A Compositional Algorithm for Parallel Model Checking of Polygonal Hybrid Systems

Gordon 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. The reachability problem as well as the computation of the phase portrait for the class of planar hybrid systems defined by constant differential inclusions (SPDI), has been shown to be decidable. The existing reachability algorithm is based on the exploitation of topological properties of the plane which are used to accelerate certain kind of cycles. The complexity of the algorithm makes the analysis of large systems generally unfeasible. In this paper we present a compositional parallel algorithm for reachability analysis of SPDIs. The parallelization is based on the qualitative information obtained from the phase portrait of an SPDI, in particular the controllability kernel.

1 Introduction

Hybrid systems are systems in which the discrete and the continuous worlds coexist. Examples can be found in avionics, robotics, bioinformatics and highway systems. For the majority of non trivial systems, reachability and most verification questions are undecidable. Various decidable subclasses have, subsequently, been identified, including timed [AD94] and rectangular automata [HKPV95], hybrid automata with linear vector fields [LPY01], piecewise constant derivative systems (PCDs) [MP93] and polygonal differential inclusion systems[3] (SPDIs) [ASY01], just to mention a few. From the practical point of view, a proof of decidability of reachability is only useful if accompanied with a decision procedure for effectively computing it, which is the case in the above-mentioned examples. Also of importance is the complexity of the algorithm: How expensive is it to compute reachability? Is it feasible with reasonable memory and time requirements? How large are the systems we can treat? Only in a few cases have the algorithms found scaled up to large industrial systems, and obtaining faster and cheaper algorithms is still an ongoing research challenge. One approach is the identification of smart ways of parallelizing and distributing reachability algorithms.

Reduction of memory and time requirements are the main reasons for seeking parallelization. In verification, in particular, the main bottleneck is usually memory. The effort in distributed programming is usually put on finding good ways

[3] In the literature the name *simple planar differential inclusion* has been used to describe the same class of systems.

of partitioning the task among different processes in order to keep a balanced distribution of the use of memory and execution time. An important issue is the communication cost; it is desirable to have a good ratio between process computation and communication time, where the communication cost should not be much greater than the analysis cost of the original system without parallelization. One way of reducing communication cost in distributed algorithms in general, is *compositionality*, that is dividing the problem in independent smaller ones. The partial results are then combined in order to exactly answer the original question. This approach reduces communication between processes to a minimum — communication is only carried out at instantiation and when returning the result.

Given the non-compositional nature of hybrid systems, obtaining distributed reachability algorithms for hybrid systems is a challenging task. A qualitative analysis of hybrid systems may, however, provide useful information for partitioning the state-space in independent subspaces, thus helping in achieving compositional analysis.

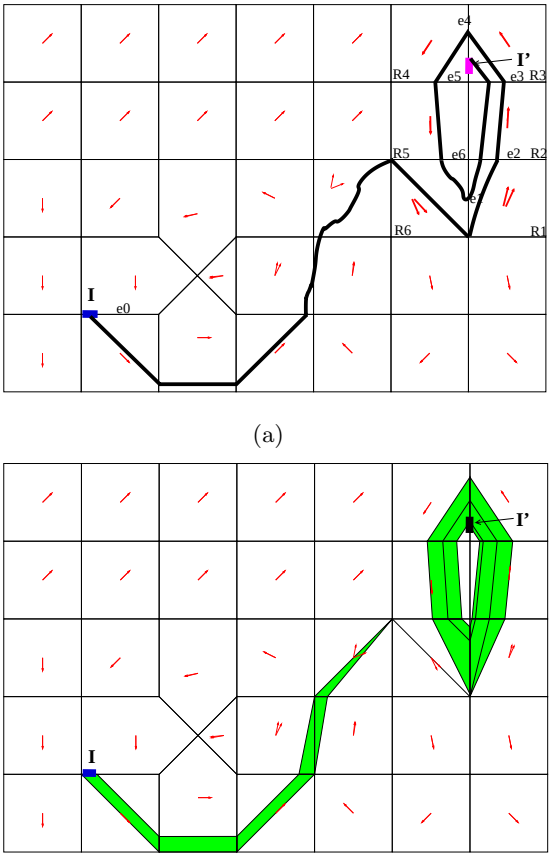
In this paper we present a compositional algorithm for parallel reachability analysis of polygonal differential inclusion systems. The identification and computation of controllability kernels is the core of our algorithm and the main reason for compositionality. We also give a lower bound for the number of parallel processes which may be launched for computing reachability in an independent way, each one operating in smaller state spaces than the original, and we prove soundness and completeness of our algorithm.

2 Preliminaries

A (positive) *affine* function $f : \mathbb{R} \to \mathbb{R}$ is such that $f(x) = ax + b$ with $a > 0$. An *affine multivalued* function $F : \mathbb{R} \to 2^{\mathbb{R}}$, denoted $F = \langle f_l, f_u \rangle$, is defined by $F(x) = \langle f_l(x), f_u(x) \rangle$ where f_l and f_u are 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 $F(\langle l, u \rangle) = \langle f_l(l), f_u(u) \rangle$. The *inverse* of F is defined by $F^{-1}(x) = \{y \mid x \in F(y)\}$. It is not difficult to show that $F^{-1} = \langle f_u^{-1}, f_l^{-1} \rangle$.

A *truncated affine multivalued* function (TAMF) $\mathcal{F} : \mathbb{R} \to 2^{\mathbb{R}}$ is defined by an affine multivalued function F and intervals $S \subseteq \mathbb{R}^+$ and $J \subseteq \mathbb{R}^+$ as follows: $\mathcal{F}(x) = F(x) \cap J$ if $x \in S$, otherwise $\mathcal{F}(x) = \emptyset$. For convenience we write $\mathcal{F}(x) = F(\{x\} \cap S) \cap J$. For an interval I, $\mathcal{F}(I) = F(I \cap S) \cap J$ and $\mathcal{F}^{-1}(I) = F^{-1}(I \cap J) \cap S$. We say that $\mathcal{F}$ is *normalized* if $S = \mathsf{Dom}(\mathcal{F}) = \{x \mid F(x) \cap J \neq \emptyset\}$ (thus, $S \subseteq F^{-1}(J)$) and $J = \mathsf{Im}(\mathcal{F}) = \mathcal{F}(S)$. TAMFs are closed under composition:

Theorem 1 ([ASY01]). The composition of two 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 $J = J_2 \cap F_2(J_1 \cap S_2)$. □



(a)

(b)

Fig. 1. (a) An SPDI and its trajectory segment; (b) Reachability analysis

2.1 SPDIs

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 $\mathbf{a}$.
A *polygonal differential inclusion system* (SPDI) is defined by giving a finite partition[4] $\mathbb{P}$ of the plane into convex polygonal sets, and associating with each $P \in \mathbb{P}$ a couple of vectors $\mathbf{a}_P$ and $\mathbf{b}_P$. Let $\phi(P) = \angle_{\mathbf{a}_P}^{\mathbf{b}_P}$. The SPDI's behavior at a point $\mathbf{x} \in P$ is expressed by the differential inclusion $\dot{\mathbf{x}} \in \phi(P)$.
Let $E(P)$ be the set of edges of P. We say that e is an *entry* of P if for all $\mathbf{x} \in e$ (considering only interior points of 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 $\epsilon < 0$.

[4] Since the edges of the adjacent polygons are shared, more precisely it is a closed cover with disjoint interiors.

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.

Assumption 1 *All the edges in $E(P)$ are either entries or exits, that is, $E(P) = in(P) \cup out(P)$.*

Reachability for SPDIs is decidable provided the above assumption holds [ASY01]; without such assumption it is not known whether reachability is decidable.
A *trajectory segment* over $T \in \mathbb{R}$ 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 1. Consider the SPDI illustrated in Fig. 1-(a). For sake of simplicity we will only show the dynamics associated to regions R_1 to R_6 in the picture. For each region R_i, $1 \leq i \leq 6$, there is a pair of vectors $(\mathbf{a}_i, \mathbf{b}_i)$, where: $\mathbf{a}_1 = (45, 100), \mathbf{b}_1 = (1, 4)$, $\mathbf{a}_2 = \mathbf{b}_2 = (1, 10)$, $\mathbf{a}_3 = \mathbf{b}_3 = (-2, 3)$, $\mathbf{a}_4 = \mathbf{b}_4 = (-2, -3)$, $\mathbf{a}_5 = \mathbf{b}_5 = (1, -15)$, $\mathbf{a}_6 = (1, -2), \mathbf{b}_6 = (1, -1)$. A trajectory segment starting on interval $I \subset e_0$ and finishing in interval $I' \subset e_4$ is depicted. ■

We say that a signature σ is *feasible* if and only if there exists a trajectory segment ξ with signature σ, i.e., $\mathsf{Sig}(\xi) = \sigma$. From this definition, it immediately follows that extending an unfeasible signature, can never make it feasible.

Successors and predecessors Given an SPDI, we fix a one-dimensional coordinate system on each edge to represent points laying on edges [ASY01]. For notational convenience, we indistinctly use letter e to denote the edge or 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 $I \subseteq e$, $\mathsf{Succ}_{e,e'}(I)$ 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., $\xi(0) \in I \wedge \xi(t) \in e' \wedge \mathsf{Sig}(\xi) = ee'$). $\mathsf{Succ}_{e,e'}$ is a TAMF [ASY01].

Example 2. Let $e_1, \ldots, e_6$ be as in Fig. 1-(a) and $I = [l, u]$. We assume a one-dimensional coordinate system. We show only the first and last edge-to-edge TAMF of the cycle:

$$\begin{array}{lll} F_{e_1e_2}(I) = \left[\frac{l}{4}, \frac{9}{20}u\right], & S_1 = [0, 10], & J_1 = \left[0, \frac{9}{2}\right] \\ F_{e_6e_1}(I) = [l, 2u], & S_6 = [0, 10], & J_6 = [0, 10] \end{array}$$

with $\mathsf{Succ}_{e_ie_{i+1}}(I) = F_{e_ie_{i+1}}(I \cap S_i) \cap J_i$, for $1 \leq i \leq 6$; S_i and J_i are computed as shown in Theorem 1. ■

Given a sequence $w = e_1, e_2, \ldots, e_n$, the successor of I along w defined as $\mathsf{Succ}_w(I) = \mathsf{Succ}_{e_{n-1},e_n} \circ \ldots \circ \mathsf{Succ}_{e_1,e_2}(I)$ is a TAMF.

Example 3. Let $\sigma = e_1 \cdots e_6 e_1$. We have that $\mathsf{Succ}_\sigma(I) = F(I \cap S_\sigma) \cap J_\sigma$, where: $F(I) = [\frac{l}{4} + \frac{1}{3}, \frac{9}{10}u + \frac{2}{3}]$, with $S_\sigma = [0, 10]$ and $J_\sigma = [\frac{1}{3}, \frac{29}{3}]$. ■

For $I \subseteq e'$, $\mathsf{Pre}_{e,e'}(I)$ is the set of points in e that can reach a point in I by a trajectory segment in P. The definition can be extended straightforwardly to signatures $\sigma = e_1 \cdots e_n$, $\mathsf{Pre}_\sigma(I)$.

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 $F = \langle f_l, f_u \rangle$ (we suppose that this representation is normalized). We denote by $\mathcal{D}_\sigma$ the one-dimensional discrete-time dynamical system defined by Succ_σ, that is $x_{n+1} \in \mathsf{Succ}_\sigma(x_n)$.

Assumption 2 *None of the two functions f_l, f_u is the identity.*

Let l^* and u^* be the fix-points[5] of f_l and f_u, respectively, and $S_\sigma \cap J_\sigma = \langle L, U \rangle$. A simple cycle is of one of the following types [ASY01]: 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, $L \leq l^* \leq U < u^*$.

Example 4. Let $\sigma = e_1 \cdots e_6 e_1$. We have that $S_\sigma \cap J_\sigma = \langle L, U \rangle = [\frac{1}{3}, \frac{29}{3}]$. The fix-points of the Eq. given in Example 3 are such that $\frac{1}{3} < l^* = \frac{11}{25} < u^* = \frac{20}{3} < \frac{29}{3}$. Thus, σ is a STAY. ■

Any trajectory that enters a cycle of type DIE will eventually quit it after a finite number of turns. If the cycle is of type 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 proving decidability of the reachability problem.

[5] The fix-point x^* is computed by solving the equation $f(x^*) = x^*$, where $f(\cdot)$ is positive affine.

Reachability Analysis It has been shown that reachability is decidable for SPDIs. Proof of the decidability result is constructive, giving an algorithmic procedure $Reach(\mathcal{S}, e, e')$ based on a depth-first search algorithm. An alternative breadth-first search algorithm which can deal with multiple edges has been presented in [PS03].

Theorem 2 ([ASY01]). *The reachability problem for SPDIs is decidable.* □

An edgelist is a set of intervals of edges. Given edgelists I and I', we denote the reachability of (some part of) I' from (some part of) I as $Reach(\mathcal{S}, I, I')$. Clearly, using the decidability result on edge intervals, reachability between edgelists is decidable. Although decidability may be point-to-point, edge-to-edge, edgelist-to-edgelist and region-to-region, in the rest of this paper, we will only talk about edgelist reachability. We define the following predicate: $I \xrightarrow{\mathcal{S}} I' \equiv Reach(\mathcal{S}, I, I')$.

Example 5. Consider the SPDI of Fig. 1-(a). Fig. 1-(b) shows part of the reach set of the interval $[8, 10] \subset e_0$, answering positively to the reachability question: Is $[1, 2] \subset e_4$ reachable from $[8, 10] \subset e_0$? Fig. 1-(b) has been automatically generated by the SPeeDI toolbox [APSY02] we have developed for reachability analysis of SPDIs based on the results of [ASY01]. ■

2.2 Controllability and Viability Kernels

We recall the definition of controllability and viability kernels of an SPDI and we show how to obtain such kernels — proofs are omitted and for further details, refer to [ASY02]. In the following, given σ a cyclic signature, we define K_σ as follows: $K_\sigma = \bigcup_{i=1}^{k} (int(P_i) \cup e_i)$ where P_i is such that $e_{i-1} \in in(P_i)$, $e_i \in out(P_i)$ and $int(P_i)$ is P_i's interior.

We say that K, a subset of $\mathbb{R}^2$, is *controllable* if for any two points $\mathbf{x}$ and $\mathbf{y}$ in K there exists a trajectory segment ξ starting in $\mathbf{x}$ that reaches an arbitrarily small neighborhood of $\mathbf{y}$ without leaving K. More formally: A set K is controllable if $\forall \mathbf{x}, \mathbf{y} \in K, \forall \delta > 0, \exists \xi : [0, t] \to \mathbb{R}^2, t > 0 \;.\; (\xi(0) = \mathbf{x} \wedge |\xi(t) - \mathbf{y}| < \delta \wedge \forall t' \in [0, t] \;.\; \xi(t') \in K)$. The *controllability kernel* of K, denoted $\mathsf{Cntr}(K)$, is the largest controllable subset of K.

For $I \subseteq e_1$ we define $\overline{\mathsf{Pre}_\sigma}(I)$ to be the set of all $\mathbf{x} \in \mathbb{R}^2$ for which there exists a trajectory segment ξ starting in $\mathbf{x}$, that reaches some point in I, such that $\mathsf{Sig}(\xi)$ is a suffix of $e_2 \ldots e_k e_1$. It is easy to see that $\overline{\mathsf{Pre}_\sigma}(I)$ is a polygonal subset of the plane which can be calculated using the following procedure. We start by defining: $\overline{\mathsf{Pre}_e}(I) = \{\mathbf{x} \mid \exists \xi : [0, t] \to \mathbb{R}^2, t > 0 \;.\; \xi(0) = \mathbf{x} \wedge \xi(t) \in I \wedge \mathsf{Sig}(\xi) = e\}$ and apply this operation k times: $\overline{\mathsf{Pre}_\sigma}(I) = \bigcup_{i=1}^{k} \overline{\mathsf{Pre}_{e_i}}(I_i)$ with $I_1 = I$, $I_k = \mathsf{Pre}_{e_k, e_1}(I_1)$ and $I_i = \mathsf{Pre}_{e_i, e_{i+1}}(I_{i+1})$, for $2 \leq i \leq k - 1$.

For $I \subseteq e_1$ let us define $\overline{\mathsf{Succ}_\sigma}(I)$ as the set of all points $\mathbf{y} \in \mathbb{R}^2$ for which there exists a trajectory segment ξ starting in some point $\mathbf{x} \in I$, that reaches $\mathbf{y}$, such that $\mathsf{Sig}(\xi)$ is a prefix of $e_1 \ldots e_k$. The successor $\overline{\mathsf{Succ}_\sigma}(I)$ is a polygonal subset of the plane which can be computed similarly to $\overline{\mathsf{Pre}_\sigma}(I)$.

For a given cyclic signature σ, we define $\mathcal{C}_{\mathcal{D}}(\sigma)$ as follows:

$$\mathcal{C}_{\mathcal{D}}(\sigma) = \begin{cases} \langle L, U \rangle & \text{if } \sigma \text{ is EXIT-BOTH} \\ \langle L, u^* \rangle & \text{if } \sigma \text{ is EXIT-LEFT} \\ \langle l^*, U \rangle & \text{if } \sigma \text{ is EXIT-RIGHT} \\ \langle l^*, u^* \rangle & \text{if } \sigma \text{ is STAY} \\ \emptyset & \text{if } \sigma \text{ is DIE} \end{cases} \tag{1}$$

$\mathcal{C}_{\mathcal{D}}(\sigma)$ is an interval on the first edge of the signature σ with the property that any point on such interval is reachable from any other point in the interval, and conversely. We compute the controllability kernel of K_σ as follows:

Theorem 3 ([ASY02]). $\mathsf{Cntr}(K_\sigma) = (\overline{\mathsf{Succ}}_\sigma \cap \overline{\mathsf{Pre}}_\sigma)(\mathcal{C}_{\mathcal{D}}(\sigma))$. □

In what follows we present some definitions and a result which are crucial for obtaining a compositional algorithm for reachability analysis of SPDIs. See [PS06] for proofs and more details.

Let $\mathsf{Cntr}^l(K_\sigma)$ be the closed curve obtained by taking the leftmost trajectory and $\mathsf{Cntr}^u(K_\sigma)$ be the closed curve obtained by taking the rightmost trajectory which can remain inside the controllability kernel. In other words, $\mathsf{Cntr}^l(K_\sigma)$ and $\mathsf{Cntr}^u(K_\sigma)$ are the two polygons defining the controllability kernel.

A non-empty controllability kernel $\mathsf{Cntr}(K_\sigma)$ of a given cyclic signature σ partitions the plane into three disjoint subsets: (1) the controllability kernel itself, (2) the set of points limited by $\mathsf{Cntr}^l(K_\sigma)$ (and not including $\mathsf{Cntr}^l(K_\sigma)$) and (3) the set of points limited by $\mathsf{Cntr}^u(K_\sigma)$ (and not including $\mathsf{Cntr}^u(K_\sigma)$).

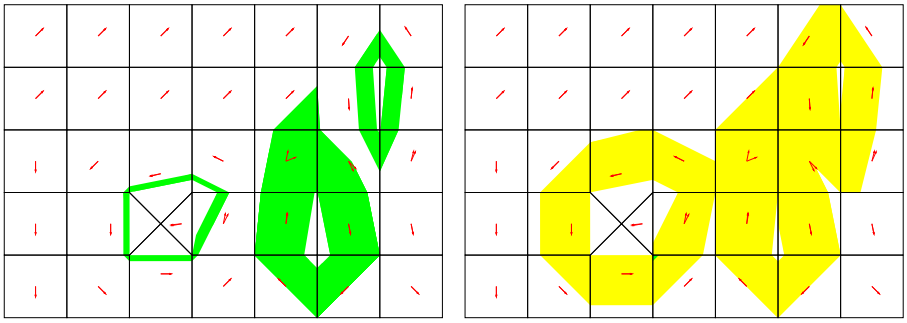
We define the *inner* of $\mathsf{Cntr}(K_\sigma)$ (denoted by $\mathsf{Cntr}_{in}(K_\sigma)$) to be the subset defined by (2) above if the cycle is counter-clockwise or to be the subset defined by (3) if it is clockwise. The *outer* of $\mathsf{Cntr}(K_\sigma)$ (denoted by $\mathsf{Cntr}_{out}(K_\sigma)$) is defined to be the subset which is not the inner nor the controllability itself.

We proceed now by defining and stating the computability result of viability kernels. A trajectory ξ is *viable* in K if $\xi(t) \in K$ for all $t \geq 0$. K is a *viability domain* if for every $\mathbf{x} \in K$, there exists at least one trajectory ξ, with $\xi(0) = \mathbf{x}$, which is viable in K. The *viability kernel* of K, denoted $\mathsf{Viab}(K)$, is the largest viability domain contained in K.

The following result provides a non-iterative algorithmic procedure for computing the viability kernel of K_σ on an SPDI:

Theorem 4 ([ASY02]). *If σ is DIE,* $\mathsf{Viab}(K_\sigma) = \emptyset$*, otherwise* $\mathsf{Viab}(K_\sigma) = \overline{\mathsf{Pre}}_\sigma(S_\sigma)$. □

Note that an edge in the SPDI may intersect a kernel. In such cases, we can generate a different SPDI, with the same dynamics but with the edge split into parts, such that each part is completely inside, on or outside the kernel. Although the signatures will obviously change, it is easy to prove that the behavior of the SPDI remains identical to the original. In the rest of the paper, we will assume that all edges are either completely inside, on or completely outside the kernels. We note that in practice splitting is not necessary since we can just consider parts of edges.



(a) Controllability kernels (b) Viability kernels

Fig. 2. Kernels of the SPDI in Fig. 1

Example 6. Fig. 2 shows all the controllability and viability kernels of the SPDI given in Example 1. There are 4 cycles with controllability and viability kernels — in the picture two of the kernels are overlapping. ■

Properties of the Kernels. Before stating two results relating controllability and viability kernels, we need the following definition:

Definition 1. *Given a controllability kernel C (of a loop σ — $C = \mathsf{Cntr}(K_\sigma)$), then let C^+ be the related viability kernel ($C^+ = \mathsf{Viab}(K_\sigma)$), C_{in} be the inside of the kernel, and C_{out} be the outside.*

Proposition 3 in [PS06] gives conditions for feasible trajectories traversing controllability kernels. The following is a generalization of such result:

Proposition 1. *Given two edges e and e', one lying completely inside a kernel, and the other outside or on the same kernel, such that ee' is feasible, then there exists a point on the kernel, which is reachable from e and from which e' is reachable.* □

The following corollary follows from [PS06, Proposition 2], asserting that the controllability kernel is the local basin of attraction of the viability kernel:

Corollary 1. *Given an controllability kernel C, and related viability kernel C^+, then for any $e \subseteq C^+$, $e' \subseteq C$, there exists a feasible path $e\sigma e'$.* □

3 Independent Questions and Parallelization

3.1 SPDI Decomposition

In this section, we propose a number of theorems which, given an SPDI and a reachability question, for each controllability kernel, allow us to either (i) answer the reachability question without any further analysis; or (ii) reduce the

state space necessary for reachability analysis; or (iii) decompose the reachability question into two smaller, and independent reachability questions.
The following theorem enables us to answer certain reachability questions without any analysis, other than the identification of controllability and viability kernels. This result is based on two properties, that within the controllability kernel of a loop, any two points are mutually reachable, and that any point on the viability kernel of the same loop can eventually reach the controllability kernel. Therefore if the source edgelist lies (possibly partially) within the viability kernel of a loop, and the destination edgelist lies (possibly partially) within the controllability kernel of the same loop, then, there must exist a trajectory from the source to the destination edgelist. The full proof of this result can be found in [PS06].

Theorem 5. *Given an SPDI $\mathcal{S}$, two edgelists I and I' and a controllability kernel C, then if $I \subseteq C^+$ and $I' \subseteq C$, then $I \xrightarrow{\mathcal{S}} I'$.* □

The following theorem allows us to reduce the state space based on controllability kernels. If both the source and destination edgelists lie on the same side of a controllability kernel, then we can disregard all edges on the other side of the kernel. The full proof of this result can be found in [PS06].

Theorem 6. *Given an SPDI $\mathcal{S}$, two edgelists I and I' and a controllability kernel C, then if $I \subseteq C_{in}$ and $I' \subseteq C_{in}$, then $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}\setminus C_{out}} I'$. Similarly, if $I \subseteq C_{out}$ and $I' \subseteq C_{out}$, then $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}\setminus C_{in}} I'$.* □

Finally, the following new result allows us to decompose a reachability question into two smaller questions independent of each other. The theorem states that if the source and destination edgelists lie on opposite sides of a controllability kernel, then we can try (i) to reach the related viability kernel from the source edgelist, and (ii) to reach the destination from the controllability kernel. The original reachability question can be answered affirmatively if and only if both these questions are answered affirmatively.

Theorem 7. *Given an SPDI $\mathcal{S}$, two edgelists I and I' and a controllability kernel C, then if $I \subseteq C_{in}$ and $I' \subseteq C_{out}$, then $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}\setminus C_{out}} C^+ \wedge C \xrightarrow{\mathcal{S}\setminus C_{in}} I'$. Similarly, if $I \subseteq C_{out}$ and $I' \subseteq C_{in}$, then $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^+ \wedge C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$.*

Proof. Without loss of generality, let $I \subseteq C_{in}$ and $I' \subseteq C_{out}$.

Soundness of decomposition: Let us assume that $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^+$ and $C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$. From $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^+$ we can conclude that there are partial edges $e_0 \subseteq I$ and $e_m \subseteq C^+$, and a path σ in $(\mathcal{S} \setminus C_{out})$, such that $e_0\sigma e_m$ is a feasible path.

Similarly, from $C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$ we can conclude that there are partial edges $e_{m'} \subseteq C$ and $e_f \subseteq I'$, and a path σ' in $(\mathcal{S} \setminus C_{in})$, such that $e_{m'}\sigma' e_f$ is a

feasible path. However, since $e_{m'}$ is in a controllability kernel, and e_m is in the related viability kernel, then by corollary 1, there exists a feasible path $e_m\sigma'' e_{m'}$ in $\mathcal{S}$. Therefore, $e_0\sigma e_m\sigma'' e_{m'}\sigma' e_f$ is a feasible path in $\mathcal{S}$. Since $e_0 \subseteq I$ and $e_f \subseteq I'$, we can conclude that $I \xrightarrow{\mathcal{S}} I'$.

Completeness of decomposition: Conversely, let us assume that $I \xrightarrow{\mathcal{S}} I'$. Then, there must be edges $e_0 \subseteq I$ and $e_f \subseteq I'$ such that $e_0\sigma e_f$ is feasible in $\mathcal{S}$. By the Jordan curve theorem [Hen79], the trajectory must cross $\mathsf{Cntr}^l(K_\sigma)$ or $\mathsf{Cntr}^u(K_\sigma)$ at least once, meaning that there exists a partial edge e_m in the controllability kernel C such that $e_0\sigma_1 e_m\sigma_2 e_f$ is feasible. But every subpath of a feasible path is itself feasible, meaning that both $e_0\sigma_1 e_m$ and $e_m\sigma_2 e_f$ are feasible in $\mathcal{S}$, implying that $I \xrightarrow{\mathcal{S}} C^+$ and $C \xrightarrow{\mathcal{S}} I'$. Consider the feasible path $e_0\sigma_1 e_m$. Recall that $I \subseteq C_{in}$, and that $e_0 \subseteq I$, hence $e_0 \subseteq C_{in}$. Assume that σ_1 contains some edges in C_{out}, and let f be the first such edge. The path is thus: $e_0\sigma_a f\sigma_b e_m$. Since f is the first edge inside the kernel, it follows that the last element of σ_a is outside the kernel. Using proposition 1, it follows that there exists a point p on the kernel reachable from the last element of σ_a. We have thus obtained a shorter discrete path $e_0\sigma_a p$ which is feasible and no point of which lies inside the kernel. Therefore, $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^+$.
Similarly, we can prove that $C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$. □

3.2 Unavoidable Kernels

Unavoidable kernels are defined geometrically to be kernels which a straight line from the source interval to the destination interval 'intersects' an odd number of times. We call the kernel unavoidable since it can be proved that any path from the source to the destination will have to pass through the kernel.

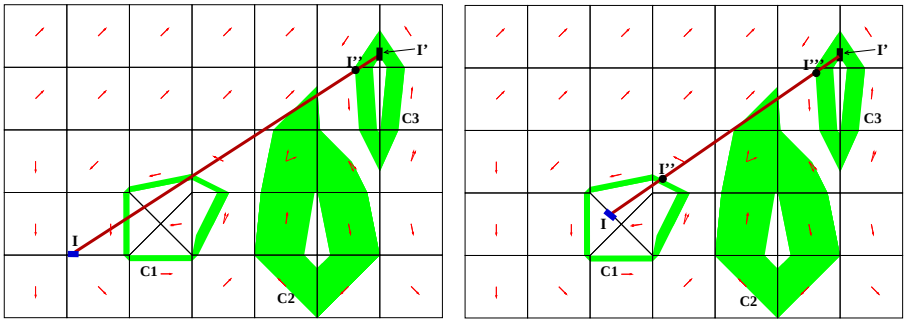
Definition 2. *Given an SPDI $\mathcal{S}$ and two edgelists I and I', we say that a controllability kernel $\mathsf{Cntr}(K_\sigma)$ is* unavoidable *if any segment of line with extremes on points lying on I and I' intersects with both the edges of $\mathsf{Cntr}^l(K_\sigma)$ and those of $\mathsf{Cntr}^u(K_\sigma)$ an odd number of times (disregarding tangential intersections with vertices).*

The following theorem enables us to discover separating controllability kernels using a simple geometric test.

Theorem 8. *Given an SPDI $\mathcal{S}$, two edgelists I and I', and a controllability kernel $C = \mathsf{Cntr}(K_\sigma)$, then C is an unavoidable kernel if and only if one of the following conditions holds (i) $I \subseteq C_{in}$ and $I' \subseteq C_{out}$; or (ii) $I \subseteq C_{out}$ and $I' \subseteq C_{in}$.*

Proof. This theorem is a standard geometrical technique frequently used in computer graphics [FvDFH96] (referred to as the odd-parity test). □

Corollary 2. *Given an SPDI $\mathcal{S}$, two edgelists I and I', and an unavoidable controllability kernel $C = \mathsf{Cntr}(K_\sigma)$, then $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}} C$ and $C \xrightarrow{\mathcal{S}} I'$.*



(a) (b)

Fig. 3. Unavoidable kernels and independent reachability questions

Proof. This follows directly from theorems 6 and 8. □

The following result relates unavoidable kernels between each other:

Proposition 2. *Given two disjoint controllability kernels C and C', both unavoidable from I to I', then either C' is unavoidable from I to C or C' is unavoidable from C to I', but not both.*

Proof. This follows directly from definition of unavoidable kernel, the disjointness assumption and theorem 8. □

3.3 Counting Sub-Problems

The following theorem bounds the number of times a reachability question may be decomposed into independent sub-questions using theorem 7. We will consider a collection of mutually disjoint controllability kernels.

Theorem 9. *Given an SPDI $\mathcal{S}$ and two edgelists I and I', the question $I \xrightarrow{\mathcal{S}} I'$ can be split into no more than k reachability questions, k is the number of mutually-disjoint controllability kernels.*

Proof. We note that whenever we decompose an SPDI, the source and destination intervals are always within the sub-SPDI under consideration.
Now consider a reachability question $I \xrightarrow{\mathcal{S}} I'$, and a controllability kernel C upon which we can branch. Without loss of generality, we assume $I \subseteq C_{in}$ and $I' \subseteq C_{out}$. The question is thus decomposed into two questions: $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^{+}$ (question a) and $C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$ (question b).
Now consider another controllability kernel C'. We now have two possible cases: (i) $C' \subseteq C_{in}$; (ii) $C' \subseteq C_{out}$. Note that by the Jordan curve theorem, C' cannot be partially in and partially out of C without intersecting C.
In the first case (i), we note that since all edges in $\mathcal{S}\setminus C_{out}$ lie outside or inside C', this new kernel cannot be be used to split question (b) or any question derived

from it. Therefore, C' can only induce a split in question (a). Case (ii), is the mirror case and the argument follows identically.
Therefore, each controllability kernel can contribute at one extra process, bounding the number of reachability questions to k. □

We now give a lower bound on the number of independent questions induced by theorem 7 in terms of the number of mutually-disjoint unavoidable controllability kernels.

Theorem 10. *Given an SPDI $\mathcal{S}$ and two edgelists I and I', the question $I \xrightarrow{\mathcal{S}} I'$ can be split into at least $u+1$ reachability questions, u is the number of mutually-disjoint unavoidable controllability kernels.*

Proof. We prove this by induction on the number of mutually-disjoint unavoidable controllability kernels.
With $u = 0$, the number of questions is clearly at least $u + 1$.
Now consider the unavoidable controllability kernel C, and the question $I \xrightarrow{\mathcal{S}} I'$. By theorem 8, it follows that I and I' are on opposite sides of C. The reachability question can be thus decomposed to $I \xrightarrow{\mathcal{S}\setminus C_{in}} C^+$ (question a) and $C \xrightarrow{\mathcal{S}\setminus C_{out}} I'$ (question b) by theorem 7. Also, by proposition 2, we known that any other unavoidable controllability kernel C' from I to I', is also an unavoidable controllability kernel from either I to C or from C to I' (but not both). In both cases we obtain a decomposition of the reachability question into $I \xrightarrow{\mathcal{S}\setminus C_{out}} C$ and $C \xrightarrow{\mathcal{S}\setminus C_{in}} C'$ (or, $I \xrightarrow{\mathcal{S}\setminus C'_{out}} C'$ and $C' \xrightarrow{\mathcal{S}\setminus C'_{in}} C$). Splitting the kernels into the relevant ones to the two questions (u_1 kernels relevant to question a and u_2 relevant to question b — $u = u_1 + u_2 + 1$), we can conclude that the number of questions we get is $(u_1 + 1) + (u_2 + 1)$ which is $u + 1$. □

We have thus given lower and upper bounds on the the number of independent questions generated by applying theorem 7 over a number of mutually disjoint unavoidable controllability kernels. The results may be extended to work with overlapping kernels.

Example 7. Let us consider again the SPDI defined in Example 1 and the same intervals I and I'. In Fig. 3-(a) we show the unavoidable kernels. The segment of line from I to I' traverses C_1 and C_2 twice and C_3 exactly once (an odd number of times). Thus, only C_3 is an unavoidable kernel. The reachability question can be split into at least 2 independent questions: $I \xrightarrow{\mathcal{S}\setminus C3_{in}} I''$ and $I'' \xrightarrow{\mathcal{S}\setminus C3_{out}} I'$.
As another example let us consider I and I' as in Fig. 3-(b). The segment of line from I to I' traverses C_1 and C_3 exactly once (an odd number of times), while C_2 is traversed twice. Thus, there are two unavoidable kernels, namely C_1 and C_3. In this case the reachability question can be split into at least 3 independent questions: $I \xrightarrow{\mathcal{S}\setminus C1_{out}} I''$, $I'' \xrightarrow{\mathcal{S}\setminus (C1_{in} \cup C3_{in})} I'''$, and $I''' \xrightarrow{\mathcal{S}\setminus C3_{out}} I'$. ■

4 Parallel Reachability Algorithm

In Fig. 4 we give an algorithm for parallel reachability analysis of SPDIs using parallel recursive calls corresponding to independent reachability questions.

```
function ReachPar(S, I, I') =
  ReachParKernels (S, ControllabilityKernels(S), I, I')

function ReachParKernels(S, [], I, I') =
  Reach(S, I, I');

function ReachParKernels(S, k:ks, I, I') =
  if (ImmedieteAnswer(S, I, I')) then
     True;
  elsif (SameSideOfKernel(S, k, I, I')) then
     S_I := S \ EdgesOnOtherSideOf(S, k, I');
     ReachParKernels(S_I, ks, I, I');
  else
     S_I := S \ EdgesOnOtherSideOf(S, k, I);
     S_I' := S \ EdgesOnOtherSideOf(S k, I');
     parbegin
        r1 := ReachParKernels(S_I, ks, I, viability(k));
        r2 := ReachParKernels(S_I', ks, k, I');
     parend;
     return (r1 and r2);
```

Fig. 4. Parallel algorithm for reachability of SPDIs.

The function `ReachParKernels` is called with the SPDI to consider, a list of kernels still to be used for reduction, and the source and destination edgelists. With no kernels to consider, the algorithm simply calls the standard sequential algorithm (`Reach`). Otherwise, one of the kernels is analyzed, with three possible cases:

1. If the source lies (possibly partially) on the extended kernel, and the destination lies (possibly partially) on the kernel, then we can give an immediate answer (using theorem 5).
2. If both the edgelists lie on the same side of the kernel, then we simply eliminate redundant parts of the SPDI — anything on the other side of the kernel (theorem 6).
3. Otherwise, if the kernels both lie on opposite sides of the kernel, we can split the problem into two independent questions (reaching the kernel from the source, and the destination from the kernel) which can be run in parallel (theorem 7). An affirmative answer from both these subquestions is equivalent to an affirmative answer to the original question.

Note that the function `ReachParKernels` is compositional in the sense that each recursive call launch a process which operates in (most cases in) disjoint state spaces which are smaller than the original one ($\mathcal{S}$). The final answer is the composition of the partial reachability questions.

Given two edgelists I and I', we define the following predicate $I \xrightarrow{\mathcal{S}}_{\|} I' \equiv \texttt{ReachPar}(\mathcal{S}, I, I')$. The following theorem states that the (compositional) parallel algorithm exactly answers the reachability question, also giving a soundness and completeness proof of the algorithm:

Theorem 11. *Given an SPDI $\mathcal{S}$ and two intervals $I \subseteq e$ and $I' \subseteq e'$, $I \xrightarrow{\mathcal{S}} I'$ if and only if $I \xrightarrow{\mathcal{S}}_{\|} I'$.*

Proof. The proof follows from theorems 5, 6 and 7 and induction on ks. □

5 Concluding Remarks

We have shown how answering reachability on an SPDI can be reduced to a number of smaller reachability questions. Moreover, our algorithm can be combined with recent optimizations developed for the reachability analysis of SPDIs [PS06] in order to optimize each local reachability question.

We note that due to the fact that we present the algorithm working on edgelists, the breadth-first-search algorithm we present in [PS03] is better adapted than the original algorithm [ASY01] to be used in this context. The algorithm consists essentially in partitioning[6] the state space into parts, discarding some of these partitions and performing reachability analysis on others. Furthermore, as long as separate reachability analysis of two disjoint state spaces is not more expensive than performing reachability analysis on the state spaces merged together (which is true for any reachability algorithm with complexity worse than linear) the state space partitioning provides a speedup over global model checking.

Part of our algorithm is based on certain geometric tests (e.g., theorem 8) which may be avoided if we consider a more abstract approach by enriching the reachability graph with information about the order among edges of each SPDI region. This is part of our on-going work, as well as the study of a variant of the algorithm which executes exactly $u + 1$ parallel processes, u being the number of unavoidable kernels.

Another natural question that arises is whether this can somehow be applicable to model checking of other models. To attempt to partially answer this question, we identify the properties of SPDIs that were used in the system decomposition. The property depends on the ability to identify subsets of the state space such that each such subset (i) is a strongly connected set (in terms of reachability); and (ii) partitions the state space into two — such that any state on one side that can reach states on the other side can do so via an intermediate state within the subset. These conditions are satisfied thanks to the planarity of SPDIs. In

[6] The division is *almost* a partition, since the controllability kernels may be shared between parts.

fact, the conditions can possibly be applied to systems with a planar state graph. The application and generalization of the results presented here remains an open research area.

One current research direction is to apply our results to semi-decide the reachability question for SPDIs defined on 2-dimensional manifolds, for which the decidability of reachability remains unresolved [AS02]. Maybe the most prominent application of SPDIs is for approximating complex non-linear differential equations on the plane, for which an exact solution is not known. The decidability of SPDI's reachability and of its phase portrait construction would be of invaluable help for the qualitative analysis of such equations. The challenge would be to find an "intelligent" partition of the plane in order to get an optimal approximation of the equations. Since such partition might produce a high number of regions, our parallel algorithm might be extremely useful here.

References

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[APSY02] E. Asarin, G. Pace, G. Schneider, and S. Yovine. SPeeDI: a verification tool for polygonal hybrid systems. In *CAV'2002*, volume 2404 of *LNCS*, pages 354–358, 2002.

[AS02] E. Asarin and G. Schneider. Widening the boundary between decidable and undecidable hybrid systems. In *CONCUR'2002*, volume 2421 of *LNCS*, pages 193–208, 2002.

[ASY01] E. Asarin, G. Schneider, and S. Yovine. On the decidability of the reachability problem for planar differential inclusions. In *HSCC'2001*, number 2034 in LNCS, pages 89–104, 2001.

[ASY02] E. Asarin, G. Schneider, and S. Yovine. Towards computing phase portraits of polygonal differential inclusions. In *HSCC'02*, volume LNCS 2289, 2002.

[FvDFH96] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer graphics (2nd ed. in C): principles and practice*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.

[Hen79] M. Henle. *A combinatorial introduction to topology*. Dover publications, Inc., 1979.

[HKPV95] T.A. Henzinger, P.W. Kopke, A. Puri, and P. Varaiya. What's decidable about hybrid automata? In *STOC'95*, pages 373–382. ACM Press, 1995.

[LPY01] G. Lafferriere, G. Pappas, and S. Yovine. Symbolic reachability computation of families of linear vector fields. *Journal of Symbolic Computation*, 32(3):231–253, September 2001.

[MP93] O. Maler and A. Pnueli. Reachability analysis of planar multi-linear systems. In *CAV'93*, pages 194–209. LNCS 697, Springer Verlag, July 1993.

[PS03] G. Pace and G. Schneider. Model checking polygonal differential inclusions using invariance kernels. In *VMCAI'04*, number 2937 in LNCS, pages 110–121. Springer Verlag, December 2003.

[PS06] G. Pace and G. Schneider. Static analysis of SPDIs for state-space reduction. Technical Report 336, Department of Informatics, University of Oslo, PO Box 1080 Blindern, N-0316 Oslo, Norway, April 2006.