Reachability Analysis of Generalized Polygonal Hybrid Systems

Gerardo Schneider
Department of Informatics
University of Oslo
gerardo@ifi.uio.no

ABSTRACT

A polygonal hybrid system (SPDIs) is a planar hybrid system, whose dynamics is defined by constant differential inclusions, for which the reachability problem is decidable. The decidability result is based, among other things, on the fact that a trajectory cannot enter and leave a given region through the same edge. SPDIs without such an assumption are called Generalized SPDIs (GSPDIs). In this paper we show that in general it is not possible to reduce GSPDI reachability to SPDI reachability. Furthermore, we provide a terminating algorithm implementing a semi-test for GSPDI reachability, based on that for SPDIs.

Categories and Subject Descriptors

D.2.4 [**Software**]: Software/Program Verification

General Terms

Verification

Keywords

Hybrid systems, verification, reachability, decidability, GSPDI.

1. INTRODUCTION

Systems combining discrete and continuous behaviors, as for instance robots and chemical processes, are called hybrid systems. Though hybrid automata [1] have gained popularity as a specification formalism for hybrid systems, their analysis remains a big challenge as most problems are undecidable. An interesting and still decidable (w.r.t reachability) class of hybrid systems is the so-called Polygonal Hybrid System (SPDI for short, [5, 7]) which is a subclass of hybrid systems on the plane whose dynamics is defined by constant differential inclusions. SPDIs are a generalization of PCDs (deterministic systems with Piece-wise Constant Derivatives) for which it has been shown that the reachability problem is decidable for the planar case but undecidable for three and higher dimensions [2].

SAC'08 March 16-20, 2008, Fortaleza, Ceará, Brazil

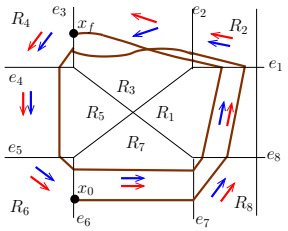

Figure 1: Example of an SPDI.

Informally, an SPDI consists of a partition of the plane into polygonal regions where in each region the dynamics is given by two vectors determining the possible directions a trajectory might take; a simple SPDI, and a typical trajectory segment, are depicted in Fig. 1. The constructive proof for deciding reachability on SPDIs given in [5] relies, among other things, on the fact that the SPDI has the *goodness* property, i.e. the dynamics of any region does not allow a trajectory to traverse any edge of the polygon defining the region in both directions. We say that an SPDI without the goodness assumption is a *Generalized SPDI* –or GSPDI for short. The frontier between decidable and undecidable low-dimensional hybrid systems has been studied in [4, 9]. In those papers, a wide are of classes was established for which the decidability/undecidability issue is still open. To date, it is not known whether reachability for GPSDIs is decidable or not.

In this paper we show that there is no *structure-preserving reduction* from GSPDI reachability to SPDI reachability. However, the SPDI algorithm reveals to be very useful to give an algorithm implementing a *semi-test* (that gives YES/don't know answers) for GSPDI reachability. We prove soundness and termination of the algorithm.

The paper is organized as follows. In next section we explain informally the problems arising when relaxing goodness while in Section 3 we give some preliminaries on SPDIs. In Section 4 we present GSPDIs, whereas Section 5 is concerned with GSPDI reachability analysis. We conclude in the last section.

2. ON GOODNESS

One of the key concepts of SPDI reachability is that of goodness. In Fig. 2 we can see a good region (isolated from the rest of the SPDI), where the two vectors **a** and **b** determine the impossibility of a trajectory to enter and leave the region P through the same edge of the polygon delimiting the region. On the other hand, the figure on the right

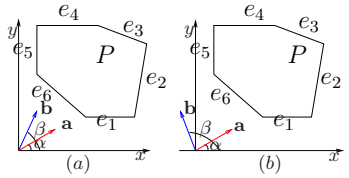

Figure 2: a) A good region. b) A bad region.

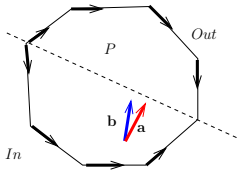

Figure 3: Ordering of edges on an SPDI.

shows a "bad" region: Both e_2 and e_5 can be crossed in both directions by a trajectory entering and leaving P.

Why Goodness is Good? The algorithm presented in [5, 7] for deciding reachability on SPDI heavily depends on the pre-processing of trajectory segments to guarantee that it is possible to list all the possible sets of *signatures*, i.e., those sequences of edges of the SPDI traversed by all the possible trajectories between two points. This is of course not possible in general as there are infinitely many such trajectories. However, a qualitative analysis allows to prove that indeed there is a finite number of abstract signatures (called *types of signatures*) preserving reachability.

The above is achieved by performing the following steps. 1) Simplification of trajectory segments: straightening them and removing self-crossings. Given an arbitrary trajectory segment from one point to another, it is possible to get a piecewise constant derivative trajectory segment without self-crossing. 2) Abstraction of trajectory segments into *signatures*, considering the sequence of traversed edges. This result is based on the *Poincaré map* [8], that relates n-dim continuous-time systems with $(n-1)$-dim discrete-time systems. 3) *Factorization* of signatures in a convenient way, having only sequences of edges and simple cycles. This factorization allows to have a nice representation of signatures. 4) Abstraction of factorized signatures into *types of signatures*, that are signatures without taking into account the number of times each simple cycle is iterated.

Many of the lemmas for proving that the above provides a finite number of types signatures critically depend on the goodness assumption, affecting the constructive proof given for deciding reachability of SPDIs.

Why Relaxing Goodness is not so Good? The main question now is, how much do we need to depend on the goodness assumption to prove decidability of reachability of SPDIs? In other words, let us consider SPDIs without the goodness assumptions (GSPDIs). Is reachability still decidable? We have two alternatives: a) Reduce GSPDI reachability to SPDI reachability. This would imply to restate the proofs to make them independent of the goodness assumption. b) Provide a completely new decidability proof for GSPDI. This will probably need to use different techniques and results than the ones used for SPDIs. The first alternative above seems the most straightforward and easy to do. However, as we show in this paper it is not possible to reduce GSPDI reachability to SPDI reachability with a structure-preserving transformation (a more precise definition of such a reduction will be given in Section 5.1). This is done by proving that it is not possible, in general, to simplify certain trajectories entering and leaving a given region through the same edge, to trajectories behaving as in SPDIs. One of the main problems when relaxing goodness is that a region cannot be bi-partitioned anymore into two semi-planes were all the edges in one semi-plane can be traversed only in one direction, w.r.t. the region, and all the edges in the other semi-plane can be traversed only in the other direction. As shown in Fig. 3 the goodness assumption determines a certain "contiguity" of *entry-only* edges (In) and *exit-only* edges (Out) belonging to two disjoint sub-regions. Some lemmas and proofs of soundness of the SPDI reachability algorithm depend on this contiguity. If we relax goodness, we should be able to re-prove all such results without assuming the contiguity of entry-only and exit-only edges.

This let us with the second alternative. Unfortunately, to date there is no proof of decidability (nor of undecidability) to the reachability problem on GSPDIs. On the other hand, we will show that we can relax the goodness assumption as to give a terminating procedure for GPSDI reachability that whenever gives YES, the answer is correct, and otherwise the answer is not known.

3. PRELIMINARIES

This section is more technical, recalling the main definitions and concepts needed to understand the rest of the paper. For a more detailed presentation see [7].

Let $\mathbf{a} = (a_1, a_2), \mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$ and $\alpha, \beta \in \mathbb{R}$. The *inner product* of two vectors $\mathbf{a} = (a_1, a_2)$ and $\mathbf{x} = (x_1, x_2)$ is defined as $\mathbf{a} \cdot \mathbf{x} = a_1 x_1 + a_2 x_2$. We denote by $\hat{\mathbf{x}}$ the vector $(x_2, -x_1)$ obtained from $\mathbf{x}$ by rotating clockwise by the angle $\pi/2$. Notice that $\mathbf{x} \cdot \hat{\mathbf{x}} = 0$.

An *angle* $\angle_{\mathbf{a}}^{\mathbf{b}}$ on the plane, defined by two non-zero vectors $\mathbf{a}, \mathbf{b}$ is the set of all positive linear combinations $\mathbf{x} = \alpha\ \mathbf{a} + \beta\ \mathbf{b}$, with $\alpha, \beta \geq 0$, and $\alpha + \beta > 0$. We can always assume that $\mathbf{b}$ is situated in the counter-clockwise direction from $\mathbf{a}$.

DEFINITION 1. *A* polygonal hybrid system *(SPDI) is a pair $\mathcal{H} = \langle \mathbb{P}, \phi \rangle$, where $\mathbb{P}$ is a finite partition of the plane (with each $P \in \mathbb{P}$ being a convex polygon), called the* regions *of the SPDI, and ϕ is a function which associates a pair of vectors to each polygon: $\phi(P) = (\mathbf{a}_P, \mathbf{b}_P)$. In an SPDI every point on the plane has its dynamics defined according to which polygon it belongs to: if $\mathbf{x} \in P$, then $\dot{\mathbf{x}} \in \angle_{\mathbf{a}_P}^{\mathbf{b}_P}$.*

Let $E(P)$ be the set of edges of P. We say that $e \in E(P)$ is an *entry-only* 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-only* of P if the same condition holds for some $\epsilon < 0$. We denote by $In(P) \subseteq E(P)$ the set of all entry-only edges of P and by $Out(P) \subseteq E(P)$ the set of all exit-only edges of P.

ASSUMPTION 1. *All the edges in $E(P)$ are either entry-only or exit-only, that is, $E(P) = In(P) \cup Out(P)$. We say then that all the regions in an SPDI are* good *or that they have the* goodness *property.*

In SPDIs, *sliding* on an edge is not allowed. That means that a trajectory segment reaching an edge being exit-only to two adjacent regions will block.

EXAMPLE 1. *In Fig. 2-(a), region P (with $\phi(P) = (\mathbf{a}, \mathbf{b})$) is good, since all are entry-only or exit-only edges. Fig. 2-(b) shows a region that is not good: edges e_2 and e_5 are not in $In(P) \cup Out(P)$.* ■

A *trajectory segment* of an SPDI is a continuous function $\xi : [0, T] \to \mathbb{R}^2$ which is smooth everywhere except in a discrete set of points, and such that for all $t \in [0, T]$, if $\xi(t) \in P$ and $\dot{\xi}(t)$ is defined then $\dot{\xi}(t) \in \phi(P)$. The *signature*, denoted $\mathsf{Sig}(\xi)$, is the ordered sequence of edges traversed by the trajectory segment, that is, $e_1, e_2, \ldots$, where $\xi(t_i) \in e_i$ and $t_i < t_{i+1}$. If $T = \infty$, a trajectory segment is called a *trajectory*.

EXAMPLE 2. *The SPDI illustrated in Fig. 1 contains 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* $\mathbf{b}_3 = (-1, -\frac{1}{4})$, $\mathbf{a}_4 = \mathbf{b}_4 = (-1, -1)$, $\mathbf{a}_5 = \mathbf{b}_5 = (0, -1)$, $\mathbf{a}_6 = \mathbf{b}_6 = (1, -1)$, $\mathbf{a}_7 = \mathbf{b}_7 = (1, 0)$, $\mathbf{a}_8 = \mathbf{b}_8 = (1, 1)$. ■

3.1 Successors and predecessors

Given an SPDI, we fix a one-dimensional coordinate system on each edge to represent points lying on edges. For notational convenience, we will use e to denote both the edge and its one-dimensional representation. Accordingly, we write $\mathbf{x} \in e$ or $x \in e$, to mean "point $\mathbf{x}$ in edge e with coordinate x in the one-dimensional coordinate system of e". The same convention is applied to sets of points of e represented as intervals (e.g., $\mathbf{x} \in I$ or $x \in I$, where $I \subseteq e$) and to trajectories (e.g., "ξ starting in x" or "ξ starting in $\mathbf{x}$").

Now, let $P \in \mathbb{P}$, $e \in In(P)$ and $e' \in Out(P)$. For $I \subseteq e$, $\mathsf{Succ}_{ee'}(I)$ is the set of all points in e' reachable from some point in I by a trajectory segment $\xi : [0, t] \to \mathbb{R}^2$ in P (i.e., $\xi(0) \in I \wedge \xi(t) \in e' \wedge \mathsf{Sig}(\xi) = ee'$). Given $I = [l, u]$, $\mathsf{Succ}_{ee'}(I) = F(I \cap S_{ee'}) \cap J_{ee'}$, where $S_{ee'} \in e$ and $J_{ee'} \in e'$ are intervals[1], $F([l, u]) = \langle f_l(l), f_u(u) \rangle$[2] and f_l and f_u are affine functions (a function $f : \mathbb{R} \to \mathbb{R}$ is affine iff $f(x) = ax + b$ with $a > 0$).

3.2 Qualitative analysis of simple edge-cycles

Let $\sigma = e_1 \cdots e_k e_1$ be a simple edge-cycle, i.e., $e_i \neq e_j$ for all $1 \leq i \neq j \leq k$. Let $\mathsf{Succ}_\sigma(I) = F(I \cap S_\sigma) \cap J_\sigma$ with $F = \langle f_l, f_u \rangle$.

Given a simple cycle σ, let l^* and u^* be the fix-points[3] of f_l and f_u, respectively, and $\langle L, U \rangle = S_\sigma \cap J_\sigma$. Any simple cycle can be of one of the following kinds. STAY: The cycle is not abandoned neither by the leftmost nor the rightmost trajectory, that is, $L \leq l^* \leq u^* \leq U$. DIE: The rightmost trajectory exits the cycle through the left (consequently the leftmost one also exits) or the leftmost trajectory exits the cycle through the right (consequently the rightmost one also exits), that is, $(u^* < L) \vee (l^* > U)$. EXIT-BOTH: The leftmost trajectory exits the cycle through the left and the rightmost one through the right, that is, $(l^* < L) \wedge (u^* > U)$. EXIT-LEFT: The leftmost trajectory exits the cycle (through the left) but the rightmost one stays inside, that is, $l^* < L \leq u^* \leq U$. EXIT-RIGHT: The rightmost trajectory exits the cycle (through the right) but the leftmost one stays inside, that is, $L \leq l^* \leq U < u^*$.

The classification above provides useful information about the qualitative behavior of trajectories. Any trajectory that enters a DIE cycle will eventually quit it after a finite number of turns. If the cycle is STAY, all trajectories that happen to enter it will keep turning inside it forever. In all other cases, some trajectories will turn for a while and then exit, and others will continue turning forever. This information is crucial for solving the reachability problem for SPDIs, and provides a means to accelerate the analysis.

We recall now the *representation theorem* for SPDIs that allows to factorize the signatures (step 3 in Section 2) in a convenient way. The theorem not only guarantees the existence of the above representation for SPDIs but also provides a constructive way of doing so [7].

THEOREM 1. *Given an SPDI, let $\sigma = e_1 \ldots e_p$ be an edge signature, then it can always be written as $\sigma_{\mathcal{A}} = r_1 s_1^{k_1} \ldots r_n s_n^{k_n} r_{n+1}$, where for any $1 \leq i \leq n+1$, r_i is a sequence of pairwise different edges and for all $1 \leq i \leq n$, s_i is a simple cycle (i.e., without repetition of edges).*

This representation of signatures is the base to obtain *types of signatures* (step 4 in Section 2) with the following good properties

LEMMA 2. *Given an SPDI, let $\sigma = e_0 \ldots e_p$ be a feasible signature, then its type, $\mathsf{type}(\sigma) = r_1, s_1, \ldots, r_n, s_n, r_{n+1}$ satisfies the following properties:*

$\mathbf{P_1}$ *For every $1 \leq i \neq j \leq n+1$, r_i and r_j are disjoint;*

$\mathbf{P_2}$ *For every $1 \leq i \neq j \leq n$, s_i and s_j are different.*

The above lemma guarantees that there are only finitely many different types of signatures, ensuring termination of the SPDI reachability algorithm.

4. GSPDI

The *goodness* restriction (Assumption 1) was originally introduced to simplify treatment of trajectories to guarantee, among other things, that each region can be partitioned into *entry-only* and *exit-only* edges in an ordered way, a fact used in the proof of decidability of the reachability problem. Without goodness there are edges that are neither of entry-only nor of exit-only as shown in Fig. 2. This naturally leads to the following definition.

DEFINITION 2. *An edge $e \in P$ is an* inout *edge of P if e is neither an entry-only nor an exit-only edge of P.* ■

Note that formally speaking the definition of SPDI does not exclude inouts edges, however, to make a clear separation between SPDIs with the goodness assumption and those without such an assumption, we call the latter *generalized SPDI (GSPDI)*. Thus, in GSPDIs there are three kinds of edges: inouts, entry-only and exit-only edges.

Self-crossing of trajectory segments of SPDIs can be eliminated which allow us to consider only non-crossing trajectory (segments). The proof given in [7] can be extended

[1] The intervals $S_{ee'}$ and $J_{ee'}$ intuitively 'truncate' the domain and co-doamin of the successor function. See [7] for a more detailed presentation.

[2] $\langle \cdot, \cdot \rangle$ denotes an interval. For notational convenience, we do not make explicit whether intervals are open, closed, left-open or right-open, unless required for comprehension.

[3] The fix-point x^* is computed by solving a linear equation $f(x^*) = x^*$, which can be finite or infinite.

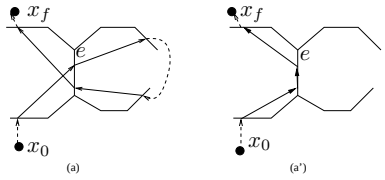

Figure 4: (a) Proper inout edge; (b) Sliding edge.

to deal with the case when the self-crossing trajectories involve inout edges, so the result still holds for GSPDIs. Thus in what follows we will consider only trajectory segments without self-crossings.

Notice that on GSPDIs a trajectory can "intersect" an edge at an infinite number of points because it can *slide* at it. Thus, a trace is not anymore a sequence of points but rather a sequence of intervals.

DEFINITION 3. *The* trace *of a trajectory ξ is the sequence* $\mathtt{trace}(\xi) = I_0 I_1 \ldots$ *of the intersection intervals of ξ with the set of traversed edges.* ■

A point interval $I = [\mathbf{x}, \mathbf{x}]$ will be written as $\mathbf{x}$ whenever no confusion might arise.

DEFINITION 4. *An* edge signature *(or simply a* signature*) of a GSPDI is a sequence of edges. The* edge signature of a trajectory ξ, $\mathsf{Sig}(\xi)$, *is the ordered sequence of traversed edges by the trajectory, that is,* $\mathsf{Sig}(\xi) = e_0 e_1 \ldots$, *with* $\mathtt{trace}(\xi) = I_0 I_1 \ldots$ *and* $I_i \subseteq e_i$. *The* region signature *of ξ is the sequence* $\mathsf{RSig}(\xi) = P_0 P_1 \ldots$ *of traversed regions, that is,* $e_i \in In(P_i)$. ■

Notice that in many cases the intervals of a trace are in fact points. We say that a trajectory with edge signature $\mathsf{Sig}(\xi) = e_0 e_1 \ldots e_i \ldots$ and trace $\mathtt{trace}(\xi) = I_0 I_1 \ldots I_i \ldots$ *interval-crosses* edge e_i if I_i is not a point.

Given a trajectory segment, we will make the difference between *proper inout* edges and *sliding* edges.

DEFINITION 5. *Let ξ be a trajectory segment from point $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$, with edge signature $\mathsf{Sig}(\xi) = e_0 \ldots e_i \ldots e_n$, and $e_i \in E(P)$ be an inout edge of P. We say that e_i is a* sliding edge of P for ξ *if ξ interval-crosses e_i, otherwise e is said to be a* proper inout edge of P for ξ. ■

We say that a trajectory segment ξ *slides* on an edge e if e is a sliding edge of P for ξ and ξ is said to be a *sliding trajectory* if there is at least one sliding edge $e \in \mathsf{Sig}(\xi)$.

EXAMPLE 3. *In Fig. 4-(a), e is a proper inout edge. Edge e on Fig. 4-(b) is a sliding edge.* ■

5. REACHABILITY ANALYSIS OF GSPDI

In order to get a sound *decision* algorithm, based on the SPDI algorithm, we would need to prove the following theoretical results: (1) Show that it is enough to consider trajectories without self-crossing (argument of its validity presented in the previous section); (2) Show that it is possible to eliminate all inout edges, preserving reachability; (3) Show that it is possible to eliminate all sliding edges, preserving reachability (Section 5.1); (4) Re-state and prove some results on SPDI reachability useful to GPSDI reachability analysis (Section 5.2); (5) Prove soundness and termination (Section 5.3).

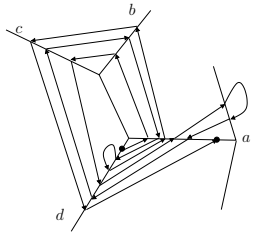

Figure 5: Counter-example for Proposition 1.

However, as we will see in Section 5.1 step (2) is not valid, and we thus get a *semi-test* algorithm for GSPDI reachability analysis instead, by proving the rest of the above steps.

Due to lack of space we do not provide full proofs here; see [10] for more details.

5.1 Simplification of Trajectory Segments

We first start by showing that the good properties of the representation theorem for SPDIs are not valid any longer for GSPDIs.

PROPOSITION 1. *Property* $\mathbf{P_2}$ *of the representation theorem for SPDIs (Lemma 2) does not hold in general for GSPDIs.*

PROOF SKETCH: Let ξ be a trajectory with signature $\mathsf{Sig}(\xi) = \sigma = e_0 \ldots e_i \ldots e_n \ldots$ of a given GSPDI. The proposition states that it is not possible in general to write σ in the form $\sigma_{\mathcal{A}} = r_1 s_1^{k_1} \ldots r_n s_n^{k_n} r_{n+1}$ with the properties stated in Lemma 2. The proof is done by providing a counter-example. A typical counter-example should allow to obtain a signature consisting of a clockwise spiral followed by a counter-clockwise spiral (or vice-versa) and then back to the first spiral. In such a case it is possible to find two simple cycles which are repeated in the type of signature. Let us consider the GSPDI of Fig. 5. To keep it simple we do not write down the dynamics of the regions and we assume that they are as to allow the segments of trajectories shown in the picture to be well-defined. In such a GSPDI it is possible to obtain the following type of signature: $r_1 s_1 r_2 s_2 r_3 s_3 \ldots$, where $s_1 = (abcd)$, $s_2 = (dcba)$, and $s_3 = (abcd)$. Since $s_1 = s_3$, then property $\mathbf{P_2}$ of Lemma 2 is not satisfied. □

The following lemma presents some typical cases where it is possible to eliminate proper inout edges.

LEMMA 3. *Let ξ be a trajectory segment with initial point $\mathbf{x}_0 \in e_0$ and final point $\mathbf{x}_f \in e_f$, with edge signature $\mathsf{Sig}(\xi) = e_0 \ldots e_i \ldots e_n$. If e_i is a proper inout edge then in some cases there exists a trajectory segment ξ' from $\mathbf{x}_0$ to $\mathbf{x}_f$ that traverses e_i in at most one sense (that is, e_i is either an entry-only or an exit-only, but no both).*

PROOF SKETCH. In Fig. 6-(a) we illustrate a typical case where edge e_i is a proper inout edge. After a straightforward algebraic vector manipulation, on the same lines of elimination of self-crossings, the trajectory segment shown in Fig. 6-(a') is obtained. □

Note that the above does not establish completeness of a reduction from GSPDIs into SPDIs reachability since there are cases where the above is not possible, as shown in the following proposition.

PROPOSITION 2. *Given a GSPDI, assume there exists a trajectory segment from points $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$, traversing*

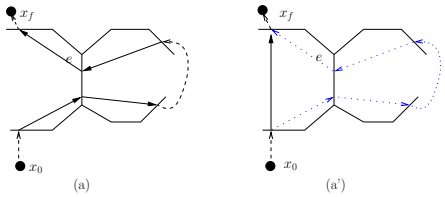

Figure 6: Inout case.

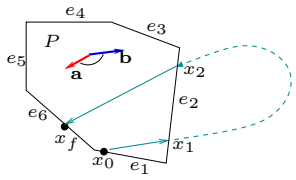

Figure 7: A non-eliminating inout edge.

inout edges in both directions. Then it is, in general, not possible to find a trajectory segment whose edge signature contains no proper inout edges (traversed in both directions), between them.

PROOF SKETCH: The GSPDI of Fig. 7 presents a typical example of an inout edge (e_2) which cannot be directly eliminated as to preserve that x_f is reachable from x_0. To keep the explanation simple we do not present here a formal GSPDI as counter-example. The example, however, sheds some light on the kind of GSPDI regions serving as counter-examples. It suffices to take any trajectory exiting a region through an edge (e_2 in the figure) and entering to the region again through the same edge, with a dynamics forbidding the sliding from the exit point (x_1) to the entry point (x_2). The trajectory must not have self-crossings. □

The above result is based on the fact that the underlying convex polygons of the GSPDI as well as its dynamics is fixed; the only restriction is the prohibition of trajectories entering and exiting a region through the same edge, though the dynamics still would allow to do so. Let $\mathcal{G}$ be a GSPDI, and $\mathcal{S}$ a GSPDI obtained from $\mathcal{G}$ with the above restriction, then we say that $\mathcal{S}$ *preserves the underlying structure* of $\mathcal{G}$, and that $\mathcal{S}$ is an *underlying SPDI* of $\mathcal{G}$ (notice that in general there are many underlying SPDIs for each GSPDI). We say that there is a *structure-preserving reduction* from the GSPDI reachability problem to the SPDI reachability problem if there is a transformation from any GSPDI instance $\mathcal{G}$ into an underlying SPDI instance $\mathcal{S}$ of $\mathcal{G}$, such that $\mathbf{Reach}(\mathcal{G}, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$ iff $\mathbf{Reach}_{SPDI}(\mathcal{S}, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$.

From the above proposition we conclude that it is not possible to reduce GSPDI reachability to SPDI reachability since we may miss some of the positive answers. We have then the following result.

PROPOSITION 3. *There is no structure-preserving reduction from the GSPDI reachability problem to the SPDI reachability problem.*

In what follows we concentrate on sliding edges; we show first that we can eliminate sliding edges.

LEMMA 4. *Let ξ be a trajectory segment from $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$ with edge signature $\mathsf{Sig}(\xi) = e_0 \dots e_i \dots e_f$. If e_i is a sliding edge for ξ then there exists a trajectory segment ξ' from $\mathbf{x}_0$ to $\mathbf{x}_f$ that does not slide on edge e_i.*

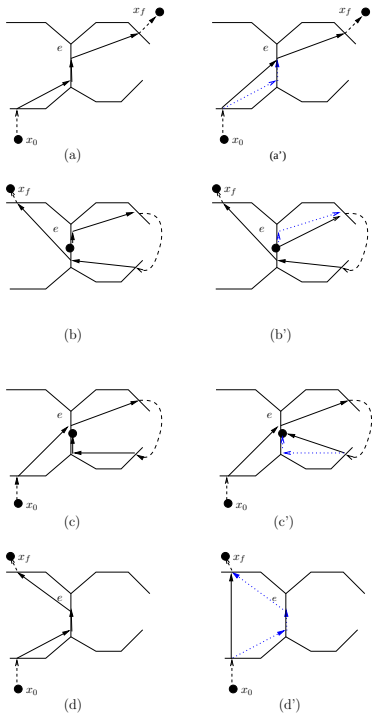

Figure 8: Sliding cases.

PROOF SKETCH. Sliding edges can arise in four different cases (plus the symmetric cases); they are shown in Fig. 8-(a) to (d). The corresponding primed figures (Fig. 8-(a') to (d')) show the transformation done in order to avoid sliding on edge e. We do not give a formal proof of completeness of the result here. Notice that indeed the above transformation is always possible since in all the cases the new obtained segment of trajectory can be expressed as a positive linear combination of two suitable existing segments of trajectory. Such two segments are the sliding segment, and another segment of trajectory with starting point at the beginning or the end of the sliding segment. □

As a consequence we have the following result, showing the existence of a non-sliding trajectory.

PROPOSITION 4. *If there exists a sliding trajectory segment from points $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$ then there always exists a non-sliding trajectory segment between them.*

PROOF. By induction on the number n of sliding edges of the signature of the trajectory segment using Lemma 4 in the induction step. □

Sliding is not easy to treat in general since an edge always belongs to two different regions with different dynamics. Thus a trajectory may be "allowed" to slide by one of the dynamics but not by the other. For our purposes we assume that at an inout edge a trajectory can slide if at least one of the dynamics allows so. This assumption does not affect the reachability analysis.

5.2 SPDI Results used in GSPDI Analysis

We will see in next subsection that the semi-test algorithm for reachability analysis of GSPDIs depends on the generation of all the possible underlying SPDIs obtained after fixing the inout edges as entry-only or exit-only edges.

In order to guarantee that we can still apply the reachability algorithm for SPDIs, we need to: (1) Redefine the edge-to-edge successor operator, Succ, to be able to deal with sliding edges; (2) "Topologically" rephrase and prove the results of [7] that use the contiguity between entry-only and exit-only edges in their proofs; (3) The proofs of soundness of the Exit-LEFT and Exit-STAY algorithms also rely on the contiguity hypothesis, and need thus to be re-proved (see [10]).

Concerning the first point above, note that it is convenient to define a (trivial) successor Succ_e where e is a single edge. The only way to do it preserving the semi-group property for SPDIs is to put $\mathsf{Succ}_e(I) = I$. Notice that for GSPDIs, however, we add the following cases in case e is an inout edge, with $I = \langle l, u\rangle$, and given $\langle L, U\rangle = S_e \cap J_e$: $\mathsf{Succ}_e(I) = \langle L, u\rangle$, or $\mathsf{Succ}_e(I) = \langle l, U\rangle$, depending on which direction e allows the sliding.

5.3 Reachability Algorithm

Given a GSPDI $\mathcal{H}$, we denote by $\mathcal{H}_{red} = \{\mathcal{H}_1, \ldots, H_n\}$ the set of all the underlying SPDIs obtained after fixing all the inout edges of $\mathcal{H}$ as entry-only or exit-only, considering all the possible permutations.

Let $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f)$ be the reachability algorithm for a GSPDI $\mathcal{H}$. It consists of the following steps:

1. Detect all the inout edges;
2. Generate the set of SPDIs $\mathcal{H}_{red} = \{\mathcal{H}_1, \ldots, H_n\}$;
3. Apply the reachability algorithm for SPDIs to each $\mathcal{H}_i$ $(1 \leq i \leq n)$, $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f)$.
4. If there exists at least one SPDI $\mathcal{H}_i \in \mathcal{H}_{red}$ such that $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$ then $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$, otherwise we do not know.

We have then the following result about termination of GSPDI reachability.

LEMMA 5. $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f)$ *always terminates.*

PROOF. The result follows from the termination of steps 1 and 2 of the above algorithm (based on a finiteness argument), as well as from that of $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f)$ (for all $\mathcal{H}_i \in \mathcal{H}_{red}$, $1 \leq i \leq n$) [5, 7]. □

We finish this section with the main result of our paper, which follows from all the previous results.

THEOREM 6. *Given a GSPDI* $\mathcal{H}$, $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$ *if* $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$ *for some* $\mathcal{H}_i \in \mathcal{H}_{red}$. *On the other hand,* $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f)$ *is inconclusive if for all* $\mathcal{H}_i \in \mathcal{H}_{red}$, $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{No}$.

PROOF. Termination is guaranteed by Lemma 5. Soundness follows from soundness of the algorithm for SPDIs [5, 7], and from that of the steps described in Section 5.2. That reachability is inconclusive whenever $\mathbf{Reach}_{SPDI}(\mathcal{H}_i, \mathbf{x}_0, \mathbf{x}_f) = \mathsf{No}$ for all $\mathcal{H}_i \in \mathcal{H}_{red}$, follows from Proposition 2. □

6. FINAL DISCUSSION

In this paper we presented a terminating algorithm implementing a semi-test for reachability analysis of GSPDIs, based on the decision procedure for SPDIs, and proved its soundness. We first showed that there is no structure-preserving reduction from GSPDI reachability to SPDI reachability by showing that for some trajectories traversing an inout edge in both directions there is no trajectory traversing the edge only in one direction. Since those trajectories are, however, obtained only on very specific cases (Proposition 2), we argue that we miss indeed few positive answers. This would need to be practically corroborated by implementing the algorithm, which remain a future work. Note that the most difficult part is already implemented in the tool SPeeDI [3] and we would only need to implement steps 1 and 2 of the GSPDI algorithm.

Complexity is another issue. In the worst case it is clear that the algorithm introduces an exponential blow-up as it has to generate all possible underlying SPDIs after fixing inout edges as entry-only or exit-only edges. Some qualitative analysis may help here, to guide the reachability analysis as to eliminate beforehand the analysis of some of the types of signatures. This might be done by computing some objects of GSPDI's phase portrait as previously done for SPDIs [6].

We believe this paper positively contributes to the analysis of low-dimensional hybrid systems, given that the GSPDI class lies on the frontier of decidable/undecidable hybrid systems. Moreover, GSPDIs may be used to approximate nonlinear planar differential equations, for which exact solutions are not easy to obtain.

7. REFERENCES

[1] R. Alur, C. Courcoubetis, T. Henzinger, and P.-H. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In *HS'93*, number 736 in LNCS, pages 209–229, 1993.

[2] E. Asarin, O. Maler, and A. Pnueli. Reachability analysis of dynamical systems having piecewise-constant derivatives. *TCS*, 138:35–65, 1995.

[3] E. Asarin, G. Pace, G. Schneider, and S. Yovine. SPeeDI: a verification tool for polygonal hybrid systems. In *CAV'02*, number 2404 in LNCS, pages 354–358, July 2002.

[4] E. Asarin and G. Schneider. Widening the boundary between decidable and undecidable hybrid systems. In *CONCUR'02*, number 2421 in LNCS, pages 193–208, 2002.

[5] E. Asarin, G. Schneider, and S. Yovine. On the decidability of the reachability problem for planar differential inclusions. In *HSCC'01*, number 2034 in LNCS, pages 89–104, 2001.

[6] E. Asarin, G. Schneider, and S. Yovine. Towards computing phase portraits of polygonal differential inclusions. In *HSCC'02*, number 2289 in LNCS, pages 49–61, 2002.

[7] E. Asarin, G. Schneider, and S. Yovine. Algorithmic Analysis of Polygonal Hybrid Systems. Part I: Reachability. *TCS*, 379(1-2):231–265, 2007.

[8] M. W. Hirsch and S. Smale. *Differential Equations, Dynamical Systems and Linear Algebra*. Academic Press Inc., 1974.

[9] V. Mysore and A. Pnueli. Refining the undecidability frontier of hybrid automata. In *FSTTCS*, number 3821 in LNCS, pages 261–272, 2005.

[10] G. Schneider. On the decidability of the reachability problem for GSPDIs. Technical Report 359, Dept. of Informatics, University of Oslo, June 2007.