Relaxing Goodness is Still Good

Gordon J. Pace[1] and Gerardo Schneider[2]

[1] Dept. of Computer Science and AI, University of Malta, Msida, Malta.
[2] Dept. of Informatics, University of Oslo, Oslo, Norway.
{gordon.pace@um.edu.mt; gerardo@ifi.uio.no}

Abstract. Polygonal hybrid systems (SPDIs) are planar hybrid systems, whose dynamics are defined in terms of constant differential inclusions, one for each of a number of polygonal regions partitioning the plane. The reachability problem for SPDIs is known to be decidable, but depends on the *goodness* assumption — which states that the dynamics do not allow a trajectory to both enter and leave a region through the same edge. In this paper we extend the decidability result to *generalised SPDIs* (GSPDI), SPDIs not satisfying the goodness assumption, and give an algorithmic solution to decide reachability of such systems.

1 Introduction

A hybrid system is one in which discrete and continuous behaviours interact. Some systems are inherently hybrid — consider a robot, with differential equations determining its speed, together with an embedded computer taking discrete decisions based on the continuous input values coming from sensors. In other cases, a system consisting only of continuous behaviour, can be *hybridised,* introducing discrete behaviour in order to facilitate the analysis. For example, exact solutions can be difficult to obtain for a non-linear differential equation, making a qualitative and approximative analysis necessary.
A class of hybrid systems for which the reachability question is known to be decidable, are Polygonal Hybrid Systems (SPDIs) — a subclass of hybrid systems on the plane whose dynamics is defined by constant differential inclusions [ASY01,ASY07]. Informally, an SPDI consists of a partition of the plane into polygonal regions, each of which enforces different dynamics given by two vectors determining the possible directions a trajectory might take; a simple SPDI is depicted in Fig. 1-(a). A constructive proof for deciding reachability on SPDIs can be found in [ASY07]. The proof is restricted to SPDIs satisfying the so-called *goodness* assumption — the dynamics of any region of the SPDI do not allow a trajectory to traverse any edge of the polygonal region in opposite directions. An SPDI without the goodness assumption is called a *Generalised SPDI* (GSPDI). Fig. 1-(b) shows an example of a good and a 'bad' region (here 'bad' indicates that the region does not satisfy the goodness criterion). In the figure on the left we can see a good region, where the two vectors $\mathbf{a}$ and $\mathbf{b}$ make it impossible for 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 shows

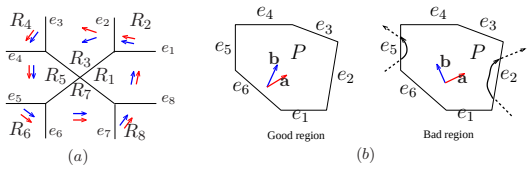

Fig. 1. (a) Example of an SPDI; (b) Good and bad regions.

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

The algorithm presented in [ASY07] for deciding reachability on SPDI depends on pre-processing of trajectory segments and a qualitative analysis to guarantee that it is possible to review the behaviour of all the possible *signatures*[1], by looking at only a finite set of abstract signatures. Informally, this is achieved as follows: (1) Trajectory segments are simplified — it is sufficient to look at trajectories made up of straight segments across regions, and which do not cross themselves; (2) Trajectory segments are abstracted into signatures, based on the *Poincaré map* that relates n-dimensional continuous-time systems with $(n-1)$-dimensional discrete-time systems; (3) It is shown that it is sufficient to look at signatures which consist only of sequences of edges and simple cycles; (4) Such signatures can be abstracted into *types of signatures* — signatures which do not take into account the number of times each simple cycle is iterated.

Many of the lemmas for proving that the above guarantee the finiteness of types of signatures critically depend on the goodness assumption, which propagate this dependency to the constructive proof given for deciding reachability of SPDIs.

The restriction to "good" SPDIs is not justified by applications or inherent interest, it was just a technical condition to facilitate the application of certain techniques and prove decidability. In fact restricting oneself only to SPDIs satisfying the goodness assumption makes it very difficult to model real-life examples. Unfortunately, extending the SPDI model in most ways, such as allowing jumps with resets (from one edge to another remote one), increasing the number of dimensions and allowing non-linear differential inclusions, have been shown to make the model undecidable (see [AS02,MP05] and references therein).

A potentially interesting and useful application of SPDIs is that of the approximation and analysis of two-dimensional non-linear differential equations. By splitting the plane into polygons, and by setting the dynamics of each polygon to be over-approximations of the non-linear differential equation in that region, one can ask reachability questions about the equation, and obtain answers accordingly. When over-approximating the dynamics, a negative reachability answer implies a negative answer in the exact equation. Using more and smaller polygons enables more precise approximations.

The problem with using this approach is that for most differential equations, using a fixed partition breaks the goodness assumption, since some edges will

[1] We call *signature* the sequence of traversed edges by the trajectory. A more formal definition will be given in a later section.

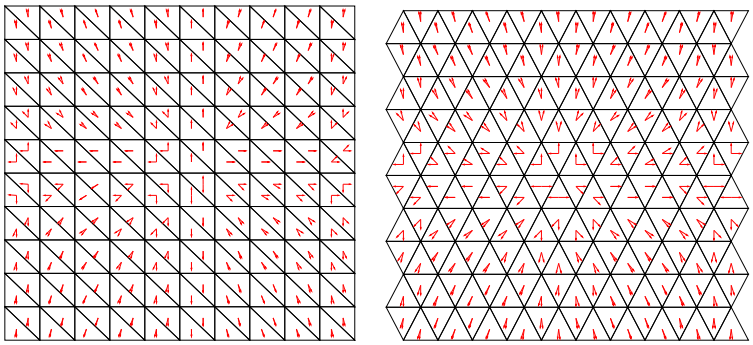

Fig. 2. Approximating a non-linear differential equation using different partitioning of the plane.

lie within the differential inclusion of that region. One solution would be to try to derive an intelligent partition which maintains goodness, but which may be impossible, or by extending the SPDI analysis algorithms to relax the goodness assumption, thus enabling the modelling of non-linear differential equations in a straightforward manner.

As a simple example, consider a pendulum with friction coefficient k, mass M, pendulum length R and gravitational constant g. If θ is the angle subtended with the vertical, the behaviour of the pendulum follows the differential equation: $MR^2\ddot{\theta} + k\dot{\theta} + MgR\sin\theta = 0$. Taking $x = \theta$, and $y = \dot{\theta}$, we get $\dot{x} = y$ and $\dot{y} = -\frac{ky}{MR^2} - \frac{g\sin(x)}{R}$. Using these formulae, SPDIs expressing these constraints can be constructed, possibly with different plane partitions. Fig. 2 gives two such partitions for $k = 1$, $R = 10$, $M = 10$, and $g = -10$. Visual inspection shows that various polygons are not good. By presenting an algorithm to decide GSPDI reachability, we can automatically analyse such systems.

In this paper we present an algorithm for solving the reachability problem for GSPDIs, contributing towards the narrowing of the undecidability frontier of low dimension hybrid systems [AS02,MP05], and enabling the use of GSPDIs to approximate planar non-linear differential equations.

In the next section we outline definitions and results about SPDIs, and then extend them to enable the analysis of GSPDIs in section 3.

2 Polygonal Hybrid Systems (SPDIs)

In this section we recall the main definitions and concepts required in the rest of the paper, and give an outline of the results for SPDIs, upon which the results presented in this paper are built. For a more detailed presentation see [ASY07]. In what follows, we will use $\mathbf{a} = (a_1, a_2)$ and $\mathbf{x} = (x_1, x_2)$ to represent 2-dimensional vectors ($\mathbf{a}, \mathbf{x} \in \mathbb{R}^2$). An *angle* $\angle_{\mathbf{a}}^{\mathbf{b}}$ on the plane, defined by two non-zero vectors $\mathbf{a}$ and $\mathbf{b}$ 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* $\mathcal{H} = \langle \mathbb{P}, \mathbb{F} \rangle$*, where* $\mathbb{P}$ *is a finite partition of the plane (each* $P \in \mathbb{P}$ *being a convex polygon), called the* regions *of the SPDI, and* $\mathbb{F}$ *is a function associating a pair of vectors to each polygon:* $\mathbb{F}(P) = (\mathbf{a}_P, \mathbf{b}_P)$*. 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* $\dot{\mathbf{x}} \in \angle_{\mathbf{a}_P}^{\mathbf{b}_P}$*.* ■

Example 1. Consider the SPDI illustrated in Fig. 1-(a), with eight regions $R_1,\ R_2, \ldots, R_8$. A pair of vectors $(\mathbf{a}_i, \mathbf{b}_i)$ is also associated to each region R_i: $\mathbf{a}_1 = \mathbf{b}_1 = (1, 5)$, $\mathbf{a}_2 = \mathbf{b}_2 = (-1, \frac{1}{2})$, $\mathbf{a}_3 = (-1, \frac{11}{60})$ and $\mathbf{b}_3 = (-1, -\frac{1}{4})$, $\mathbf{a}_4 = \mathbf{b}_4 = (-1, -1)$, $\mathbf{a}_5 = \mathbf{b}_5 = (0, -1)$, $\mathbf{a}_6 = \mathbf{b}_6 = (1, -1)$, $\mathbf{a}_7 = \mathbf{b}_7 = (1, 0)$, $\mathbf{a}_8 = \mathbf{b}_8 = (1, 1)$. ■

We define $E(P)$ to be the set of edges of region P. We say that an edge e ($e \in E(P)$) is an *entry-only* of P if for all $\mathbf{x} \in e$ and for all $\mathbf{c} \in \angle_{\mathbf{a}_P}^{\mathbf{b}_P}$, $\mathbf{x} + \mathbf{c}\epsilon \in P$ for some $\epsilon > 0$. We say that e is an *exit-only* of P if the same condition holds for some $\epsilon < 0$. Intuitively, an entry-only (exit-only) edge of a region P allows at least a trajectory in P starting (terminating) on edge e, but allows no trajectories in P terminating (starting) on edge e. We write $\mathit{In}(P)$ ($\mathit{In}(P) \subseteq E(P)$) to denote the set of all entry-only edges of P and $\mathit{Out}(P)$($\mathit{Out}(P) \subseteq E(P)$) to denote the set of exit-only edges of P. From the definition, it follows immediately that no edge can be both an entry-only and an exit-only edge of a region: $\mathit{In}(P) \cap \mathit{Out}(P) = \emptyset$. A region P is said to be *good*, if all the edges of that region are either entry-only or exit-only: $E(P) = \mathit{In}(P) \cup \mathit{Out}(P)$. An SPDI is said to be *good*, or satisfy the *goodness* assumption, if it consists of only good regions: $\forall P \in \mathbb{P} \cdot E(P) = \mathit{In}(P) \cup \mathit{Out}(P)$.

Example 2. In Fig. 1-(b), the region P shown on the left is good since all edges are either entry-only or exit-only. The region depicted on the right shows a region that is not good, since neither edge e_2 nor edge e_5 are in $\mathit{In}(P) \cup \mathit{Out}(P)$. ■

Note that though the definition of SPDIs does not exclude the possibility of having dynamics with $\mathbf{a}$ and $\mathbf{b}$ co-linear (i.e., $\mathbf{a} = -\mathbf{b}$), this is excluded under the goodness assumption. In what follows, 'SPDI' will always denote a good SPDI, unless specified otherwise.
We will use the notation $e_{\circlearrowright}^P$ to indicate the directed edge e such that it follows a clockwise direction around region P, and similarly $e_{\circlearrowleft}^P$ to indicate the directed edge e following an anti-clockwise direction around region P. Given a directed edge e, its inverse will be written as e^{-1}.

Definition 2. The set of directed edges of an SPDI $\mathcal{H}$ with partition $\mathbb{P}$, written $E_d(\mathcal{H})$, is defined to be: $E_d(\mathcal{H}) = \{e_{\circlearrowright}^P \mid P \in \mathbb{P},\ e \in \mathit{In}(P)\} \cup \{e_{\circlearrowleft}^P \mid P \in \mathbb{P},\ e \in \mathit{Out}(P)\}$. Similarly, we define $\mathit{In}_d(P)$ and $\mathit{Out}_d(P)$ to correspond to $\mathit{In}(P)$ and $\mathit{Out}(P)$ but with directed edges. ■

Since an edge appears in two adjacent regions, the direction induced in the two regions may be different. However, it is easy to see that edges which are entry-only in one region, and exit-only in the other, result in matching induced directions: $e \in E_d(\mathcal{H})$ or $e^{-1} \in E_d(\mathcal{H})$, but not both [ASY01,MP93]. In an SPDI

the only case where one can have both e and e^{-1} in a signature is when e is an entry-only (or exit-only) edge in both adjacent regions it belongs to.
A *trajectory segment* of an SPDI $\mathcal{H}$, is a continuous and almost-everywhere (everywhere except on finitely many points) differentiable function $\xi \in [0,T] \to \mathbb{R}^2$ such that for all $t \in [0,T]$, if $\xi(t) \in P$ and $\dot{\xi}(t)$ is defined then $\dot{\xi}(t) \in \angle_{\mathbf{a}_P}^{\mathbf{b}_P}$.
The *signature* of a trajectory segment ξ, written $\mathsf{Sig}(\xi)$, is the sequence of edges traversed by the trajectory, that is, $e_1, e_2, \ldots e_n$ resulting from $\xi \cap E_d(\mathcal{H})$, where edges are arranged in the order they are "visited" by ξ.
One important result is that the behaviour of any trajectory is equivalent, with respect to reachability, to the behaviour of some trajectory which does not cross itself and follows straight-line segments within regions.

Lemma 1 ([ASY07]). *Given a trajectory segment $\xi \in [0,T] \to \mathbb{R}^2$, there exists another trajectory segment $\xi' \in [0,T'] \to \mathbb{R}^2$ starting and finishing at the same points as ξ ($\xi(0) = \xi'(0)$ and $\xi(T) = \xi'(T')$) such that (i) ξ' does not cross itself (ξ is injective); and (ii) ξ' follows straight-line segments inside regions.* □

Though in general the reachability problem for an SPDI $\mathcal{H}$ may be considered for region-to-region, for simplicity of presentation we define it as the following predicate: $\mathit{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f) \equiv \exists \xi \, \exists t \geq 0 \; . \; (\xi(0) = \mathbf{x}_0 \wedge \xi(t) = \mathbf{x}_f)$. lemma 1 shows that to decide reachability, it is sufficient to look at non-self-crossing trajectories consisting of straight-line segments. In the rest of the discussion, we will restrict our use of trajectory to mean 'a non-self-crossing trajectory composed of straight-line segments between edges'. Similarly, the term signature will be used to indicate the signature of a trajectory with these constraints.
As usual in reachability analysis we need to compute successors, which are built upon special kind of multi-valued functions introduced in what follows.

Truncated Affine Multi-Valued Functions. An *affine* function $f \in \mathbb{R} \to \mathbb{R}$ is such that $f(x) = ax + b$. If $a > 0$ we say that f is *positive affine*, and if $a < 0$ we say that f is *negative affine*; we call this the parity of the affine function.
An *affine multivalued* function (AMF) $F \in \mathbb{R} \to 2^{\mathbb{R}}$, written $F = \langle f_l, f_u \rangle$, is defined by $F(x) = \langle f_l(x), f_u(x) \rangle$ where f_l and f_u are positive affine and $\langle \cdot, \cdot \rangle$ 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. For an interval $I = \langle l, u \rangle$ we have that $F(\langle l, u \rangle) = \langle f_l(l), f_u(u) \rangle$.
An *inverted affine multivalued* function $F \in \mathbb{R} \to 2^{\mathbb{R}}$, is defined by $F(x) = \langle f_u(x), f_l(x) \rangle$ where f_l and f_u are both negative affine and $\langle \cdot, \cdot \rangle$ denotes an interval.
Given an AMF F and two intervals $S \subseteq \mathbb{R}^+$ and $J \subseteq \mathbb{R}^+$, a *truncated affine multivalued* function (TAMF) $\mathcal{F}_{F,S,J} \in \mathbb{R} \to 2^{\mathbb{R}}$ is defined as follows: $\mathcal{F}_{F,S,J}(x) = F(x) \cap J$ if $x \in S$, otherwise $\mathcal{F}_{F,S,J}(x) = \emptyset$. In what follows we will write $\mathcal{F}$ instead of $\mathcal{F}_{F,S,J}$ whenever no confusion may arise. Moreover, in the rest of the paper F will always denote an AMF and $\mathcal{F}$ a TAMF. For convenience we write $\mathcal{F}(x) = F(\{x\} \cap S) \cap J$ instead of $\mathcal{F}(x) = F(x) \cap J$ if $x \in S$. We overload the application of a TAMF on an interval I: $\mathcal{F}(I) = F(I \cap S) \cap J$. We say that $\mathcal{F}$ is *normalised* if $S = \mathsf{Dom}(\mathcal{F}) = \{x \mid F(x) \cap J \neq \emptyset\}$ and $J = \mathsf{Im}(\mathcal{F}) = \mathcal{F}(S)$.

As in the case of affine multivalued functions, an *inverted truncated affine multivalued* function (inverted TAMF) is similar to a TAMF, but defined in terms of an inverted affine multivalued function as opposed to a normal one. An important result is that normal TAMFs are closed under composition.

Theorem 1 ([ASY07]). The functional composition of two normal TAMFs $\mathcal{F}_1(I) = F_1(I \cap S_1) \cap J_1$ and $\mathcal{F}_2(I) = F_2(I \cap S_2) \cap J_2$, is the TAMF $(\mathcal{F}_2 \circ \mathcal{F}_1)(I) = \mathcal{F}(I) = F(I \cap S) \cap J$, where $F = F_2 \circ F_1$, $S = S_1 \cap F_1^{-1}(J_1 \cap S_2)$ and $J = J_2 \cap F_2(J_1 \cap S_2)$. □

The following new corollary extends the above result.

Corollary 1. *The composition of two inverted TAMFs gives a normal TAMF. Conversely, the composition of one normal and one inverted TAMF (in either order) gives an inverted TAMF.* □

To avoid having to reason about the length of every edge, we normalise every edge e such that its TAMF has the domain $[0, 1]$ (that is, the normalised version of e has length 1, with 0 corresponding to the starting point of the directed edge, and 1 to the end point).

Successors 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 directed edge and its one-dimensional representation. Accordingly, we write $\mathbf{x} \in e$ and $x \in e$, to mean "point $\mathbf{x}$ lies on edge e" and "coordinate x in the one-dimensional coordinate system of e" respectively. The same convention applied to sets of points of e represented as intervals (for example, $\mathbf{x} \in I$ and $x \in I$, where $I \subseteq e$) and to trajectories (for example, "ξ starting at x" or "ξ starting at $\mathbf{x}$").
Consider a polygon $P \in \mathbb{P}$, with $e_0 \in In_d(P)$ and $e_1 \in Out_d(P)$. For $I \subseteq e_0$, $\mathsf{Succ}_{e_0e_1}(I)$ is defined to be the set of all points lying on e_1 reachable from some point in I by a trajectory segment $\xi \in [0, t] \to \mathbb{R}^2$ in P (that is, $\xi(0) \in I \wedge \xi(t) \in e_1 \wedge \mathsf{Sig}(\xi) = e_0e_1$). Given $I = [l, u]$, $\mathsf{Succ}_{e_0e_1}(I) = F(I \cap S_{e_0e_1}) \cap J_{e_0e_1}$, where $S_{e_0e_1}$ and $J_{e_0e_1}$ are intervals, $F([l, u]) = \langle f_l(l), f_u(u)\rangle$ and f_l and f_u are positive affine functions. Successors are thus normal TAMFs.

Qualitative analysis of simple edge-cycles In what follows a sequence of edges in parenthesis, $\sigma = (e_1 \dots e_k)$, will denote a simple edge-cycle – that is, a signature that can be repeated at least once, and such that all edges are distinct ($e_i \neq e_j$ for all $1 \leq i < j \leq k$). Given an SPDI, its topology determines when a sequence of edges may form a simple cycle [ASY07]. Let $\mathsf{Succ}_\sigma(I) = F(I \cap S_\sigma) \cap J_\sigma$ with $F = \langle f_l, f_u\rangle$, and S_σ and J_σ computed as in theorem 1.
We assume that neither of the two functions f_l, f_u is the identity function. The following analysis, taken from [ASY01], allows us to calculate the behaviour of cycles provided that the path along the cycle has a normal (not inverted) TAMF. Since, in SPDIs, the TAMF between a pair of edges is normal, and the

composition of two normal TAMFs is itself a normal TAMF, this approach is universally applicable as long as the goodness assumption holds.
Let σ be a simple cycle, and l^* and u^* be the fix-points[2] of f_l and f_u, respectively, and $S_\sigma \cap J_\sigma = \langle L, U \rangle$. It can be shown that σ is 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 behaviour of trajectories. Any trajectory that enters a cycle of kind DIE will eventually leave it after a finite number of turns. In a cycle of kind 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. Also note that the above analysis gives us a non-iterative solution of cycle behaviour for most cycles; the theoretical algorithm [ASY07] as well as the tool SPeeDI [Spe] uses such acceleration techniques. An important result to prove the decidability of SPDIs is that any valid signature can be expressed in a normal form, consisting of alternating sequential paths and simple cycles:

Theorem 2 ([ASY07]). *Given an SPDI with the goodness assumption, any edge signature $\sigma = e_1 \dots e_p$ can be written as $\sigma_{\mathcal{A}} = r_1 s_1^{k_1} \dots r_n s_n^{k_n} r_{n+1}$, 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 (no edges are repeated within s_i).* □

Let $\sigma = e_1 \dots e_p$ be an edge signature and $\sigma_{\mathcal{A}} = r_1 s_1^{k_1} \dots r_n s_n^{k_n} r_{n+1}$ be its representation as in the above theorem. Then we define the *type of a signature* σ as $\mathtt{type}(\sigma) = r_1, s_1, \dots, r_n, s_n, r_{n+1}$. We say that a signature σ is *feasible* if and only if there exists a trajectory segment ξ with signature σ, i.e., $\mathsf{Sig}(\xi) = \sigma$. Types of signatures have the following properties:

Lemma 2 ([ASY07]). *Given an SPDI, let $\sigma = e_0 \dots e_p$ be a feasible signature, then its type, $\mathtt{type}(\sigma) = r_1, s_1, \dots, r_n, s_n, r_{n+1}$ satisfies the following properties: (i) every $1 \leq i < j \leq n+1$, r_i and r_j are disjoint; (ii) every $1 \leq i < j \leq n$, s_i and s_j are different.* □

The finiteness of types of signatures is the basis of the proof of decidability of (good) SPDI reachability, and of the termination of the reachability algorithm (together with acceleration results for simple cycles).

[2] The fix-point x^* is the solution of $f(x^*) = x^*$, where $f(\cdot)$ is positive affine. The existence and computation of such fix-points are detailed in [ASY07].

Theorem 3 ([ASY07]). *Point-to-point, interval-to-interval and region-to-region reachability for SPDIs is decidable.* □

3 Relaxing Goodness: Generalised SPDIs

The original proof of the decidability of the reachability question for SPDIs, depended on the concept of monotonicity of TAMFs and their composition. Before starting the analysis, the algorithm fixed the direction of the edges separating regions. An interesting result guaranteed that the orientation of the edges resulted in each polygon split into two contiguous sequences of edges — one being entry-only edges, the other being exit-only edges. Furthermore, the orientation of an edge in one region is guaranteed to match the orientation of the same edge in the adjacent region[3], as shown in Fig. 3-(a). When one moves on to GSPDIs, *inout* edges (those that may be traversed in both directions) break this result, since the direction of an edge when considered as an input edge clashes with the direction it is given when used as an output edge in the same region. The previous result however, still guaranteed that the entry-only edges and the exit-only edges can be assigned in one fixed direction (see Fig. 3-(b)).
To solve this problem, we use directed edges, and differentiate between the edge used as an input, and when it is used as an output, just as though they were two different edges in the GSPDI. Fig. 3-(c) shows how an inout edge can be seen in this manner. Note that depending on in which direction the trajectory traverse the inout edge e_1, it is an input edge in region R_1, but an output edge in region R_2, and similarly, e_1^{-1} is an output edge in region R_1 and an input edge in region R_2; that is why we did not draw the direction vector in the picture. In other words, any path passing through the edge such as $\sigma = e_0e_1e_2 \dots e_3e_1^{-1}e_4$ (see Fig. 3-(d)) can be analysed as before, and through monotonicity, one can deduce that Succ_σ is a positive TAMF. e_1 and e_1^{-1} are considered distinct edges, and the above path contains no cycle.
It can be seen that the standard analysis for SPDIs works well for such cases. However, paths can now 'bounce' off an edge. Recall that any pair of edges e_0e_1 is part of a path if e_0 is an input edge of a region, and e_1 is an output edge of the same region. One can calculate the TAMF for such a trajectory. However, ee^{-1} can now be a valid path, whose behaviour cannot be expressed as a normal TAMF. This breaks the analysis used in SPDIs, to accelerate the analysis of simple cycles. The standard SPDI analysis thus needs to be extended to handle such 'bounces' in paths.

3.1 Preliminary Results

The *goodness* assumption was originally introduced to simplify treatment of trajectories and to guarantee that each region can be partitioned into *entry-only*

[3] There are special cases when an edge is an entry-only to a region and an exit-only to an adjacent region. From the reachability point of view this does not cause any problem as these cases can be identified and treated accordingly.

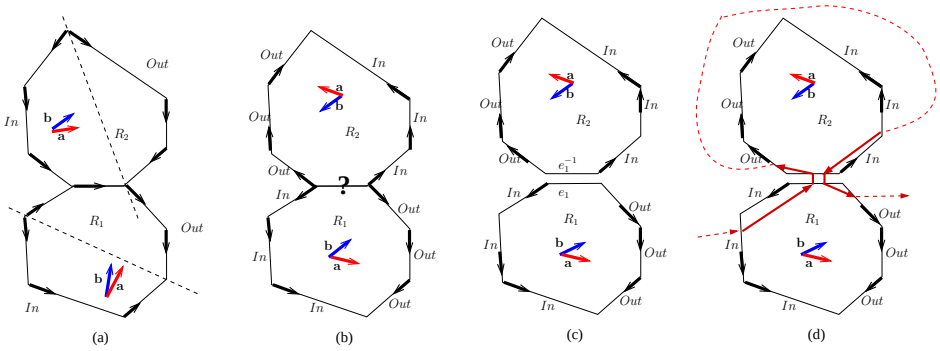

Fig. 3. (a) An SPDI with matching order of edges; (b) a GSPDI showing that the order breaks the contiguity of the edge directions; (c) a GSPDI with a duplicated inout edge; (d) a path through the GSPDI using edge e_1 in both directions.

and *exit-only* edges in an ordered way, a fact used in the proof of decidability of the reachability problem. In this section, we will introduce further background, and provide new results concerning GSPDIs, needed to prove our decidability result.

Definition 3. *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.* ■

An SPDI without the goodness assumption is called a *Generalised SPDI (GSPDI)*. Thus, in GSPDIs there are three kinds of edges: inouts, entry-only and exit-only. Self-crossing of trajectory (segments) of SPDIs can be eliminated, which allows us to consider only non-crossing trajectory (segments). Standard algebraic manipulation of vector suffices to show that lemma 1 [ASY07]) also applies to GSPDIs. Therefore, in what follows, we will consider only trajectory segments without self-crossings. Note that on GSPDIs, a trajectory can "intersect" an edge at an infinite number of points by *sliding* along it. A trace is thus no longer a sequence of points, but rather, a sequence of intervals.

Definition 4. *The* trace *of a trajectory ξ is the sequence* $\mathtt{trace}(\xi) = I_0 I_1 \dots I_n$ *of the intersection intervals of ξ with the set of edges: $I_i \subseteq \xi \cap E_d(\mathcal{H})$.* ■

In Fig. 4-(a) we show a trajectory segment ξ, such that $\mathtt{trace}(\xi) = I_0 I_1 I_2 \dots I_3 I_4 I_5$ where I_1, I_2, I_3, I_4 and I_5 are points.

Definition 5. *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 sequence of traversed edges by the trajectory segment, that is,* $\mathsf{Sig}(\xi) = e_0 e_1 \dots e_n$, *with* $\mathtt{trace}(\xi) = I_0 I_1 \dots I_n$ *and* $I_i \subseteq e_i$. ■

In Fig. 4-(a) the signature of the trajectory segment ξ is $\mathsf{Sig}(\xi) = e e' e'' \dots e''^{-1} e'^{-1} e'''$ (to simplify the picture we do not draw the "duplicated" edges e''^{-1} and e'^{-1}).

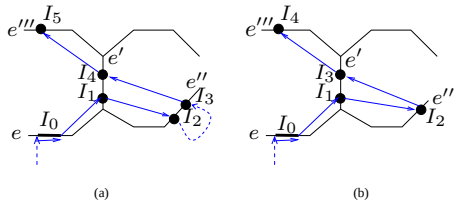

Fig. 4. (a) A sliding trajectory with a proper inout edge; (b) A sliding trajectory with a proper inout edge and a bounce.

Note that, in many cases, the intervals of a trace are in fact points. We say that a trajectory with edge signature $\mathsf{Sig}(\xi) = e_0 e_1 \dots e_n$ and trace $\mathtt{trace}(\xi) = I_0 I_1 \dots I_n$ *interval-crosses* edge e_i if I_i is not a point. Given a trajectory segment, we will distinguish between *proper inout* edges and *sliding* edges.

Definition 6. *Let ξ be a trajectory segment from point $\mathbf{x}_0 \in e_0$ to $\mathbf{x}_f \in e_f$, with edge signature $\mathsf{Sig}(\xi) = e_0 \dots e_i \dots e_n$, and $e_i \in E(P)$ be an 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* along an edge e, if e is a sliding edge of P for ξ, and that ξ is a *sliding trajectory* if it contains at least one sliding edge. Fig. 4-(a) shows a sliding trajectory, where e is a sliding edge while e' and e'' are proper inout edges. The following is a useful property of inout edges.

Proposition 1. *If e is an inout edge, then any trajectory reaching the edge can always slide on the edge (in one or the other direction, or both).* □

Since inout edges may appear in different situations, we need to explain our strategy to deal with them, for which we need some preliminaries. For a region P let us say that a vector $\mathbf{c}$ is *compatible* with the orientation of P if $\mathbf{c}$ is a positive combination of vectors $\mathbf{a}$ and $\mathbf{b}$ associated with P.
Let e be an edge separating two polygons P_1 and P_2. There are the following four new cases in addition to those from [ASY07]: (1) e is an inout edge in P_1 and an entry-only edge in P_2; (2) e is an inout edge in P_1 and an exit-only edge in P_2; (3) e is an inout edge in P_1 and also an inout edge in P_2[4]; (4) e is an inout edge in P_1 and also an inout edge in P_2 like in the previous case[5]. In all these cases a trajectory may slide on the edge if at least one of the dynamics on the region allows it. For instance, in case (4), the trajectory may slide in both directions. The above cases are included in our reachability analysis.

[4] If $\mathbf{c}_1$ is a vector compatible with the orientation of P_1 and $\mathbf{c}_2$ a vector compatible with the orientation of P_2 (see definition above) such that $\mathbf{c}_1$ and $\mathbf{c}_2$ are parallel to edge e then both vectors are oriented in the same direction.

[5] In this case if $\mathbf{c}_1$ is a vector compatible with the orientation of P_1 and $\mathbf{c}_2$ a vector compatible with the orientation of P_2 such that both $\mathbf{c}_1$ and $\mathbf{c}_2$ are parallel to edge e then $\mathbf{c}_1$ and $\mathbf{c}_2$ are oriented in the opposite directions.

As for SPDIs, we have the following property of Succ: for any edge signatures σ_1 and σ_2 and edge e: $\mathsf{Succ}_{e\sigma_1} \circ \mathsf{Succ}_{\sigma_2 e} = \mathsf{Succ}_{\sigma_2 e \sigma_1}$.
The following lemma shows that the edge-to-edge successor function is a normal TAMF whenever the two edges are not the inverse of each other. It follows directly from the similar result for SPDIs [ASY07], which makes no assumption regarding goodness.[6]

Lemma 3. *For any two edges e_0 and e_1, $\mathsf{Succ}_{e_0 e_1}$ is always a normal TAMF, whenever $e_1 \neq e_0^{-1}$.* □

Besides sliding, the signatures that we will be analysing in GSPDIs may include subsequences of the form ee^{-1}. The behaviour between such edges does not correspond to a normal TAMF, and thus has to be analysed separately. A *bounce* is a part of a trajectory which crosses an edge twice in immediate succession. More formally:

Definition 7. *Given a signature $\sigma = e_0 e_1 \dots e_n$, a pair of edges $e_i e_{i+1}$ is said to be a* bounce *if $e_{i+1} = e_i^{-1}$. We say that a signature $e_0 e_1 \dots e_n$ contains m bounces, if there are exactly m distinct indices $I = \{i_1,\ i_2,\ \dots i_m\}$ such for every $i \in I$, $e_i = e_{i+1}^{-1}$.* ■

Fig. 4-(b) shows a sliding trajectory $\mathsf{Sig}(\xi) = ee'e''e''^{-1}e'^{-1}e'''$. There is only one bounce, namely $e''e''^{-1}$.
Let $\mathsf{Flip}[l, u] = [1 - u,\ 1 - l]$ be an interval function. The following result establishes that the successor function for bounces can be defined in terms of the Flip function. It is easy to prove that $\mathsf{Succ}_{ee^{-1}} = \mathsf{Flip}$.
One of the useful properties of SPDIs is that the successor function of any given signature is a normal TAMF. For GSPDIs, however, we need to take into account bounces, and hence analyse the composition of normal TAMFs with Flip:

Lemma 4. *Composing Flip with an inverted TAMF gives a normal TAMF and an inverted TAMF if we compose it with a normal TAMF.* □

The parity of the number of bounces occurring in a given signature influences the form of the underlying TAMF, as shown in the following result, whose proof follows immediately by induction on the number of bounces.

Corollary 2. *Any signature with an even number of bounces has its behaviour characterised by a normal TAMF, while a signature with an odd number of bounces is characterised by an inverted TAMF.* □

Recall that the analysis of simple cycle behaviour given for SPDIs depends only on the assumption that the TAMF of the cycle body is a normal one. From the previous result, it thus follows that whenever the number of bounces is even on a given cyclic signature, the composed TAMF is a normal one, so the analysis of simple cycles can be conducted as for SPDIs:

[6] Note that the underlying AMFs are not necessarily positive affine whenever applied to an inout edge, since the leftmost (rightmost) function may give $-\infty$ ($+\infty$). However, this is not a problem for successors, as in this case the underlying TAMFs are of the form $[0, ax + b]$ or $[ax + b, 1]$ due to sliding.

Lemma 5. *Given a simple cycle σ containing an even number of bounces, its iterated behaviour can be calculated as for SPDIs.* □

Since a trajectory slides only along inout edges, and can only bounce off inout edges, we can prove that simple cycles which include at least one bounce are never STAY cycles. This gives us the advantage of the use of acceleration techniques already used for SPDIs.

Lemma 6. *Simple cycles which include bounces are not STAY cycles.* □

From lemma 5 we have that only simple cycles with an odd number of bounces need to be analysed. Considering the case when a bounce appears as the first pair of elements of a simple cycle body, we can accelerate the analysis by running through the simple cycle only once. The proof follows from the fact that the initial bounce enables a slide, thus allowing us to identify the limits through only one application of the simple cycle body:

Lemma 7. *Given a signature $\sigma = e_0(e_1e_1^{-1}e_2 \dots e_n)^k e_1$ (i) with only one simple cycle; (ii) with $k > 0$; (iii) with an odd number of bounces; and (iv) starts with a bounce; its behaviour is equivalent to following the simple cycle only once $\sigma' = e_0e_1e_1^{-1}e_2 \dots e_ne_1$. In other words: $\mathsf{Succ}_\sigma = \mathsf{Succ}_{\sigma'}$.* □

Based on the above lemma, we can prove that any simple cycle containing an odd number of bounces can be accelerated. The proof works by unwinding the simple cycle body to push the first bounce to the beginning, and then applying the previous lemma:

Lemma 8. *Given a simple cycle s with an odd number of bounces, we can calculate the limit of its iterated behaviour without iterating.* □

Therefore, we can now analyse any type of signature in GSPDIs using the results from lemma 3 (to deal with inout edges), and lemmas 5 and 8 (to deal with bounces). Given a simple cycle s, let s^+ be the cycle iterated one or more times.

Theorem 4. *We can (constructively) compute the behaviour of a signature $r_1s_1^+ r_2s_2^+ \dots r_n$.* □

3.2 Decidability Results

The following lemma guarantees that it is sufficient to consider simple cycles which occur in a type of signature with certain patterns. Any type of signature containing two occurrences of the same simple cycle can be reduced to another type of signature where the simple cycle s occurs only once, provided the cycle with the edges in reverse order does not occur between them. The proof is based on the fact that, assuming the trajectory does not cross itself, between two instances of a repeated simple cycle, one can always find either the reverse of the cycle or a bounce, in which case, the bounce can be eliminated to avoid leaving the simple cycle.

Lemma 9. *Given a GSPDI, and assuming only trajectories without self-crossing, if there is a type of signature where a simple cycle $s = (e_0, e_1, \ldots, e_n)$ appears twice, i.e.* $\mathtt{type}(\mathsf{Sig}(\xi)) = \sigma'\sigma''\sigma'''$ *with* $\sigma'' = s^k \ldots s^{k''}$*, then if there is no reverse*(s) *between the two occurrences of* s*, then* $\mathtt{type}(\mathsf{Sig}(\xi)) = \sigma' s^{k'''} \sigma'''$*.* □

We also prove that a trajectory which takes a simple cycle (any number of times), then takes it again (any number of times) but in reverse order, and finally takes it a number of times in the forward direction, can be simulated by another trajectory which simply takes the simple cycle a number of times. The proof is based on the fact that whichever direction the first edge of the simple cycle under consideration allows sliding in, it is possible to obtain a type of signature preserving reachability without such a pattern.

Lemma 10. *Given a GSPDI, if there is a trajectory segment* $\xi : [0, T] \to \mathbb{R}^2$*, with* $\xi(0) = \mathbf{x}$ *and* $\xi(t) = \mathbf{x}'$ *for some* $t > 0$*, such that* $\mathtt{type}(\mathsf{Sig}(\xi)) = r_1 s_1^{k_1} r_2 s_2^{k_2} r_3 s_3^{k_3} r_4$*, with* $s_2 = s_1^{-1}$ *and* $s_3 = s_1$*, then it is always possible to find a trajectory segment* $\xi' : [0, T] \to \mathbb{R}^2$ *such that* $\xi'(0) = \mathbf{x}$ *and* $\xi'(t) = \mathbf{x}'$ *for some* $t > 0$*, and* $\mathtt{type}(\mathsf{Sig}(\xi)) = r_1 s_1^{k'_1} r'_4$*.* □

Based on the above, we can conclude that for GSPDIs we can always transform a type of signature into one where simple cycles are not repeated.

Corollary 3. *Given a GSPDI, an edge signature* σ *can 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* $1 \leq i \leq n+1$*,* s_i *is a simple cycle (no repetition of edges), and for every* $1 \leq i < j \leq n$*,* s_i *and* s_j *are different.* □

We can define the notion of *type of signature* as for SPDIs, abstracting the number of times simple cycles are iterated on signatures of the kind shown on the above corollary. Note that the statement of corollary 3 is weaker than the corresponding result for SPDIs (theorem 2 and lemma 2) since it does not have any restriction on the sequences of edges r_i. However, the result is enough to prove that there is only a finite number of types of signatures for a given GSPDI.

Corollary 4. *A GSPDI has finitely many different types of signatures.* □

Given a type of signature σ where each edge is traversed in exactly one direction, let $\mathbf{Reach}_{\sigma}(\mathbf{x}_0, \mathbf{x}_f)$ be the SPDI reachability algorithm from [ASY07]. The reachability algorithm for a GSPDI $\mathcal{H}$, $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f)$, works as follows: (1) Generate the finite set of types of signatures $\Sigma = \{\sigma_0, \ldots, \sigma_n\}$ (taking into account e and e^{-1} as different edges), and such that the simple cycles are all distinct; (2) Apply $\mathbf{Reach}_{\sigma_i}(\mathbf{x}_0, \mathbf{x}_f)$ for each $\sigma_i \in \Sigma$; (3) Answer Yes if and only if for some $\sigma_i \in \Sigma$, $\mathbf{Reach}_{\sigma_i}(\mathbf{x}_0, \mathbf{x}_f) = \mathsf{Yes}$.
In step 2 we apply Succ progressively on the abstract signature, using lemmas 5 and 8 to compute the successor of a simple cycle with bounces, and the Succ function as in the case of SPDIs for the rest. Based on these results, it is possible to show termination, correctness and completeness of GSPDI reachability. From this, the main theoretical result follows immediately:

Theorem 5. $\mathbf{Reach}(\mathcal{H}, \mathbf{x}_0, \mathbf{x}_f)$ *is a sound and complete algorithm calculating GSPDI reachability. The reachability problem for GSPDIs is decidable.* □

4 Final Remarks

We have proved that the reachability question for GSPDIs is decidable. The proof is constructive, extending the algorithm for SPDIs [ASY07]. The key lies in showing that the previous analysis works in all cases except when a simple cycle contains an odd number of bounces. The algorithm is extended to deal with such cases, considering now inout edges which enable sliding, but the overall effect is to accelerate the analysis of an SPDI, since at least one end of the edge is immediately covered once the edge is reached.

Concerning complexity, the algorithm presented here has the same worst-case space complexity as for SPDIs, with the only extra additional drawback of eventually doubling the number of edges due to duplication of inout edges. Concerning time complexity, the reachability algorithm developed for SPDIs makes massive use of acceleration techniques, reducing the practical complexity of the analysis. For GSPDIs acceleration is used even in more cases: every simple cycle containing an inout edge can be accelerated. Overall, if compared with SPDI reachability analysis, we have a slight increase on the size of the search state-space, but a faster way of analysing simple cycles. Furthermore, we conjecture that the techniques presented in [PS06b] for reducing the search state-space as well as the compositional analysis introduced in [PS06a] for SPDIs could be applied without further development for GSPDIs.

The main contribution of our paper is interesting in a theoretical sense since it extends the class of decidable hybrid systems, narrowing further the gap between what is known to be be decidable and what is known to be undecidable [AS02,MP05]. The result is also interesting in a practical sense, since it provides a good foundation to approximate planar non-linear differential equations, complementing other works using piecewise linear hybrid systems.

Reachability analysis of GSPDIs is not easy. An early (unsuccessful) attempt to prove decidability of GSPDI reachability was presented in [Sch08] — in which it is shown that no structure-preserving reduction of GPSDI reachability into SPDI reachability is possible. Instead, a semi-test algorithm, which reduces reachability of GSPDI into reachability of an exponential number of SPDIs was developed. The main idea behind this algorithm is that in most cases reachability is preserved when fixing inout edges as entry-only or exit-only edges, and considering all possible permutations of SPDIs generated from this pre-processing, reducing then the problem to SPDI reachability. However, there are cases where it is not possible to eliminate inout edges while preserving reachability. Moreover, the proposed algorithm introduces an extra exponential blow-up to the analysis. The decidable algorithm presented in the present paper for GSPDIs follows a completely different approach than the semi-test presented in [Sch08].

Reachability analysis over SPDIs converges in various cases in which semi-algorithms for general n-dimensional hybrid systems diverge (eg. see [APSY02] for a comparative analysis with HyTech [HHW95]) This extends for GSPDI analysis. Decidability of low-dimensional hybrid systems is addressed in [AS02,MP05]. In particular, in [AS02] it was shown that by slightly modifying PCDs to obtain 2-dimensional linear hybrid automata, the reachability problem becomes

undecidable, showing that GSPDIs really lies on the edge of decidability. The relation between GSPDIs and rectangular hybrid automata [HM00] restricted to 2-dimensional systems, is that not all GSPDIs can be reduced into a rectangular automaton, but on the other hand, no resets are allowed in GSPDIs — making them incomparable.
Multi-affine functions have also been used in [KB06], in which the reachability problem is translated into an abstract discrete system resulting in an over-approximation. The notion of trace and edge signatures has also been used in [BMRT04] to build a bisimulation relation for o-minimal hybrid systems [LPS00] — GSPDIs are not o-minimal systems since the flow is non-deterministic.
Further comparison of other work with GSPDIs can be induced from their comparison to SPDIs [ASY07]. A full version of this paper can be found in [PS08].

References

[APSY02] E. Asarin, G. Pace, G. Schneider, and S. Yovine. SPeeDI: a verification tool for polygonal hybrid systems. In *CAV'02*, LNCS 2404, 2002.

[AS02] E. Asarin and G. Schneider. Widening the boundary between decidable and undecidable hybrid systems. In *CONCUR'02*, LNCS 2421, 2002.

[ASY01] E. Asarin, G. Schneider, and S. Yovine. On the decidability of the reachability problem for planar differential inclusions. In *HSCC'01*, LNCS 2034, 2001.

[ASY07] E. Asarin, G. Schneider, and S. Yovine. Algorithmic Analysis of Polygonal Hybrid Systems. Part I: Reachability. *TCS*, 379(1-2):231–265, 2007.

[BMRT04] T. Brihaye, C. Michaux, C. Rivière, and C. Troestler. On o-minimal hybrid systems. In *HSCC*, volume 2993 of *LNCS*, pages 219–233, 2004.

[HHW95] T.A. Henzinger, P-H. Ho, and H. Wong-Toi. Hytech: The next generation. In *Proc. IEEE Real-Time Systems Symposium RTSS'95*, 1995.

[HM00] T. A. Henzinger and R. Majumdar. Symbolic model checking for rectangular hybrid systems. In *TACAS*, LNCS 1785, pages 142–156, 2000.

[KB06] M. Kloetzer and C. Belta. Reachability analysis of multi-affine systems. In *HSCC*, LNCS 3927, pages 348–362, 2006.

[LPS00] G. Lafferriere, G.J. Pappas, and S. Sastry. O–Minimal hybrid systems. *Mathematics of control, signals and systems*, 13:1–21, 2000.

[MP93] O. Maler and A. Pnueli. Reachability analysis of planar multi-linear systems. In *CAV'93*, LNCS 687, 1993.

[MP05] V. Mysore and A. Pnueli. Refining the undecidability frontier of hybrid automata. In *FSTTCS*, LNCS 3821, 2005.

[PS06a] G. J. Pace and G. Schneider. A compositional algorithm for parallel model checking of polygonal hybrid systems. In *ICTAC'06*, LNCS 4281, 2006.

[PS06b] G. J. Pace and G. Schneider. Static analysis for state-space reduction of polygonal hybrid systems. In *FORMATS'06*, LNCS 4202, 2006.

[PS08] G. J. Pace and G. Schneider. Relaxing Goodness is Still Good for SPDIs. Technical Report 372, Dept. of Informatics, Univ. of Oslo, Norway, Feb. 2008.

[Sch08] G. Schneider. Reachability analysis of Generalized Polygonal Hybrid Systems. In *ACM SAC-SV'08*, pages 327–332. ACM, March 2008.

[Spe] SpeeDI$^+$. http://www.cs.um.edu.mt/speedi/.