On the Decidability of the Reachability Problem for Planar Differential Inclusions *

E. Asarin **, G. Schneider ***, and S. Yovine

VERIMAG
2, Ave de Vignate
38610 - Gières, France
{Eugene.Asarin, Gerardo.Schneider, Sergio.Yovine}@imag.fr

Abstract. In this paper we develop an algorithm for solving the reachability problem of two-dimensional piece-wise rectangular differential inclusions. Our procedure is not based on the computation of the reach-set but rather on the computation of the limit of individual trajectories. A key idea is the use of one-dimensional affine Poincar maps for which we can easily compute the fixpoints. As a first step, we show that between any two points linked by an arbitrary trajectory there always exists a trajectory without self-crossings. Thus, solving the reachability problem requires considering only those. We prove that, indeed, there are only finitely many "qualitative types" of those trajectories. The last step consists in giving a decision procedure for each of them. These procedures are essentially based on the analysis of the limits of extreme trajectories. We illustrate our algorithm on a simple model of a swimmer spinning around a whirlpool.

1 Introduction

One of the main research areas in hybrid systems is reachability analysis. It comprises two (closely related) issues, namely, the study of decidability and the development of algorithms. Most of the proved decidability results are based on the existence of a finite and computable partition of the state space into classes of states which are equivalent with respect to reachability. This is the case for timed automata [2], and classes of rectangular automata [12] and hybrid automata with linear vector fields [15]. Except for timed automata, these results rely on stringent hypothesis such as the resetting of variables along transitions.

Although analysis techniques based on the construction of a finite partition have been proposed [7], mainly all implemented computational procedures resort to (forward or backward) propagation of constraints, typically (unions of convex) polyhedra or ellipsoids [1,3,6,9,11,14]. In general, these techniques provide semi-decision procedures, that is, if the given final set of states is reachable, they will

* This work was partially supported by Projet IMAG MASH "Modélisation et Analyse de Systèmes Hybrides".
** Partially supported by the NATO under grant CRG-961115.
*** Supported by ESPRIT-LTR Project 26270 VHS "Verification of Hybrid Systems".

terminate, otherwise they may fail to. This is a property of the techniques, not of the problem, that is, it does not imply that the reachability problem itself is undecidable, but only that they do not implement a decision procedure for it. In other words, these algorithms may be unsuccessful (i.e., not terminate) for certain classes of systems for which the reachability problem is indeed decidable (by other means). Nevertheless, they provide tools for computing (approximations of) the reach-set for large classes of hybrid systems with linear and non-linear vector fields.

Maybe the major drawback of set-propagation, reach-set approximation procedures is that they pay little attention to the geometric properties of the specific (class of) systems under analysis. To our knowledge, in the context of hybrid systems there are two lines of work in the direction of developing more "geometric" approaches. One is based on the existence of (enough) integrals and the ability to compute them all [7,10]. These methods, however, do not necessarily result in decision procedures (they are actually not meant to). The other, applicable to two-dimensional dynamical systems, relies on the topological properties of the plane, and explicitly focuses on decidability issues. This approach has been proposed in [16]. There, it is shown that the reachability problem for two-dimensional systems with piece-wise constant derivatives (PCD) is decidable. This result has been extended in [8] for planar piece-wise Hamiltonian systems. In [4] it has been shown that the reachability problem for PCD is undecidable for dimensions higher than two.

In this paper we develop an algorithm for solving the reachability problem of two-dimensional piece-wise rectangular differential inclusions. As in [16], our procedure is not based on the computation of the reach-set but rather on the computation of the limit of individual trajectories. A key idea is the use of one-dimensional affine Poincar maps for which we can easily compute the fixpoints. The decidability result of [16] fundamentally relies on the determinism of PCD which implies that planar trajectories do not intersect themselves. This property is no longer true for differential inclusions. As a first step, we show that between any two points linked by an arbitrary trajectory there always exists a trajectory without self-crossings. Thus, solving the reachability problem requires considering only those. We prove that, indeed, there are only finitely many "qualitative types" of those trajectories. The last step consists in giving a decision procedure for each of them. These procedures are essentially based on the analysis of the limits of extreme trajectories (which do not cut themselves).

2 Simple Planar Differential Inclusions

A *simple planar differential inclusion system* (SPDI) consists of a partition of the plane into convex polygonal regions, together with a differential inclusion associated with each region. As an example consider the problem of a swimmer trying to escape from a whirlpool in a river.

Example. The dynamics $\dot{\mathbf{x}}$ of the swimmer around the whirlpool is approximated by the piece-wise differential inclusion defined as follows. The zone of the river

nearby the whirlpool is divided into 8 regions $R_1, \ldots, R_8$. To each region R_i we associate a pair of vectors $(\mathbf{a}_i, \mathbf{b}_i)$ meaning that $\dot{\mathbf{x}}$ belongs to their positive hull: $\mathbf{a}_1 = \mathbf{b}_1 = (1, 5)$, $\mathbf{a}_2 = \mathbf{b}_2 = (-1, \frac{1}{2})$, $\mathbf{a}_3 = (-1, \frac{11}{60})$ and $\mathbf{b}_3 = (-1, -\frac{1}{4})$, $\mathbf{a}_4 = \mathbf{b}_4 = (-1, -1)$, $\mathbf{a}_5 = \mathbf{b}_5 = (0, -1)$, $\mathbf{a}_6 = \mathbf{b}_6 = (1, -1)$, $\mathbf{a}_7 = \mathbf{b}_7 = (1, 0)$, $\mathbf{a}_8 = \mathbf{b}_8 = (1, 1)$. The corresponding SPDI is illustrated in Fig. 1. □

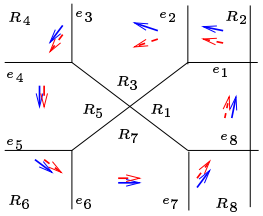

Fig. 1. The SPDI of the swimmer.

More formally, a SPDI is a pair $\mathcal{H} = (\mathcal{P}, \phi)$, where $\mathcal{P}$ is a finite partition of the plane into convex polyhedral sets, and for each $P \in \mathcal{P}$, $\phi(P)$, also denoted by $\angle_{\mathbf{a}_P}^{\mathbf{b}_P}$, is the set of all linear combinations $\mathbf{x} = \alpha\ \mathbf{a}_P + \beta\ \mathbf{b}_P$, with $\alpha, \beta \geq 0$, and $\alpha + \beta > 0$, of two vectors $\mathbf{a}_P$ and $\mathbf{b}_P$, such that $\hat{\mathbf{a}}_P \cdot \mathbf{b}_P < 0$, where $\cdot$ is the scalar product and $\hat{\mathbf{a}}_P = (a_2, -a_1)$ is the clockwise rotation of $\mathbf{a}_P$ by the angle $\frac{\pi}{2}$ (notice that $\hat{\mathbf{a}}_P \cdot \mathbf{a}_P = 0$).

Let $E(P)$ be the set of edges of P, that is, the set of open segments forming the boundary of P, and $V(P)$ be the set of vertices in the boundary of P. We say that e is an *entry* of P if for all $\mathbf{x} \in e$ and for all $\mathbf{c} \in \phi(P)$, $\mathbf{x} + \mathbf{c}\epsilon \in P$ for some $\epsilon > 0$. We say that e is an *exit* of P if the same condition holds for some $\epsilon < 0$. We denote by $in(P) \subseteq E(P)$ the set of all entries of P and by $out(P) \subseteq E(P)$ the set of all exits of P. In general, $E(P) \neq in(P) \cup out(P)$. We say that P is a *good* region iff all the edges in $E(P)$ are entries or exits, that is, $E(P) = in(P) \cup out(P)$. Notice that, if P is a good region, then for all $e \in E(P)$, the director vector of e does not belongs to $\phi(P)$ (Fig. 2). Hereinafter, we assume that all regions are good regions. Let $\mathbf{x} \in V(P)$ be a common vertex of two edges e and e'. $\mathbf{x}$ is an *entry point* to P if both e and e' are entry edges; it is an exit point if both e and e' are exit edges. In fact, vertices can be seen as a particular kind of edges, with exactly one point. In what follows the term *edge* will be understood as belonging to the set $EV(P) = E(P) \cup V(P)$. If needed, the difference between edge and vertex will be explicitly specified.

A *trajectory* in some interval $[0, T] \subseteq \mathbb{R}$, with initial condition $\mathbf{x} = \mathbf{x}_0$, is a continuous and almost-everywhere (everywhere except on finitely many points) derivable function $\xi(\cdot)$ such that $\xi(0) = \mathbf{x}_0$ and for all $t \in (0, T)$, if $\xi(t) \in P \setminus EV(P)$, then $\dot{\xi}(t)$ is defined and $\dot{\xi}(t) \in \phi(P)$.

The point-to-point reachability problem for $\mathcal{H}$, is the following: Given $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^2$, is there a trajectory ξ and $t \geq 0$ such that $\xi(0) = \mathbf{x}$ and $\xi(t) = \mathbf{x}'$?. If the

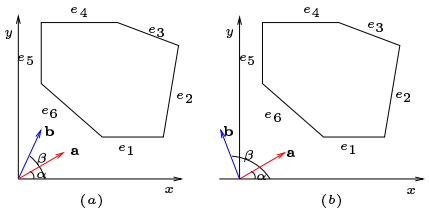

Fig. 2. a) A good region. b) A bad region ($e_5 \notin in(P) \cup out(P)$).

answer is yes, we say that $\mathbf{x}'$ is *reachable* from $\mathbf{x}$. The edge-to-edge reachability problem is the following: Given two edges e and e' of $\mathcal{H}$, is there $\mathbf{x} \in e$ and $\mathbf{x}' \in e'$ such that $\mathbf{x}'$ is reachable from $\mathbf{x}$? The region-to-region reachability problem is defined similarly.

3 Properties of Trajectories

W.l.o.g. we will consider in what follows that $\xi(0) \in e$ for some edge e. The *trace* of a trajectory ξ is the sequence $\tau(\xi) = \mathbf{x}_0\mathbf{x}_1 \ldots$ of the intersection points of ξ with the set of edges, that is, $\mathbf{x}_i \in \xi \cap \bigcup EV(P)$ for all $P \in \mathcal{P}$. The *edge signature* of ξ is the sequence $\sigma(\xi) = e_0e_1 \ldots$ of traversed edges, that is, $\mathbf{x}_i \in e_i$. The *region signature* of ξ is the sequence $\rho(\xi) = P_0P_1 \ldots$ of traversed regions, that is, $e_i \in in(P_i)$.

Let ξ be a trajectory whose trace is $\tau(\xi) = \mathbf{x}_0 \ldots \mathbf{x}_k$. Let $0 = t_0 < t_1 < \ldots < t_k$ be such that $\xi(t_i) = \mathbf{x}_i$. Since ξ is continuous and derivable in the interval (t_i, t_{i+1}), there exists a unique trajectory ξ' with $\xi'(t_i) = \xi(t_i)$ for all $i \in [0, k-1]$, such that the derivative $\dot{\xi}'$ is constant in the interval (t_i, t_{i+1}). That is,

Proposition 1. *For every trajectory ξ there exists a trajectory ξ' with the same initial and final points, and edge and region signatures, such that for each P_i in the region signature, there exists $\mathbf{c}_i \in \phi(P_i)$, such that $\dot{\xi}'(t) = \mathbf{c}_i$ for all $t \in (t_i, t_{i+1})$.*

Hence, in order to solve the reachability problem it is enough to consider trajectories having piecewise constant slopes. Notice that, however, such slopes need not be the same for each occurrence of the same region in the region signature. Hereinafter, we use the word "trajectory" to mean trajectories whose derivatives are piecewise constant.

Consider a region P and let $\mathbf{c} \in \phi(P)$. The mapping $\Omega : \mathbb{R}^2 \to \mathbb{R}$, defined as $\Omega(\mathbf{x}) = \mathbf{x} \cdot \hat{\mathbf{c}}$, assigns to every $\mathbf{x} \in \mathbb{R}^2$ a value proportional to the length of the projection of the vector $\mathbf{x}$ on the right rotation of $\mathbf{c}$ (see [4]). Indeed, the ordering is given by the direction of $\hat{\mathbf{c}}$ and one can easily see that the relation $\preceq$, defined as $\mathbf{x}_1 \preceq \mathbf{x}_2$ if $\Omega(\mathbf{x}_1) \leq \Omega(\mathbf{x}_2)$, is a dense linear order on $in(P)$ and $out(P)$ (Fig. 3). We use $\prec$ to denote the strict variant of $\preceq$ and say that $e_1 \prec e_2$

iff $e_1 \neq e_2$ and $\mathbf{x}_1 \preceq \mathbf{x}_2$ for every $\mathbf{x}_1 \in e_1, \mathbf{x}_2 \in e_2$. For example, in Fig. 3 we have $e_1 \prec e_2 \prec e_3$. Notice that the order does not depend on the choice of $\mathbf{c}$.

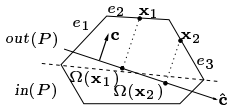

Fig. 3. Ordering: $\mathbf{x}_1 \preceq \mathbf{x}_2$.

We say that a trajectory ξ crosses itself if there exist $t \neq t'$ such that $\xi(t) = \xi(t')$. If a trajectory does not cross itself, the sequence of consecutive intersection points with $in(P)$ or $out(P)$ is monotone with respect to $\preceq$. That is, for every three points $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ (visited in this order), if $\mathbf{x}_1 \prec \mathbf{x}_2 \prec \mathbf{x}_3$ the trajectory is a "counterclockwise expanding spiral"(Fig. 4(a)) or a "clockwise contracting spiral" (Fig. 4(b)) and if $\mathbf{x}_3 \prec \mathbf{x}_2 \prec \mathbf{x}_1$, the trajectory is a "counterclockwise contracting spiral" (Fig. 4(c)) or a "clockwise expanding spiral" (Fig. 4(d)). On the other hand, if the sequence of intersections points with $in(P)$ or $out(P)$ is monotone (both increasing or both decreasing), the trajectory does not cross itself.

Lemma 1. *For every trajectory ξ, if ξ does not cross itself, then for every edge e, the sequence $\tau(\xi) \cap e$ is monotone (with respect to $\prec$).*

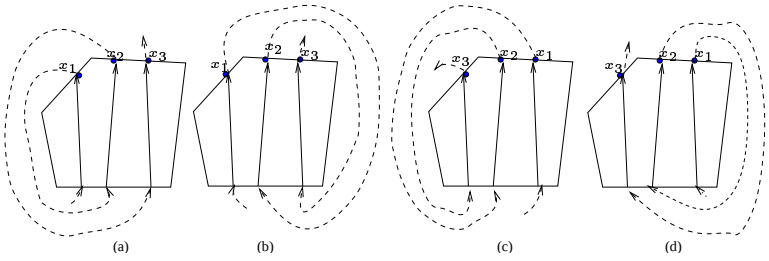

Fig. 4. Spirals.

Now suppose that the trajectory ξ with trace $\tau(\xi) = \mathbf{x}_0 \dots \mathbf{x}_f$ crosses itself *once* inside the region P. Let $e_1, e_2 \in in(P)$ be the input edges and $e'_1, e'_2 \in out(P)$ be the output ones. Let $\mathbf{x} = \mathbf{x}_i \in e_1$ and $\mathbf{y} = \mathbf{x}_j \in e_2$, with $i < j$, be the points in $\tau(\xi)$ the first and the second times ξ enters P, and let $\mathbf{x}' = \mathbf{x}_{i+1} \in e'_2$ and $\mathbf{y}' = \mathbf{x}_{j+1} \in e'_1$ be the corresponding output points. Let $\mathbf{c}_x, \mathbf{c}_y \in \phi(P) = \angle^{\mathbf{b}}_{\mathbf{a}}$ be the derivatives of ξ in the time intervals (t_i, t_{i+1}) and (t_j, t_{j+1}), respectively. Indeed,

$\mathbf{c}_x$ and $\mathbf{c}_y$ are the director vectors of the segments $\overline{\mathbf{xx}'}$ and $\overline{\mathbf{yy}'}$, respectively (Fig. 5(a)).

Consider now the segment $\overline{\mathbf{xy}'}$. Notice that the director vector $\mathbf{c}'_x$ of this segment can be obtained as a positive combination of the vectors $\mathbf{c}_x$ and $\mathbf{c}_y$. Thus, $\mathbf{c}'_x \in \phi(P)$. Hence, there exists a trajectory ξ' that does not cross itself in P having a trace $\tau(\xi') = \mathbf{x}_0 \ldots \mathbf{xy}' \ldots \mathbf{x}_f$ (Fig. 5(b)). Notice that the result also works for the *degenerate* case when the trajectory crosses itself at an edge (or vertex).

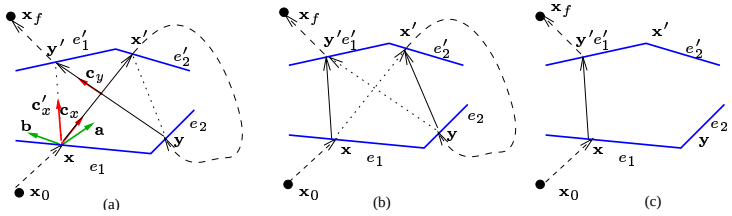

Fig. 5. Obtaining a non-crossing trajectory

If the trajectory ξ crosses itself more than once in region P, then the number of times the trajectory ξ', obtained by cutting away the loop (Fig. 5(c)), crosses itself in P is strictly smaller than the number of times ξ does it (see Fig. 6). After replacing $\overline{\mathbf{xx}'}$ and $\overline{\mathbf{yy}'}$ by $\overline{\mathbf{xy}'}$, the intersection q of $\overline{\mathbf{xx}'}$ and $\overline{\mathbf{yy}'}$ disappears. If the new segment of line $\overline{\mathbf{xy}'}$ crosses another segment $\overline{\mathbf{zz}'}$ (say at a point t), then $\overline{\mathbf{zz}'}$ necessarily crosses either $\overline{\mathbf{xx}'}$ (at r) or $\overline{\mathbf{yy}'}$ (at s) -or both-, before the transformation. The above is due to the fact that if $\overline{\mathbf{zz}'}$ crosses one side of the triangle $\mathbf{xy}'q$ then it must also cross one of the other sides of the triangle, say at r. Thus, no new crossing can appear and the number of crossings in the new configuration is always less than in the old one.

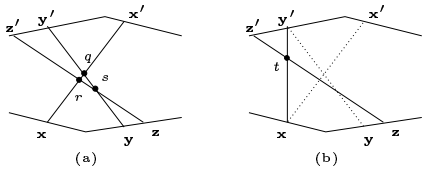

Fig. 6. The number of crossings decreases: (a) Before (3 crossings); (b) After (1 crossing).

Lemma 2. *For every trajectory ξ that crosses itself at least once, there exists a trajectory ξ' with the same initial and final points of ξ having a number of self-crossings strictly smaller.*

The above result follows from a straightforward inductive reasoning, as well as the following one.

Proposition 2. *If there exists an arbitrary trajectory from points $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$ then there always exists a non-crossing trajectory between them.*

Hence, in order to solve the reachability problem we only need to consider non-crossing trajectories with piecewise constant derivatives. In what follows, we only deal with trajectories of this kind.

4 Properties of Edge Signatures

Let ξ be a trajectory with trace $\tau(\xi) = \mathbf{x}_0 \dots \mathbf{x}_p$, edge signature $\sigma(\xi) = e_0 \dots e_p$, and region signature $\rho(\xi) = P_0 \dots P_p$. An edge e is said to be *abandoned* by ξ after position i, if $e_i = e$ and for some j, k, $i \leq j < k$, $P_j \dots P_k$ forms a region cycle and $e \notin \{e_{i+1}, \dots, e_k\}$. Since trajectories are *finite* we should add the trivial case when $e \neq e_j$ for all $j > i$.

Lemma 3 (Claim 2 in [4]). *For every trajectory ξ and edge e, if e is abandoned by ξ after position i, e will not appear in $\sigma(\xi)$ at any position $j > i$.*

Given a sequence s, we use notations $first(s)$ and $last(s)$ for the first and last elements of the sequence respectively. ε denotes the empty sequence An edge signature $\sigma(\xi)$ can be canonically expressed as a sequence of edges and cycles of the form $\sigma_c(\xi) = r_1 s_1^{k_1} r_2 s_2^{k_2} \dots r_n s_n^{k_n} r_{n+1}$, where

1. For all $i \in [1, n+1]$, r_i is a sequence of pairwise different edges;
2. For all $i \in [1, n]$, s_i is a simple cycle (i.e., without repetition of edges) repeated k_i times;
3. For all $i \in [1, n-1]$, $first(r_{i+1}) \neq first(s_i)$ if $r_{i+1} \neq \varepsilon$, otherwise $first(s_{i+1}) \neq first(s_i)$;
4. For all $i \in [1, n]$, if $r_i \neq \varepsilon$ then $last(r_i) = last(s_i)$;
5. $r_{n+1} \neq \varepsilon$. Moreover, $r_{n+1} = first(s_n)$ if $\sigma(\xi)$ ends in a loop and $first(r_{n+1}) \neq first(s_n)$ otherwise.

This canonical representation can be obtained as follows. Let $\sigma(\xi) = e_1 \dots e_{p-1} e_p$ be an edge signature. Starting from e_{p-1} and traversing backwards, take the first edge that occurs the second time. If there is no such edge, then trivially the signature can be expressed in a canonical way and we are done. Otherwise, suppose that the edge e_j occurs again at position i (i.e. $e_i = e_j$ with $i < j$), thus $\sigma_c(\xi) = wsr$, where w, s and r are obtained as follows, depending on the repeated edge:

$$w = e_0 \dots e_i$$
$$s = e_{i+1} \dots e_j$$
$$r = e_{j+1} \dots e_{p-1}$$

Clearly r is not a cycle and s is a simple cycle with no repeated edges. We continue the analysis with w. Let $k_m = max\{l \mid s^l \text{ is a suffix of } w\}$. Thus, $\sigma_c(\xi) = w's^k r$ with $w' = e_0 \dots e_h$ (a prefix of w) and $k = k_m + 1$. We repeat recursively the procedure above with w'. Adding the edge e_p to the last r (at the end) we obtain $\sigma_c(\xi) = r_1 s_1^{k_1} \dots r_n s_n^{k_n} r_{n+1}$ that is a canonical representation of signature σ.

Let us define the *type of a signature* σ as $type(\sigma(\xi)) = r_1, s_1, \dots, r_n, s_n, r_{n+1}$. Notice that the "preprocessing" (taking away the last edge e_p) is done in order to differentiate edges signatures that end with a cycle from those that do not. There exists many other (maybe easier) ways of decomposing a signature σ in a canonical form (in particular, traversing forward instead of backwards), but the one chosen here permits a clearer and simpler presentation of the reachability algorithm. In fact in this canonical form, the last visited edge in a cycle $e_1 \dots e_k$ is always the last one (e_k).

Example. Let us consider the following examples. Suppose that $\sigma = abcdbcefg\ efgefgefhi$. Then, after applying once the above procedure we obtain that $\sigma_c = w(s_2)^3 r_1$, with $w = abcdbcef$; $s_2 = gef$; $r_1 = h$. Applying the procedure once more to w we obtain $w = w'(s_3)^1 r_2$ with $w' = r_3 = abc$; $s_3 = dbc$; $r_2 = ef$. Putting all together and adding the last edge (i) gives $\sigma_c = abc(dbc)^1 ef(gef)^3 hi$ with type $type(\sigma) = abc, dbc, ef, gef, hi$. Suppose now, that the signature ends with a cycle: $\sigma = abcdbcefgefgefgefgef$. In this case we apply the preprocessing obtaining $\sigma_c = w(s_2)^4 r_1$ with $w = abcdbce$; $s_2 = fge$; $r_1 = \varepsilon$. Applying the procedure to w we finally obtain $w = w'(s_3)^1 r_2$ with $w' = r_3 = abc$; $s_3 = dbc$; $r_2 = e$ and that gives $\sigma_c = abc(dbc)^1 e(fge)^4 f$ (adding f to the end). □

Lemma 4. *The type of a signature σ, $type(\sigma)$, has the following properties:*

1. *For every $1 \le i \ne j \le n+1$, r_i and r_j are disjoint;*
2. *For every $1 \le i \ne j \le n$, s_i and s_j are different;*
3. *If v is a vertex appearing in $type(\sigma)$, then it can only occur exactly once in r_i for some $1 \le i \le n+1$ in σ. Moreover, $v \notin last(r_i)$ unless $i = n+1$.*

Proposition 3. *The set of types of edge signatures is finite.*

Thus, to solve the reachability problem we can proceed by examining one by one the types of signatures.

5 Affine Operators

Before getting into the problem of analyzing types of edge signatures, we need to introduce some useful notions.

An *affine function* $f : \mathbb{R} \to \mathbb{R}$ is defined by a formula $f(x) = ax + b$ with $a > 0$. An *affine multivalued operator* $F : \mathbb{R} \to 2^{\mathbb{R}}$ is determined by two affine functions $f_l(x)$ and $f_u(x)$ and maps x to the interval $\langle f_l(x), f_u(x)\rangle$, where $\langle a, b\rangle$ means (a, b), $[a, b]$, $(a, b]$ or $[a, b)$: $F(x) = \langle f_l(x), f_u(x)\rangle$. We use the notation $F = \langle f_l, f_u\rangle$. Such an operator can be naturally extended to subsets of

$\mathbb{R}$: $F(S) = \bigcup_{x \in S} F(x)$. In particular, if $S = \langle l, u\rangle$: $F(\langle l, u\rangle) = \langle f_l(l), f_u(u)\rangle$. A *truncated affine multi-valued operator* $G : \mathbb{R} \to 2^{\mathbb{R}}$ is determined by an affine multi-valued operator F and an interval $\langle L, U\rangle$ as follows: $G(x) = F(x) \cap \langle L, U\rangle$. Such operators can be also extended to sets. We use notations $G = F \cap \langle L, U\rangle$ and $F = \widetilde{G}$.

Lemma 5 (composition of affine operations). *Affine functions, affine multi-valued operators, and truncated affine multi-valued operators are closed under composition.*

Example. Let $\tilde{G}_1(x) = (2x+3,\, 3x+5]$ and $\tilde{G}_2(x) = [5x+2,\, 7x+6]$ be two (non-truncated) affine multi-valued functions, $G_1 = \tilde{G}_1 \cap (1,\, 6]$, and $G_2 = \tilde{G}_2 \cap [6,\, 10)$ their truncated versions. The truncated affine multi-valued operator $G_2 \circ G_1(x)$ is obtained as follows:

$$\begin{aligned} G_2 \circ G_1(x) &= \tilde{G}_2 \circ \tilde{G}_1(x) \cap \tilde{G}_2((1,\, 6]) \cap [6,\, 10) \\ &= (5(2x+3)+2,\, 7(3x+5)+6] \cap (5 \cdot 1 + 2,\, 7 \cdot 6 + 6] \cap [6,\, 10) \\ &= (10x+17,\, 21x+41] \cap (7,\, 48] \cap [6,\, 10) \\ &= (10x+17,\, 21x+41] \cap (7,\, 10). \end{aligned}$$

Notice also that for any interval $\langle l, u\rangle$ its image is $G_2 \circ G_1(\langle l, u\rangle) = \langle 10l + 17,\, 21u + 41\rangle \cap (7,\, 10)$. □

Let f be an affine function, x_0 be any initial point and $x_n = f^n(x)$. Clearly, the sequence x_n is monotonous, and it converges to a limit x^* (finite or infinite). Indeed, x^* can be effectively computed knowing a, b and x_0, as follows. If $a < 1$, x^* is the unique fixpoint of f, that is, $ax^* + b = x^*$, which yields $x^* = b/(1-a)$. If $a = 1$, $x^* = -\infty$ if $b < 0$, $x^* = \infty$ if $b > 0$, and $x^* = x_0$, if $b = 0$. If $a > 1$, let $x_* = b/(1-a)$, then $x^* = -\infty$ if $x_0 < x_*$, $x^* = \infty$ if $x_0 > x_*$, $x^* = x_0 = x_*$, otherwise. This result can be extended to multi-valued affine functions.

Lemma 6. *Let $\langle l_0, u_0\rangle$ be any initial interval and $\langle l_n, u_n\rangle = \tilde{F}^n(\langle l_0, u_0\rangle)$. Then*

1. *The sequences l_n and u_n are monotonous;*
2. *They converge to limits l^* and u^* (finite or infinite), which can be effectively computed.*

Proposition 4. *Let F be truncated affine and $I \subseteq \langle L, U\rangle$. Then $F^n(I) = \tilde{F}^n(I) \bigcap \langle L, U\rangle$.*

6 Computing the Successor Function

To solve the reachability problem for SPDI, the next step is to provide a procedure for computing the successors of a point (and an interval), which requires having an effective representation of (rational) points and intervals on edges.

Let us first introduce a one-dimensional coordinate system on each edge. For each edge e we chose (1) a point on it (the origin) with radius-vector $\mathbf{v}$, and

(2) a director vector $\mathbf{e}$ going in the positive direction in the sense of the order $\prec$. Now to characterize e we need the coordinates of its extreme points, namely, $e^l, e^u \in \mathbb{Q} \cup \{-\infty, \infty\}$ such that $e = \{\mathbf{x} = \mathbf{v} + x\mathbf{e} \mid e^l < x < e^u\}$. That is, an edge $e \in E$ can be represented by a triplet $(\mathbf{v}, \mathbf{e}, (e^l, e^u))$. If the edge is a vertex, the representation is simply $(\mathbf{v}, [0, 0])$. Now, every point $\mathbf{x} = \mathbf{v} + x\mathbf{e} \in e$ is represented by the pair (e, x) (Fig.7(a)), and every interval $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle \subseteq e$ is represented as $(e, \langle x_1, x_2 \rangle)$, where $\mathbf{x}_1 = (e, x_1)$ and $\mathbf{x}_2 = (e, x_2)$ (Fig.7(b)). Now, having fixed

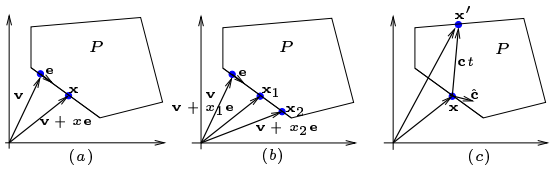

Fig. 7. (a) Representation of edges; (b) Representation of an interval; (c) One-step successor.

a one-dimensional coordinate system to represent points, the question now is to take advantage of it to compute the successor of a point or an interval.

Let $e = \langle e^l, e^u \rangle \in in(P)$ and $e' = \langle e'^l, e'^u \rangle \in out(P)$. For $\mathbf{x} = (e, x)$ and $\mathbf{c} \in \phi(P)$, we denote by $\mathsf{Succ}^{\mathbf{c}}_{e,e'}(x)$ the unique $\mathbf{x}' = (e', x')$ such that $\mathbf{x}' = \mathbf{x} + \mathbf{c}t$ for some $t > 0$. The point (e', x') is the successor of (e, x) in the direction $\mathbf{c}$ (see Fig.7(c)). Expanding, $\mathbf{v}' + x'\mathbf{e}' = \mathbf{v} + x\mathbf{e} + t\mathbf{c}$. Multiplying both expressions by $\hat{\mathbf{c}}$ we obtain that $(\mathbf{v}' + x'\mathbf{e}')\hat{\mathbf{c}} = \mathbf{v} \cdot \hat{\mathbf{c}} + x\mathbf{e}\hat{\mathbf{c}}$, i.e. $x'(\mathbf{e}' \cdot \hat{\mathbf{c}}) = x(\mathbf{e} \cdot \hat{\mathbf{c}}) + (\mathbf{v} - \mathbf{v}') \cdot \hat{\mathbf{c}}$. Thus, $x' = \mathsf{Succ}^{\mathbf{c}}_{e,e'}(x) = \frac{\mathbf{e} \cdot \hat{\mathbf{c}}}{\mathbf{e}' \cdot \hat{\mathbf{c}}} x + \frac{\mathbf{v} - \mathbf{v}'}{\mathbf{e}' \cdot \hat{\mathbf{c}}} \cdot \hat{\mathbf{c}}$ and $x' \in \langle e'^l, e'^u \rangle$. Indeed, putting $\alpha(\mathbf{c}) = \frac{\mathbf{e}\hat{\mathbf{c}}}{\mathbf{e}'\hat{\mathbf{c}}}$, and $\beta(\mathbf{c}) = \frac{\mathbf{v} - \mathbf{v}'}{\mathbf{e}' \cdot \hat{\mathbf{c}}} \cdot \hat{\mathbf{c}}$ we have the following result.

Lemma 7. *The function* $\mathsf{Succ}^{\mathbf{c}}_{e,e'}(x) = \alpha(\mathbf{c})x + \beta(\mathbf{c}) \cap \langle e'^l, e'^u \rangle$ *is truncated affine.*

$\widetilde{\mathsf{Succ}}_{e,e'}(x)$ will denote the *non-truncated* function $\alpha(\mathbf{c})x + \beta(\mathbf{c})$. The notion of *successor* can be extended on all possible directions $\mathbf{c} \in \phi(P)$ and it can be applied to any subset $S \subseteq \langle e^l, e^u \rangle$ and in particular to intervals $\langle l, u \rangle$:

Lemma 8. *Let* $\phi(P) = \angle^{\mathbf{b}}_{\mathbf{a}}$, $\mathbf{x} = (e, x)$ *and* $\langle l, u \rangle \subseteq \langle e^l, e^u \rangle$. *Then:*

1. $\mathsf{Succ}_{e,e'}(x) = \bigcup_{\mathbf{c} \in \phi(P)} \mathsf{Succ}^{\mathbf{c}}_{e,e'}(x) = \langle \widetilde{\mathsf{Succ}}^{\mathbf{b}}_{e,e'}(x), \widetilde{\mathsf{Succ}}^{\mathbf{a}}_{e,e'}(x) \rangle \cap \langle e'^l, e'^u \rangle$;
2. $\mathsf{Succ}_{e,e'}(\langle l, u \rangle) = \langle \widetilde{\mathsf{Succ}}^{\mathbf{b}}_{e,e'}(l), \widetilde{\mathsf{Succ}}^{\mathbf{a}}_{e,e'}(u) \rangle \cap \langle e'^l, e'^u \rangle$.

The successor operator will be used as a building block in the reachability algorithm. It can be naturally extended on edge signatures: for $w = e_1, e_2, \ldots, e_n$ let

$$\mathsf{Succ}_w(I) = \mathsf{Succ}_{e_{n-1},e_n} \circ \ldots \circ \mathsf{Succ}_{e_2,e_3} \circ \mathsf{Succ}_{e_1,e_2}(I)$$

that by Lemma 5 is truncated affine. Notice that since we use edge signatures the semi-group property takes the following form.

Lemma 9. *For any edge signatures w and v and an edge e, $\mathsf{Succ}_{ew} \circ \mathsf{Succ}_{ve} = \mathsf{Succ}_{vew}$.*

Example. Let us come back to the example of the swimmer trying to escape from a whirlpool in a river (see Fig. 1). Suppose that the swimmer is following a trajectory with edge signature $(e_1 \dots e_8)^*$. It is not difficult to find a representation of the edges such that for each edge e_i, $(e_i^l, e_i^u) = (0, 1)$. Besides, the truncated affine successor functions are:

$$\mathsf{Succ}_{e_1e_2}(x) = \left[\tfrac{x}{2}, \tfrac{x}{2}\right] \cap (0,1) \qquad \mathsf{Succ}_{e_2e_3}(x) = \left[x - \tfrac{3}{10}, x + \tfrac{2}{15}\right] \cap (0,1)$$
$$\mathsf{Succ}_{e_ie_{i+1}}(x) = [x, x] \cap (0,1), \text{ for all } i \in [3,7] \qquad \mathsf{Succ}_{e_8e_1}(x) = \left[x + \tfrac{1}{5}, x + \tfrac{1}{5}\right] \cap (0,1)$$

The successor function for the loop $s = e_1 \dots e_8$ is obtained by composition of the above functions as follows. Let us first compute

$$\begin{aligned}\mathsf{Succ}_{e_1e_2e_3}(l,u) &= \mathsf{Succ}_{e_2e_3} \circ \mathsf{Succ}_{e_1e_2}(l,u)\\ &= [\tfrac{l}{2} - \tfrac{3}{10}, \tfrac{u}{2} + \tfrac{2}{15}] \cap (0 - \tfrac{3}{10}, 1 + \tfrac{2}{15}) \cap (0,1)\\ &= [\tfrac{l}{2} - \tfrac{3}{10}, \tfrac{u}{2} + \tfrac{2}{15}] \cap (0,1)\end{aligned}$$

Since $\widetilde{\mathsf{Succ}}_{e_ie_{i+1}}$ for $i \in [3,7]$ are the identity functions, we have that

$$\begin{aligned}\mathsf{Succ}_{e_1\dots e_8}(l,u) &= \mathsf{Succ}_{e_8e_1} \circ \mathsf{Succ}_{e_1e_2e_3}(l,u)\\ &= [\tfrac{l}{2} - \tfrac{3}{10} + \tfrac{1}{5}, \tfrac{u}{2} + \tfrac{2}{15} + \tfrac{1}{5}] \cap (\tfrac{0}{2} + \tfrac{1}{5}, \tfrac{1}{2} + \tfrac{1}{5}) \cap (0,1)\\ &= [\tfrac{l}{2} - \tfrac{1}{10}, \tfrac{u}{2} + \tfrac{1}{3}] \cap (\tfrac{1}{5}, 1)\end{aligned}$$

By Lemma 6 we have that $l^* = \frac{-\frac{1}{10}}{1-\frac{1}{2}} = -\frac{1}{5}$, and $u^* = \frac{\frac{1}{3}}{1-\frac{1}{2}} = \frac{2}{3}$. □

7 Reachability Analysis

The algorithm for solving the reachability problem between two points $\mathbf{x}_0 = (e_0, x_0)$ and $\mathbf{x}_f = (e_f, x_f)$ is depicted in Fig. 8. The proofs of soundness and termination are given in the extended version ([5]). It works as follows.

Reach. From the section above we know that there exists a finite number of type signatures of the form $r_1, s_1, \dots, r_n, s_n, r_{n+1}$. Moreover, the type signatures are restricted to those with $e_0 = first(r_1)$ and $e_f = last(r_{n+1})$. Given such a set of type signatures $type(e_0, e_f)$, the algorithm $Reach(\cdot)$ is guaranteed to terminate, answering YES if $\mathbf{x}_f$ is reachable from $\mathbf{x}_0$ or NO otherwise. Reachability from $\mathbf{x}_0$ to $\mathbf{x}_f$ with fixed signature w is tested by the function $Reach_{type}(x_0, x_f, w)$.

Reach$_{type}$. Let the type w have the form $w = r_1, s_1, \ldots, r_n, s_n, r_{n+1}$. Put $f_i = first(s_i)$ and $ex_i = first(r_{i+1})$ if r_{i+1} is non-empty and f_{i+1} otherwise (i.e. ex_i is the edge to which the trajectory exits from the loop s_i). Let us say that a type signature w has a $\mathsf{loop}_{\mathsf{end}}$ property if $r_{n+1} = first(s_n)$, i.e. signatures of type w terminate by several repetitions of the last loop. This algorithm uses two functions: $Test(S, s, x)$ that answers whether x is reachable from a set S (represented as a finite union of intervals) in the loop s (formally, whether $x \in \mathsf{Succ}_{s^+ first(s)}(I)$); and the function $Exit(S, s, e)$ that for an initial set S, a loop s, and an edge e (not in this loop) finds all the points on e reachable by making s several times and then exiting to e (formally, it computes $\mathsf{Succ}_{s^+e}(I)$, which is always a finite union of intervals). Since we know how to calculate the successor of a given interval in one and in several steps ($\mathsf{Succ}_{ee'}(\cdot)$ and $\mathsf{Succ}_r(\cdot)$), in order to implement $Test(\cdot)$ and $Exit(\cdot)$ it remains to show how to analyze the (simple) cycles s_i and eventually their continuation. Both algorithms $Test(\cdot)$ and $Exit(\cdot)$ start by qualitative analysis of the cycle implemented in the function $Analyze(I, s)$. This analysis proceeds as follows.

Analyze. The function $Analyze(I, s)$ returns the kind of qualitative behavior of the interval $I = \langle l, u\rangle$ under the loop s. Let s be a simple cycle, $f = first(s)$ its first edge, and $I = \langle l, u\rangle \subset f$ an initial interval and $\mathsf{Succ}_{s,f}(x) = \widetilde{\mathsf{Succ}}_{s,f}(x) \cap \langle L, U\rangle$. The first thing to do is to determine the qualitative behavior of the leftmost and rightmost trajectories of the interval endpoints in the cycle. This can be done without iterating Succ_{sf}. Indeed, by Lemma 6, we can compute the limits $(l_1^*, u_1^*) = \lim_{n\to\infty} \widetilde{\mathsf{Succ}}^n_{s,f}(\langle l, u\rangle)$ (notice that those are limits only for the *non-truncated operator* $\widetilde{\mathsf{Succ}}$), not taking into account that the edges are possible bounded (we use Lemma 4) and compare these limit points corresponding to unrestricted dynamics with L and U. There are five possibilities:

1. STAY The cycle is not abandoned by any of the two trajectories: $L \le l^* \le u^* \le U$.

function $Reach(\mathbf{x}_0, \mathbf{x}_f)$ $R = \mathsf{false}$ **for each** $w \in type(e_0, e_f)$ $R = R \vee Reach_{type}(x_0, x_f, w)$ $\longleftarrow R$	**function** $Reach_{type}(x_0, x_f, w)$: $S = \mathsf{Succ}_{r_1 f_1}(x_0)$ **for** $i = 1$ *to* $n-1$ $S = \mathsf{Succ}_{r_{i+1} f_{i+1}}(Exit(S, s_i, ex_i))$ **if** $\mathsf{loop}_{\mathsf{end}}(w)$ **then** $\longleftarrow Test(S, s_n, x_f)$ **else** $\longleftarrow x_f \in \mathsf{Succ}_{r_{n+1}}(Exit(S, s_n, ex_n))$?
function $Exit(S, s, ex)$ $E = \emptyset$ **for each** $I \in S$ such that $\mathsf{Succ}_{s,f}(I) \neq \emptyset$ $E = E \cup Exit_{Analyze}(\mathsf{Succ}_{s,f}(I), s, ex)$ $\longleftarrow E$	**function** $Test(S, s, x)$ $R = \mathsf{false}$ **for each** $I \in S$ such that $\mathsf{Succ}_{s,f}(I) \neq \emptyset$ $R = R \vee Test_{Analyze}(\mathsf{Succ}_{s,f}(I), s, x)$ $\longleftarrow R$

Fig. 8. Main algorithm.

2. **DIE** The right trajectory exits the cycle through the left (consequently the left one also exits) or the left trajectory exits the cycle through the right (consequently the right one also exits). In symbols, $u^* < L \vee l^* > U$.
3. **EXIT-BOTH** Both trajectories exit the cycle (the left one through the left and the right one through the right): $l^* < L \wedge u^* > U$.
4. **EXIT-LEFT** The leftmost trajectory exits the cycle but not the other: $l^* < L \leq u^* \leq U$.
5. **EXIT-RIGHT** The rightmost trajectory exits the cycle but not the other: $L \leq l^* \leq U < u^*$.

Exit. The function $Exit(S, s, ex)$ should return $\mathsf{Succ}_{s^+ex}(S)$. Both the argument S and the result are finite collections of intervals. The exploration is made for each initial interval separately. Notice that the call $\mathsf{Succ}_{s,f}(I)$ ensures that $I \subseteq \langle L, U \rangle$. All the work for each initial interval I is done by the function $Exit_{Analyze}(I, s, ex)$ which launches the $Analyze(\cdot)$ procedure described above and last, according to the result of this analysis launches one of five specialized procedures $Exit_{STAY}$, $Exit_{LEFT}$, $Exit_{RIGHT}$, $Exit_{BOTH}$, $Exit_{DIE}$ which calculates the exit set (Fig. 10).

function $Search(I, x)$ **while** $Found(I, x) =$ NOTYET $I = \mathsf{Succ}_{s,f}(I)$ $\longleftarrow Found(I, x)$	**function** $Found(I, x)$ **cases** $x \in I$: $\longleftarrow$ YES $I = \emptyset$: $\longleftarrow$ NO $x < I \wedge l \uparrow$: $\longleftarrow$ NO $x > I \wedge u \downarrow$: $\longleftarrow$ NO **else** : $\longleftarrow$ NOTYET

Fig. 9. *Search* and *Found.*

Test. The upper-level structure is the same as for EXIT: each initial interval is treated separately by $Test_{Analyze}$, which makes one turn of the loop, calls *Analyze* and delegates all the remaining to one of the five specialized functions $Test_{STAY}$, $Test_{LEFT}$, $Test_{RIGHT}$, $Test_{BOTH}$, $Test_{DIE}$ (Fig. 10). The five specialized *Test* functions use the following two procedures (Fig. 9). The function $Found(I, x)$ determines if the current interval I contains x (YES), does not contain x and moves in the opposite direction (NO), or none of both these cases (NOTYET). $Found(I, x)$ uses the fact that the sequences l_n and u_n are increasing or decreasing (which can be easily determined at the stage of the preliminary analysis of the loop): $l \uparrow$ means that the sequence $l, l_1, l_2, \ldots$ of successive successors of l is increasing whereas $l \downarrow$ means that the sequence is decreasing, and similarly for $u \uparrow$ and $u \downarrow$. The function $Search(I, x)$ iterates the loop s until the previous function *Found* gives a definite answer YES or NO (Fig. 9). It is used only when its convergence is guaranteed.

	Exit	*Test*
STAY	**function** $Exit_{STAY}(I,s,ex)$ $\longleftarrow \emptyset$	**function** $Test_{STAY}(I,s,x)$ **cases** $l^* < x < u^* :$ $\longleftarrow$ YES $x \leq l^* \wedge l \downarrow :$ $\longleftarrow$ NO $x \geq u^* \wedge u \uparrow :$ $\longleftarrow$ NO **else** : $\longleftarrow Search(I,x)$
DIE	**function** $Exit_{DIE}(I,s,ex)$ $f = first(s)$ $S = \emptyset$ **repeat** $I = \mathsf{Succ}_{sf}(I)$ $S = S \bigcup \mathsf{Succ}_{s,ex}(I)$ **until** $I = \emptyset$ $\longleftarrow$ S	**function** $Test_{DIE}(I,s,x)$ $\longleftarrow Search(I,x)$
BOTH	**function** $Exit_{BOTH}(I,s,ex)$ $\longleftarrow \mathsf{Succ}_{s,ex}(\langle L,U\rangle)$	**function** $Test_{BOTH}(I,s,x)$ $\longleftarrow x \in \langle L,U\rangle ?$
LEFT	**function** $Exit_{LEFT}(I,s,ex)$ $\longleftarrow \mathsf{Succ}_{s,ex}(\langle L,u\rangle)$	**function** $Test_{LEFT}(I,s,x)$ **cases** $x \in \langle L,u^*\rangle :$ $\longleftarrow$ YES $x < \langle L,u^*\rangle :$ $\longleftarrow$ NO $\langle L,u^*\rangle < x \wedge u \uparrow :$ $\longleftarrow$ NO **else** : $\longleftarrow Search(I,x)$
RIGHT	Similar to the previous case.	Similar to the previous case.

Fig. 10. *Exit* and *Test*.

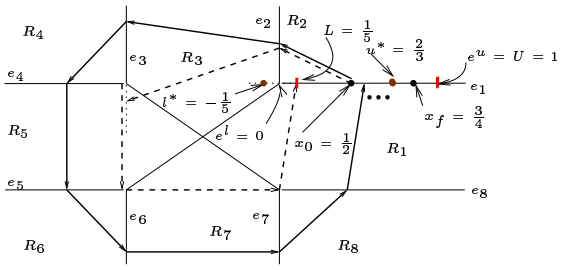

Fig. 11. Example: $\mathbf{x}_f = (e_1, \frac{3}{4})$ is not reachable from $\mathbf{x}_0 = (e_1, \frac{1}{2})$ $(u^* < \frac{3}{4})$.

Example. Consider again the swimmer. Let $\mathbf{x}_0 = (e_1, \frac{1}{2})$ be her initial position. We want to decide whether she is able to escape from the whirlpool and reach the final position $\mathbf{x}_f = (e_1, \frac{3}{4})$. Recall that $l^* = \frac{-\frac{1}{10}}{1-\frac{1}{2}} = -\frac{1}{5}$ and $u^* = \frac{\frac{1}{3}}{1-\frac{1}{2}} = \frac{2}{3}$. Thus, by the *Analyze* function we know that the cycle behaves as an Exit-LEFT and applying the function $Test_{LEFT}$ we obtain that $\mathbf{x}_f = (e_1, \frac{3}{4})$ is not reachable from $\mathbf{x}_0 = (e_1, \frac{1}{2})$ because we have that $u \uparrow$ and $u^* < x_f$ ($\frac{2}{3} < \frac{3}{4}$) (Fig. 11). □

From all the results above we have the following theorem.

Theorem 1 (Point–to–Point Reachability). *The point-to-point, edge-to-edge and region-to-region reachability problems for SPDI systems are decidable.* □

8 Concluding Remarks

We have presented an algorithm for solving the reachability problem for simple planar differential inclusions. The novelty of the approach for the domain of Hybrid System is the combination of two techniques, namely, the representation of the two-dimensional continuous dynamics as a one-dimensional discrete system (due to Poincaré), and the characterization of the set of qualitative behaviors of the latter as a finite set of types of signatures.

One possible direction of future work is to try to apply the same method for solving the *parameter synthesis problem* for SPDI's, that is, for any two points, $\mathbf{x}_0$ and $\mathbf{x}_f$, assign a constant slope $\mathbf{c}_P \in \phi(P)$ to every region P such that $\mathbf{x}_f$ is reachable from $\mathbf{x}_0$, or conclude that such an assignment does not exist. Clearly, the decidability of the reachability problem does not imply the decidability of the parameter synthesis one.

Another question that naturally arises is decidability of the reachability problem for hybrid automata whose locations are equipped with SPDI's. We can certainly find (stringent) conditions, such as planarity of the automaton, "memoryless" resets, etc., under which decidability follows almost straightforwardly from the decidability of SPDI's. On the other hand, it is not difficult to see that this problem, without such conditions, is equivalent to deciding whether given a piece-wise linear map f on the unit interval and a point x in this interval, the sequence of iterates x, $f(x)$, $f(f(x))$, and so on, reaches some point y. This last question is still open [13]. And last but not the least, another interesting issue is the complexity analysis of the algorithm. It should be based on counting all "feasible" types of signatures. Our finiteness argument of lemma 4 gives a doubly exponential estimation, which can certainly be improved.

References

1. R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. TCS **138** (1995) 3–34.
2. R. Alur and D.L. Dill. A theory of timed automata. TCS **126** (1994) 183–235.

3. E. Asarin, O. Bournez, T. Dang, and O. Maler. Reachability analysis of piecewise-linear dynamical systems. In *HSCC'00*, 20–31. LNCS 1790, Springer Verlag, 2000.
4. E. Asarin, O. Maler, and A. Pnueli. On the analysis of dynamical systems having piecewise-constant derivatives. TCS, **138** (1995) 35–65.
5. E. Asarin, G. Schneider and S. Yovine. On the Decidability of the Reachability Problem for Planar Differential Inclusions. VERIMAG Technical Report. 2001. http://www-verimag.imag.fr/∼gerardo.
6. O. Botchkarev and S. Tripakis. Verification of hybrid systems with linear differential inclusions using ellipsoidal approximations. In *HSCC'00*. LNCS 1790, Springer Verlag, 2000.
7. M. Broucke. A geometric approach to bisimulation and verification of hybrid systems. In *HSCC'99*. LNCS 1569, Springer Verlag, 1999.
8. K. Čerāns and J. Vīksna. Deciding reachability for planar multi-polynomial systems. In *Hybrid Systems III*. LNCS 1066, Springer Verlag, 1996.
9. T. Dang and O. Maler. Reachability analysis via face lifting. In *HSCC'98*, 96–109. LNCS 1386, Springer Verlag, 1998.
10. J. Della Dora and S. Yovine. Looking for a methodology for analyzing hybrid systems. Submitted to ECC 2001, 2000.
11. M. R. Greenstreet and I. Mitchell. Reachability analysis using polygonal projections. In *HSCC'99*. LNCS 1569, 103–116. Springer Verlag, 1999.
12. T.A. Henzinger, P.W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? In *27th Annual Symposium on Theory of Computing*, 373–382. ACM Press, 1995.
13. P. Koiran. My favourite problems. http://www.ens-lyon.fr/˜koiran/problems.html.
14. A.B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In *HSCC'00*. LNCS 1790, Springer Verlag, 2000.
15. G. Lafferriere, G. J. Pappas, and S. Yovine. A new class of decidable hybrid systems. In *HSCC'99*. LNCS 1569, 137–151. Springer Verlag, 1999.
16. O. Maler and A. Pnueli. Reachability analysis of planar multi-linear systems. In *CAV'93*. LNCS 697, 194–209. Springer Verlag, 1993.