zonotopic tubes resulting from ODEs can be seen as a Constraint Satisfaction Problem (CSP). Because enumerating all solutions of CSPs is generally impossible when the domains of the variables are continuous, CSP solvers generally compute a set of abstract elements that *covers* the solution space.

We have proposed in [27] a *tree abstract domain* $\mathbb{T}(D)$. This domain can be viewed as a k-d tree [8] in which leaves are defined using the numerical abstract domain D used (e.g. zonotopes) and internal nodes, also called summaries, give information about their subtrees. The tree abstraction exploits the continuous aspect of ODEs solutions to propose a fast pre-computation of the intersection of zonotopes (see Figure 1). It can thus be seen as an incremental powerset that gradually increases its precision (*i.e.* decreasing over-estimation), starting from the

[4] Wrapping effect is the overestimation induced by set abstractions in some iterative computations.

[5] $t \in [t_i]$ means $\underline{[t_i]} \leqslant t \leqslant \overline{[t_i]}$ with $\underline{[t_i]}$ and $\overline{[t_i]}$ the lower and upper bounds of $[t_i]$ respectively.

[6] As two consecutive time intervals $[t_i, t_{i+1}]$ and $[t_{i+1}, t_{i+2}]$ shared one bound (a floating point value t_{i+1}), $\mathbf{y}(t_{i+1})$ is verified for two constraints. However, as $\mathbf{y}(t_{i+1})$ is unique, this degenerated case is not an issue for the following.

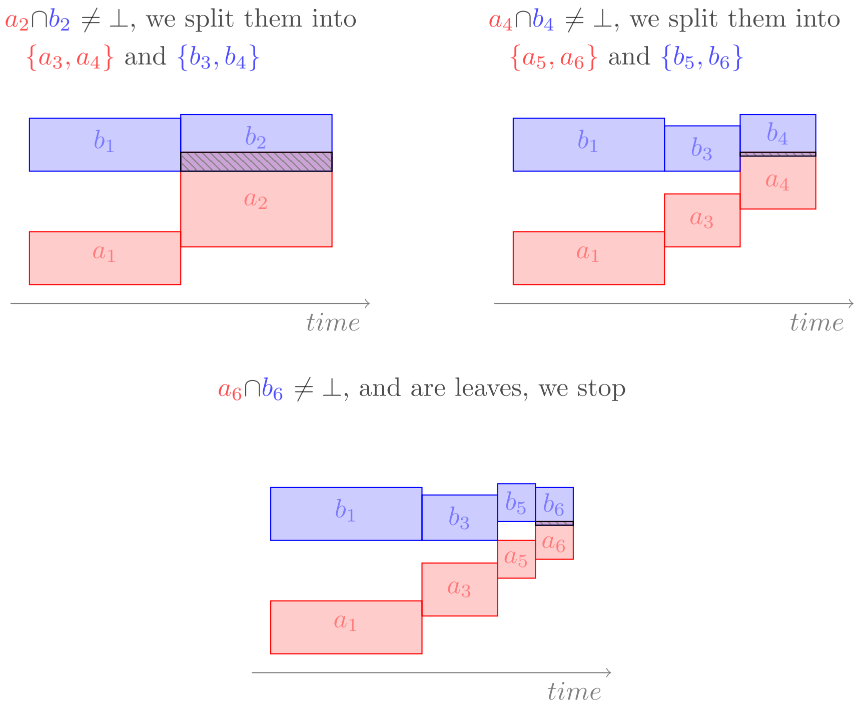

$a_2 \cap b_2 \neq \perp$, we split them into
$\{a_3, a_4\}$ and $\{b_3, b_4\}$
b_1
b_2
a_2
a_1
time
$a_4 \cap b_4 \neq \perp$, we split them into
$\{a_5, a_6\}$ and $\{b_5, b_6\}$
b_1
b_3
b_4
a_4
a_3
a_1
time
$a_6 \cap b_6 \neq \perp$, and are leaves, we stop
b_1
b_3
b_5
b_6
a_6
a_5
a_3
a_1
time

Figure 1: Recursive intersection computation (boxes for illustration).

precision of a base abstract domain D to the precision of its powerset $P(D)$ only if needed. This greatly speeds up the solving process in our experiments.

A tube $\mathcal{T}$ can be defined using a disjunction of predicates as described in Subsection 2.4. $\mathcal{T} = ([t_1] \wedge e_1) \vee ([t_2] \wedge e_2) \cdots \vee ([t_n] \wedge e_n)$ where each e_i represents the set of values of solution functions within time frame $[t_i]$, can be read as: *the solution is either in set e_1 during the time frame $[t_1]$, or in set e_2 during the time frame $[t_2]$, etc.* This property is always true, as exactly one of its atoms will be true at a given time (cf. Footnote 6 on Page 759) and defines the reachable tube.

Considering initial values given as a polytope $\mathcal{P}$, $\mathcal{P}$ is decomposed as an intersection of s zonotopes $\mathcal{Z}_i$ as described in Subsection 2.2. The reachable tube of the ODE is therefore described by the conjunction of s tubes $\mathcal{T}^1 \wedge \cdots \wedge \mathcal{T}^i \wedge \cdots \wedge \mathcal{T}^s$, each tube $\mathcal{T}^i$ being obtained by the zonotopic simulation of the ODE with initial value taken as a zonotope $\mathcal{Z}_i$ (cf. Section 2). As the initial polytope is not empty, the s tubes obtained with validated simulation have a non empty intersection due to the Lipschitz and continuous properties of ODEs. This intersection is a correct enclosure of the theoretical tube computed with initial value as polytope $\mathcal{P}$.

Note that there is no synchronization between the tubes, *i.e.* the time frames t_*^i of tube $\mathcal{T}^i$ may be different than the time frames t_*^j of tube $\mathcal{T}^j$, but each time frame $[t_*^i]$ of $\mathcal{T}^i$ intersects with at least one time frame $[t_*^j]$ of $\mathcal{T}^j$.

Figure 2 shows that the intersection of two zonotope-based tubes $\mathcal{T}^1$ (in blue) and $\mathcal{T}^2$ (in red) produces the polytope-based tube shown in green. Using our proposed tree abstraction we are able to efficiently compute the polytope-based tube from the two zonotope-based tubes without having to explicitly define the constraints corresponding to such an intersection.

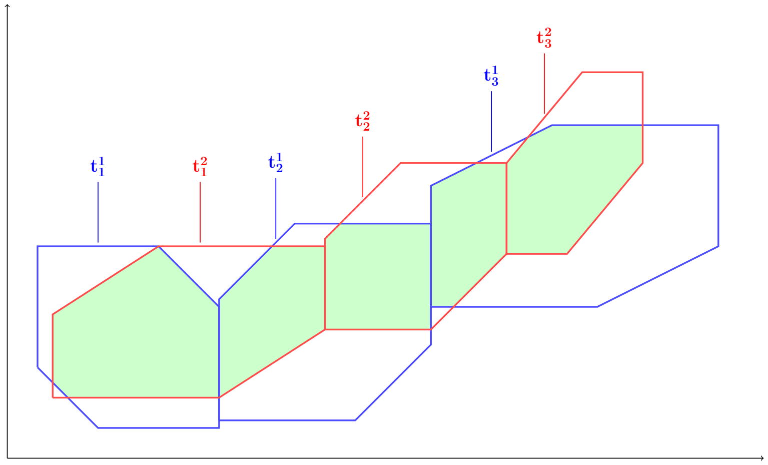


Figure 2: Polytopes as intersections of zonotopes with their own time frames.

Such a CSP C, declared as the intersection of the CSPs describing $\mathcal{T}^1$ and $\mathcal{T}^2$, can be used to:

- pave the reachable tube (with a branch and prune algorithm for example). Notice that such paving may be computationally demanding.

- detect collision with obstacles or other tubes representations by adding constraints. Let us suppose that that we want to verify that the previous tube does not intersect a given obstacle, defined by a polytope $\mathcal{P}_o$. We build the CSP $C \wedge (t \in [t_0, t_{end}] \wedge \mathcal{P}_o)$ and ask our solver to solve the CSP: if there is no solution, then the tube does not intersect the obstacle, otherwise the solver should give us a collision example.

4 Experiments

The following experiments have been conducted using the open-source library DynIbex [1] to compute the zonotope-based reachable tubes and the open-source constraint solver AbSolute [21] equipped with the tree domain presented in Section 3 to check if constraints over the computed tubes hold (no intersection for instance). AbSolute is also used to pave intersections of zonotopes to visualize the corresponding polytope.

4.1 First experiments

We first choose the two classic problems presented in [4] in order to validate our approach. In a third experiment we also compare our polytope based technique with a pointwise Monte-Carlo approach. We analyze these three experiments in Paragraph 4.1.4.

4.1.1 The circle problem

The circle problem is defined by the following equation:

$$\left\{ \begin{array}{lcl} \dot{y}_1 & = & -y_2 \\ \dot{y}_2 & = & y_1 \end{array} \right. \tag{2}$$

The initial condition is taken in a polytope given by the five vertices $(-1,-3)$, $(-1.5,3)$, $(0,6)$, $(1.5,4)$, and $(1,-4)$ at $t_0 = 0$. This polytope can be defined as the intersection of three zonotopes. The simulation with these three zonotopes as initial conditions is performed with Kutta's third order validated method (also called RK32 in [11]), with absolute error tolerance $10^{-6}$and till $t = 20$ seconds.

Figures 3a shows the last zonotopes of the three reachable tubes. The polytope defined as the intersection of these three zonotopes is guaranteed to contain the reachable state at the end of simulation horizon ($t = 20$). To compute this polytope, AbSolute is used with a reduced domain for time: we consider $t \in [19.99, 20.00]$. Figure 3b presents the results using the box abstract domain to pave the solution while Figure 3c presents the results using the polytope abstract domain as base domain to pave the solution. The full reachable tube can also be computed with AbSolute (with time domain $t \in [0, 20]$) and is depicted in Figure 4a and Figure 4b.

4.1.2 The Lotka-Volterra problem

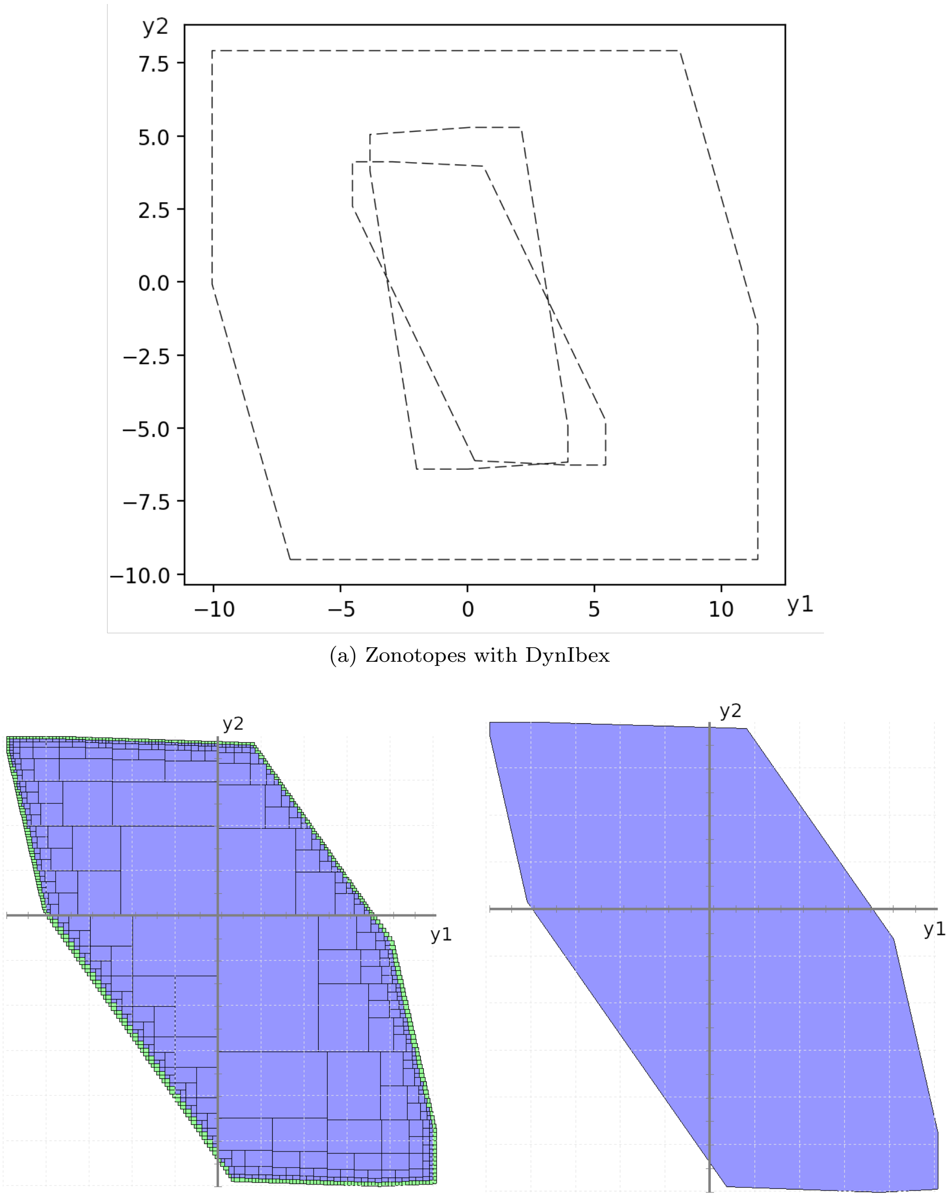
The Lotka-Volterra problem is defined by the following equation:

$$\left\{ \begin{array}{lcl} \dot{y}_1 & = & 2y_1(1-y_2) \\ \dot{y}_2 & = & -y_2(1-y_1) \end{array} \right. \tag{3}$$

The initial condition is taken in a polytope given by the eight vertices $(1.1035, 3.0457)$, $(1.1041, 3.0386)$, $(1.0981, 3.0366)$, $(1.1039, 3.0358)$, $(1.0983, 3.0339)$, $(1.1020, 3.0320)$, $(1.0989, 3.0498)$ and $(1.0995, 3.0510)$. This polytope is covered by three zonotopes. Simulations with these three zonotopes as initial conditions are performed with Kutta's third order validated method with absolute error tolerance 10^{-6} and till $t = 6$ seconds. The reachable tube is computed as the intersection of the three zonotope-based reachable tubes. The result is shown on Figure 5a and Figure 5b and corresponds to the classic solution of the problem.

4.1.3 Comparison on the Van der Pol problem

The last problem aims to compare the results obtained with our polytope based technique and a pointwise integration obtained with a Monte-Carlo approach (154



(a) Zonotopes with DynIbex

(b) Polytopes with AbSolute and box domain (c) Polytopes with AbSolute and polytope domain

Figure 3: Last solution of the tube for Example 4.1.1.

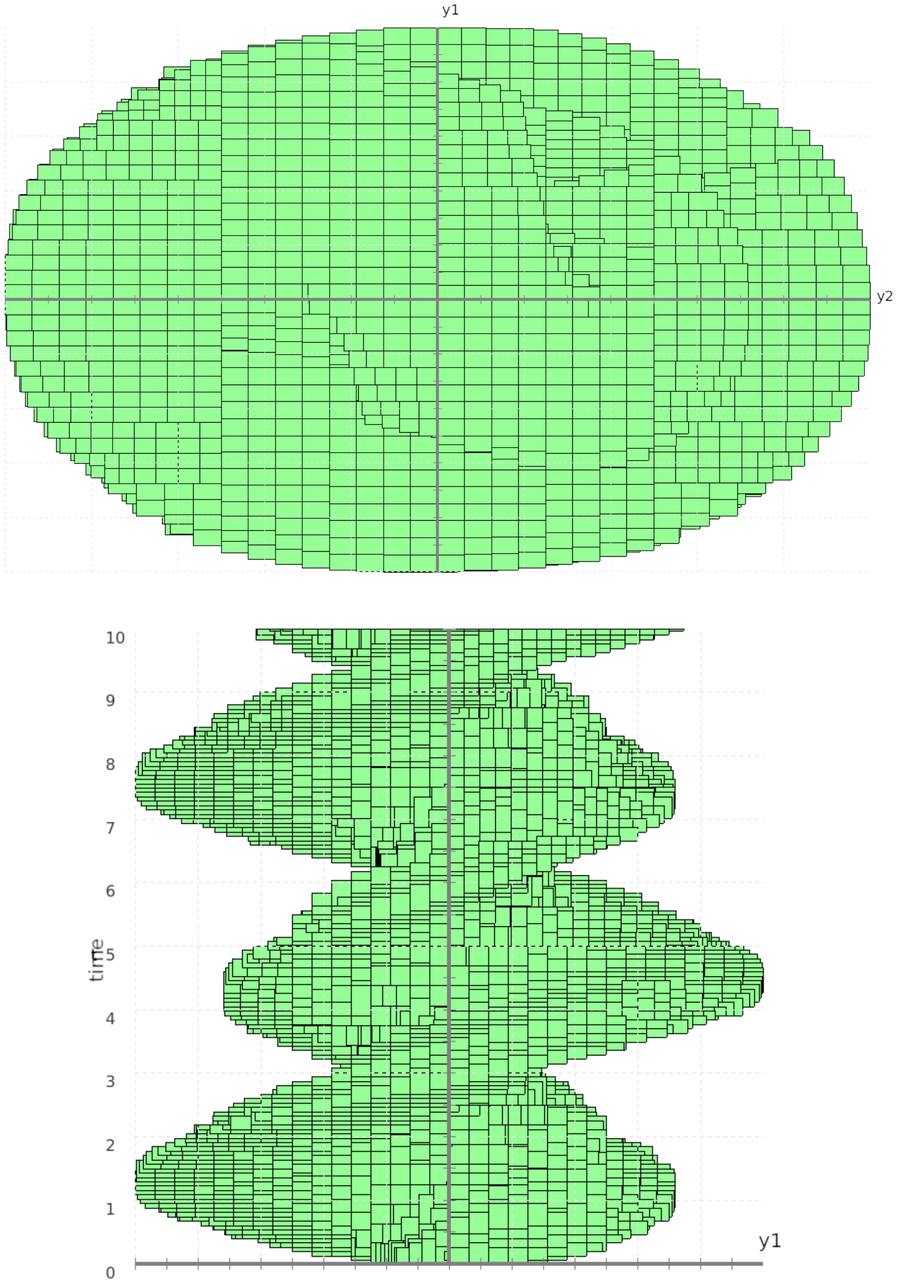
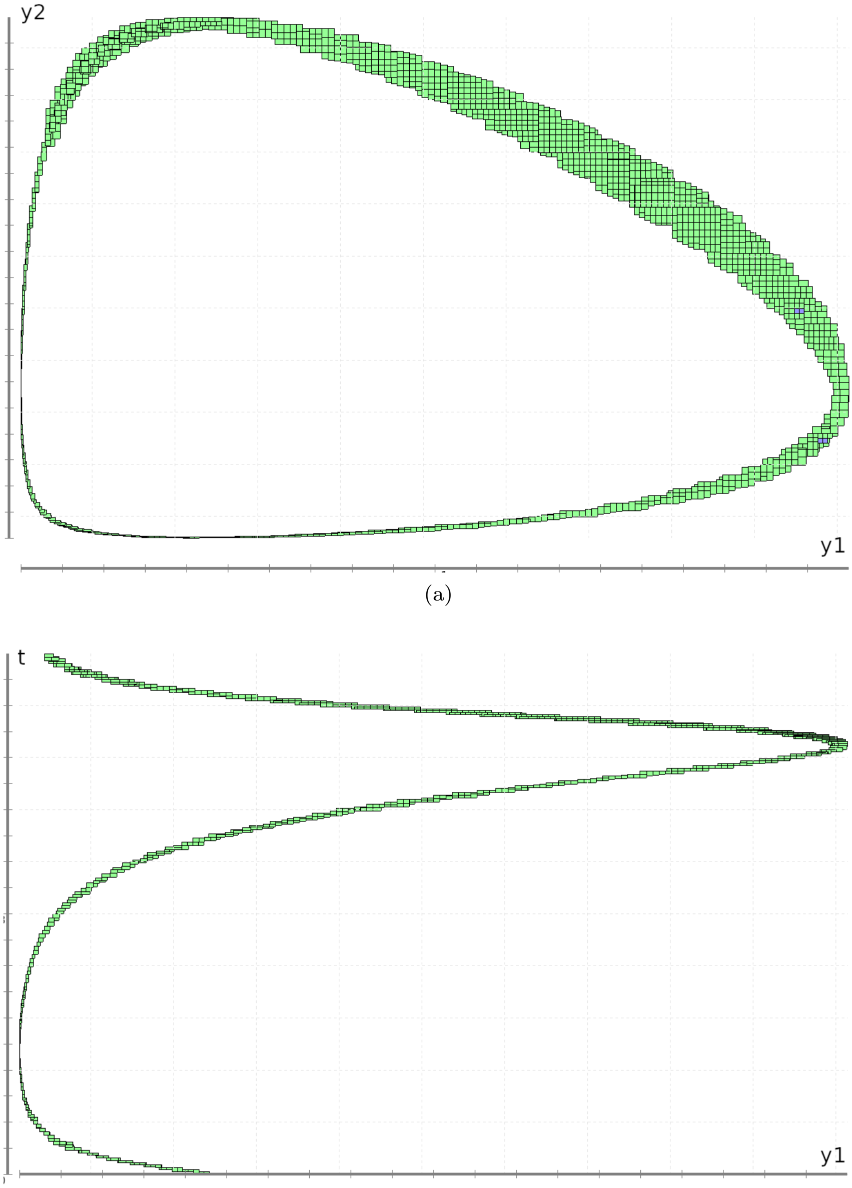


Figure 4: Reachable tube obtained with box domain for Example 4.1.1.



(a)

(b)

Figure 5: Reachable tube obtained with box domain for Example 4.1.2.

points from a regular mesh of the initial polytope, depicted in blue in the following figures). The chosen problem is the Van der Pol oscillator, known to be interesting to challenge tools, as it has a limit cycle which attracts pointwise trajectories but can stress set-based simulators because of system rotation. It is defined by:

$$\left\{ \begin{array}{rcl} \dot{y}_1 & = & y_2 \\ \dot{y}_2 & = & y_2(1 - y_1^2) - y_1 \end{array} \right. . \tag{4}$$

Initial conditions are defined as the intersection of three zonotopes and are plotted in Figure 6 and the whole tube computed with our technique is shown in Figure 7.

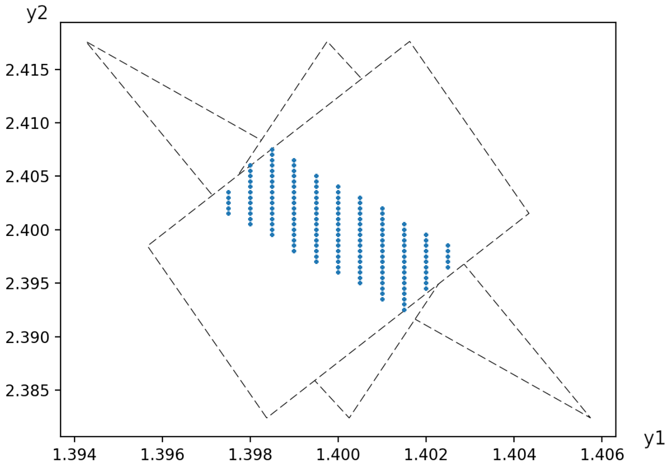


Figure 6: Initial condition, given as the intersection of three zonotopes, for polytope based simulation and initial points for pointwise simulation.

Figures 8a, 8b and 8c present respectively at time $t = 1$, $t = 3$ and $t = 7$ the polytope computed by AbSolute and points obtained with pointwise initial conditions and validated simulation by DynIbex. All simulations (zonotopes-based and pointwise) are performed with Runge-Kutta four order validated method with absolute error tolerance 10^{-8} and till $t = 7$ seconds (time for a revolution).

4.1.4 Appraisal

The circle problem presented in Subsection 4.1.1 shows the results of the polytope computation at some instants and on a whole tube. At a chosen instant, the obtained polytope is similar to the one obtained in [4]. The full tube shows a good stability as the circle is well depicted with a restrained overestimation (property of Hamiltonian conservation is preserved). To verify this conservation, we have

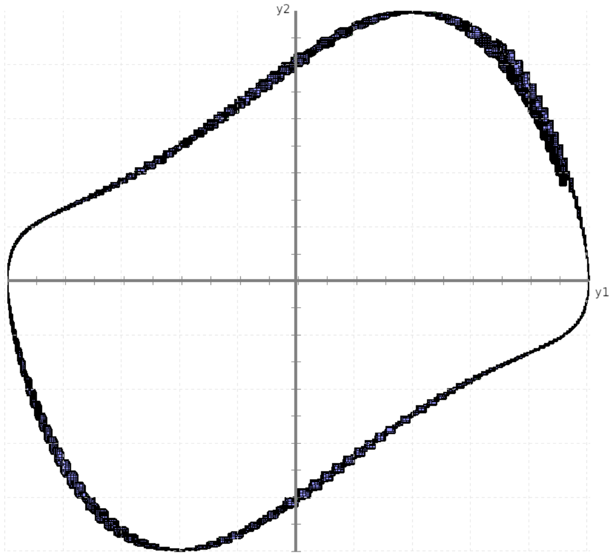


Figure 7: Van der Pol oscillator limit cycle computed with initial condition as a polytope.

computed the maximal distance to the frame center such as max $\sqrt{y_1(t)^2 + y_2(t)^2}$ at different instants ($t = 0, 5, 10, 15, 20$) and we obtained $6.0, 7.58, 6.40, 8.16, 7.63$. The values oscillate as selected times do not follow the system period, but do not show any divergence.

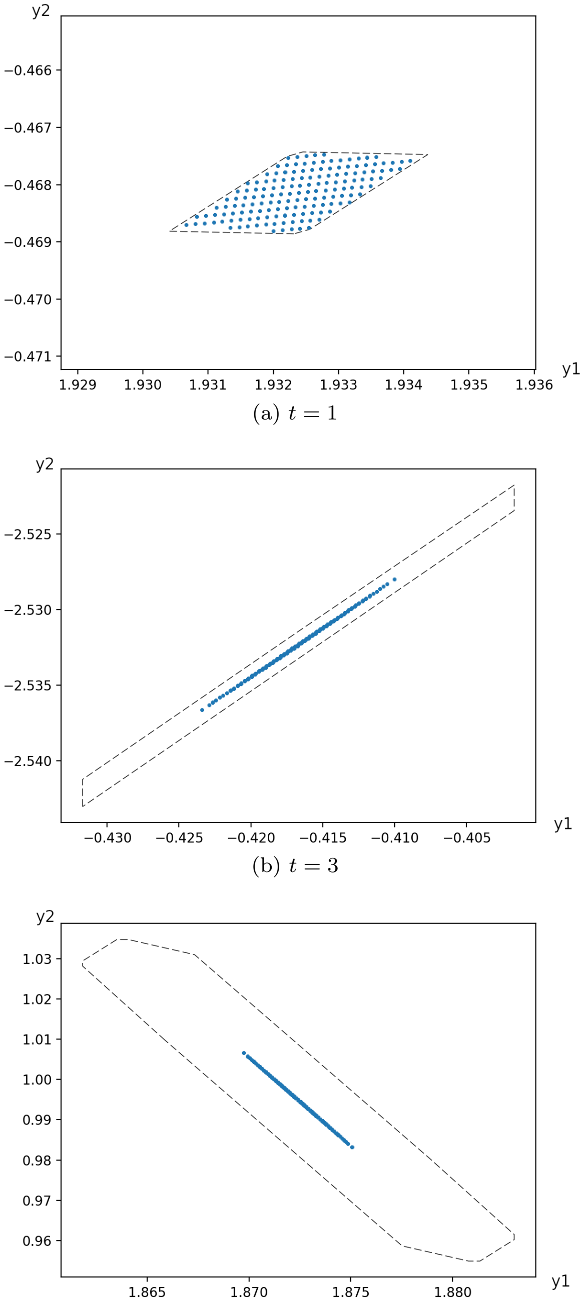
The Lotka-Volterra problem confirms that the contracting instant (bottom left on Figure 5a) is preserved, and that the dissipative instant (top right) has restrained effect on the size of polytopes. The Van der Pol oscillator problem shows that wrapping effect has small effect and that polytopes are able to preserve, at least better than boxes, a limit cycle.

4.2 Application on collision detection and rendez-vous

4.2.1 Collision detection

In our previous work [2], we have applied the CP approach with interval domains to check a collision-free property on a state of the art distributed multi-agent formation control protocol [26], briefly recalled in the following.

Consider n agents in $\mathbb{R}^d$ ($n \geqslant 2$ and $d \geqslant 2$). Their position at time t is denoted by $p_i(t) \in \mathbb{R}^d$ with $i \in \{1, \ldots, n\}$. Interactions between agents are described by a graph $\mathcal{G}$ with a vertex set $\mathcal{V} = \{1, \ldots, n\}$ and an edge set $\mathcal{E}$. An edge $(i, j) \in \mathcal{E}$ means that agent i can measure the relative bearing of agent j so is a neighbor of Agent j. The set of all neighbors of agent i is denoted by $\mathcal{N}_i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$. Edges are not directed: if i is a neighbor of j, then j is also a neighbor of i. A



(a) $t = 1$

(b) $t = 3$

(c) $t = 7$

Figure 8: Comparison between the polytope obtained with AbSolute and Monte-Carlo approach.

formation denoted by $\mathcal{G}(p)$ is a graph $\mathcal{G}$ such that each vertex i of $\mathcal{G}$ is associated to $p_i(t)$. The relative bearing of p_j with respect to p_i is defined by $e_{ij} := p_j - p_i$, $g_{i,j} := \frac{e_{ij}}{\|e_{ij}\|}$, with $\| \,.\, \|$ being the Euclidean norm.

Suppose the velocity of n_ℓ agents is given. Such agents are called *leaders* and the other n_f agents are called *followers*. The dynamics of leaders and followers follows a single integrator model that can be used to follow a predefined plan made of different way-points. When dealing with a formation of n agents, we have to define n trajectories $r_1, \dots r_n$ and the constraint corresponding to the possible collision between agents is expressed by $(r_1 \wedge r_2) \vee \dots (r_1 \wedge r_n) \vee (r_2 \wedge r_3) \vee \dots \vee (r_{n-1} \wedge r_n)$: there is a collision if two constraints representing two trajectories are true at the same time and therefore the global formula is true.

We perform a new verification with our polytopic approach. As in [2], we only consider a set of four agents using displacements respecting a square-based formation: translation, scaling and rotation. Composition of these three displacements is performed to generate trajectories for the agents following algorithm presented in [26]. A finite number of values for each displacement is considered: north, south, east, west direction for translation; contraction and dilation for scaling; rotations with angles $\pi/4$, $\pi/2$, $3\pi/4$ and π over the centroid. 108 atomic displacements are thus considered. Two different sources of uncertainties are taken into account: uncertainties on initial positions of agents and on inter-agent distances, allowing four different scenarios.We aim to detect which combination of displacements may generate a collision.

A summary of the verification of collision-free property on atomic displacements is given in Table 1: `NO` stands for no uncertainty, `EI` stands for uncertainty on initial conditions, `ED` stands for uncertainty on distance measure and `EID` stands for uncertainties on both initial position and distance measures. Two values are considered for uncertainties, namely 0.01 and 0.1. The number of satisfiable, unsatisfiable and

Table 1: Collision-free checking on atomic displacements for robot formation.

	T1T2	T1T3	T1T4	T2T3	T2T4	T3T4
`NO`	27/81/0 0/107/1	1/107/0 0/108/0	4/104/0 0/107/1	3/105/0 0/106/2	3/105/0 0/108/0	4/104/0 0/106/2
`EI` 0.01	27/81/0 0/107/1	1/107/0 0/108/0	9/99/0 1/105/2	9/99/0 0/108/0	3/105/0 0/108/0	17/91/0 0/105/3
`EI` 0.1	27/81/0 0/105/3	12/96/0 0/108/3	41/67/0 1/101/6	52/56/0 0/108/0	26/82/0 0/108/0	54/54/0 0/97/11
`ED` 0.01	27/81/0 0/108/0	3/105/0 0/108/0	16/92/0 0/107/1	19/89/0 0/108/0	11/97/0 0/108/0	52/56/0 0/107/1
`ED` 0.1	27/81/0 0/107/1	31/77/0 0/108/0	80/28/0 0/107/1	48/60/0 0/108/0	50/58/0 0/108/0	65/43/0 0/106/2
`EID` 0.01	27/81/0 0/105/3	4/104/0 0/108/0	22/86/0 1/105/2	28/80/0 0/108/0	11/97/0 0/108/0	54/54/0 0/107/1
`EID` 0.1	27/81/0 0/106/2	38/70/0 0/108/0	92/16/0 1/106/1	70/38/0 0/108/0	56/52/0 0/108/0	69/39/0 0/107/1

inconclusive problems (read SAT/UNSAT/MAYBE) is reported for each scenario and each pair of trajectory between agents (T1T2 stands for trajectory of agent 1 and agent 2). An UNSAT scenario means that no collision have been found and so the atomic displacement is safe while a SAT scenario implies that a potential collision has been found. No conclusion can be done with a MAYBE result.

Looking at this experimental evaluation of the bearing-based formation control, we note that increasing uncertainties increases the number of possible collisions for interval domain but the polytopic approach seems more robust with respect to uncertainties. Moreover, polytopic approach provides sharper trajectory tubes and so reduce the number of possible collisions.

4.2.2 Location of a rendez-vous area

As a second application, we consider the problem of the location of a meeting point or a rendez-vous area. In a mathematical point of view, it is equivalent to the detection of a global attractor, which is a difficult problem. This application could be used to simulate a crowd behavior [25] or study risk of contamination during an epidemic (as where people meet is an important question in epidemiology).

We build a domain with a limit cycle (from Van Der Pol oscillator), a repulsor at position $(1, 1)$ and an attractor at position $(-1, -1)$. The resulting ODE is:

$$\left\{\begin{array}{lcl} \dot{y}_1 & = & \alpha y_2 - \beta(y_1 + 1) + \gamma(y_1 - 1) \\ \dot{y}_2 & = & \alpha(y_2(1 - y_1^2) - y_1) - \beta(y_2 + 1) + \gamma(y_2 - 1) \end{array}\right. \qquad (5)$$

with $\alpha = 1$, $\beta = 0.8$ and $\gamma = 0.3$. We randomly drop 10 particles in the corresponding vector field and we detect if a rendez-vous occurs (on a 30 seconds horizon), *i.e.*, if two particles are at the same place at the same time. This is the opposite of the previous problem of collision detection, but the technique is the same.

A zonotopic tube is computed for each particle and two CSP are solved: 1) a disjunction of tubes to depict the trajectories, and 2) a conjunction of tubes to locate a possible rendez-vous area. Results are shown in Figure 9.

An inner region is tagged as a rendez-vous for all the particles in the interval $[-1.69944786755,\ -1.69888555045]$; $[0.252656244732, 0.253503277614]$ during instant $[11.152624717, 11.1533191391]$, while, between $t = 10.35$ and $t = 30$, a meeting is possible between at least two particles, without certitude, around $(-1.70; 0.25)$ (inside the penumbra). This place seems to be an attractor.

5 Discussion and conclusion

In this paper, we presented a method based on Constraint Programming to compute reachable tubes of Ordinary Differential Equations in the particular case where the initial domain is given as a polytope. This polytopic approach is of interest in the field of cyber-physical system verification, because polytopes offer in general a sharper enclosure than classical boxes. Moreover, initial domains are often defined in engineering processes with constraints producing in general a polytope. The

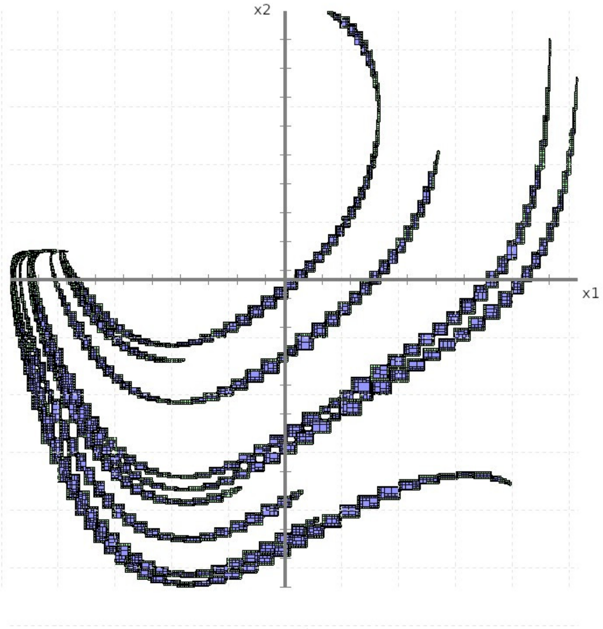


Figure 9: The result for the disjunction of the trajectories of the ten particles.

proposed approach is therefore related to engineering requirements. Our method has been experimented using two open-source tools known to be efficient in their respective communities: DynIbex and AbSolute. The results obtained are good and promising for many applications such as control synthesis (for example with respect to some safety constraints even with uncertainties in sensors), motion planning (for example with respect to uncertainties and obstacle avoidance), etc.

Concerning computation time, the circle problem is solved on a basic laptop with few seconds for the three zonotopic simulations, between few seconds and a minute for the CSP generation (depending on required precision) and few seconds for the CSP solving, which is globally reasonable. CSP generation may be included to the simulation step to optimize execution time.

As future work, even if the method does not present any limitation in term of problems that can be handled, we plan to test and compare it on more complex and higher dimensional problems. After this first step, applications such as properties verification for cyber-physical systems, invariant computation for ODEs or hybrid system simulation will be considered.

References

[1] Alexandre dit Sandretto, J. and Chapoutot, A. Validated explicit and implicit Runge–Kutta methods. *Reliable Computing*, 22(1):79–103, 2016. URL: https:

//hal.science/hal-01243053.

[2] Alexandre dit Sandretto, J., Chapoutot, A., Garion, C., Thirioux, X., and Ziat, G. Constraint-based verification of formation control. In *Proceedings of the 60th IEEE Conference on Decision and Control*, pages 7136–7141. IEEE, 2021. DOI: 10.1109/CDC45484.2021.9683622.

[3] Alexandre dit Sandretto, J., Chapoutot, A., and Mullier, O. Constraint-based framework for reasoning with differential equations. In Koç, Ç. K., editor, *Cyber-Physical Systems Security*, pages 23–41. Springer International Publishing, 2018. DOI: 10.1007/978-3-319-98935-8_2.

[4] Alexandre dit Sandretto, J. and Wan, J. Reachability analysis of nonlinear ODEs using polytopic based validated Runge-Kutta. In *Proceedings of Reachability Problems*, Volume 11123 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2018. DOI: 10.1007/978-3-030-00250-3_1.

[5] Althoff, M. An introduction to CORA 2015. In *Proceedings of the Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 120–151, 2015. DOI: 10.29007/zbkv.

[6] Althoff, M., Frehse, G., and Girard, A. Set propagation techniques for reachability analysis. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(1), 2021. DOI: 10.1146/annurev-control-071420-081941.

[7] Althoff, M. and Krogh, B. H. Zonotope bundles for the efficient computation of reachable sets. In *Proceedings of the IEEE Conference on Decision and Control and European Control Conference*, pages 6814–6821, 2011. DOI: 10.1109/CDC.2011.6160872.

[8] Bentley, J. L. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975. DOI: 10.1145/361002.361007.

[9] Bouissou, O., Goubault, E., Putot, S., Tekkal, K., and Vedrine, F. Hybrid-Fluctuat: A static analyzer of numerical programs within a continuous environment. In *Proceedings of Computer Aided Verification*, Volume 5643 of *Lecture Notes in computer Science*, pages 620–626. Springer, 2009. DOI: 10.1007/978-3-642-02658-4_46.

[10] Bouissou, O. and Martel, M. Abstract interpretation of the physical inputs of embedded programs. In *Proceedings of Verification, Model Checking, and Abstract Interpretation*, Volume 4905 of *Lecture Notes in Computer Science*, pages 37–51. Springer, 2008. DOI: 10.5555/1787526.1787534.

[11] Butcher, J. C. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2008. DOI: 10.1002/9781119121534.

[12] Cruz, J. and Barahona, P. Constraint reasoning over differential equations. *Applied Numerical Analysis and Computational Mathematics*, 1(1):140–154, 2004. DOI: 10.1002/anac.200310012.

[13] de Figueiredo, L. H. and Stolfi, J. *Self-Validated Numerical Methods and Applications*. Brazilian Math. Colloquium monographs. IMPA/CNPq, 1997. URL: https://www.iri.upc.edu/people/thomas/Collection/details/45477.html.

[14] Eggers, A., Ramdani, N., Nedialkov, N., and Fränzle, M. Improving SAT modulo ODE for hybrid systems analysis by combining different enclosure methods. In *Proceedings of Software Engineering and Formal Methods*, Volume 7041 of *Lecture Notes in Computer Science*, pages 172–187. Springer, 2011. DOI: 10.1007/s10270-012-0295-3.

[15] Ghorbal, K., Goubault, E., and Putot, S. A logical product approach to zonotope intersection. In Touili, T., Cook, B., and Jackson, P. B., editors, *Proceedings of the 22th International Conference on Computer Aided Verification*, Volume 6174 of *Lecture Notes in Computer Science*, pages 212–226. Springer, 2010. DOI: 10.1007/978-3-642-14295-6_22.

[16] Goldsztejn, A., Mullier, O., Eveillard, D., and Hosobe, H. Including ordinary differential equations based constraints in the standard CP framework. In *Proceedings of Principles and Practice of Constraint Programming*, Volume 6308 of *Lecture Notes in Computer Science*, pages 221–235. Springer, 2010. DOI: 10.5555/1886008.1886031.

[17] Henzinger, T. A., Horowitz, B., Majumdar, R., and Wong-Toi, H. Beyond HYTECH: Hybrid systems analysis using interval numerical methods. In *Proceedings of Hybrid Systems: Computation and Control*, Volume 1790 of *Lecture Notes in Computer Science*, pages 130–144. Springer, 2000. DOI: 10.1007/3-540-46430-1_14.

[18] Janssen, M., Deville, Y., and Van Hentenryck, P. Multistep filtering operators for ordinary differential equations. In *Proceedings of Principles and Practice of Constraint Programming*, Volume 1713 of *Lecture Notes in Computer Science*, pages 246–260. Springer, 1999. DOI: 10.1007/978-3-540-48085-3_18.

[19] Moore, R. E. *Interval Analysis*. Prentice Hall, 1966. ISBN: 2577-9435.

[20] Nedialkov, N. S., Jackson, K. R., and Corliss, G. F. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105(1):21–68, 1999. DOI: 10.1016/S0096-3003(98)10083-8.

[21] Pelleau, M., Miné, A., Truchet, C., and Benhamou, F. A constraint solver based on abstract domains. In *Proceedings of Verification, Model Checking, and Abstract Interpretation*, Volume 7737 of *Lecture Notes in Computer Science*, pages 434–454. Springer, 2013. DOI: 10.1007/978-3-642-35873-9_26.

[22] Rohou, S., Bedouhene, A., Chabert, G., Goldsztejn, A., Jaulin, L., Neveu, B., Reyes, V., and Trombettoni, G. Towards a generic interval solver for differential-algebraic CSP. In *Proceedings of Principles and Practice of Constraint Programming*, Volume 12333 of *Lecture Notes in Computer Science*, pages 548–565. Springer, 2020. DOI: 10.1007/978-3-030-58475-7_32.

[23] Tucker, W. A rigorous ODE solver and Smale's 14th problem. *Foundations of Computational Mathematics*, 2(1):53–117, 2002. DOI: 10.1007/s002080010018.

[24] Wan, J., Sharma, S. K., and Sutton, R. Guaranteed state estimation for nonlinear discrete-time systems via indirectly implemented polytopic set computation. *IEEE Transactions on Automatic Control*, 63(12):4317–4322, 2018. DOI: 10.1109/TAC.2018.2816262.

[25] Wong, K. Y., Thyvetil, M.-A., Machaira, A., and Loscos, C. System for simulating dynamic features of crowd behavior. In *ACM SIGGRAPH 2005 Posters*, SIGGRAPH '05. ACM, 2005. DOI: 10.1145/1186954.1187085.

[26] Zhao, S. and Zelazo, D. Bearing-based formation stabilization with directed interaction topologies. In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 6115–6120, 2015. DOI: 10.1109/CDC.2015.7403181.

[27] Ziat, G., Mullier, O., Alexandre dit Sandretto, J., Garion, C., Chapoutot, A., and Thirioux, X. Abstract domains for constraint programming with differential equations. In *Proceedings of the 9th International Workshop on Numerical and Symbolic Abstract Domains*, pages 2–11, 2020. DOI: 10.1145/3427762.3429453.