

Denotational Semantics of Hybrid Automata

Abbas Edalat, Dirk Pattinson¹

Department of Computing, Imperial College London, UK

1 Introduction

A hybrid automaton [16,2] is a digital, real-time system that interacts with an analogue environment. Hybrid automata are ubiquitous in all areas of modern engineering and technology. For example, the (digital) height control of an automobile chassis depends on and influences the (continuous) driving conditions of the vehicle [22]. Hybrid automata typically operate in safety critical areas, such as the highway control systems [25,21] and air traffic control [24]. They combine a finite set of control states with continuous dynamics. In every control state, the continuous variables evolve according to an ordinary differential equation (or, more generally, differential inclusion [4]), and the system changes control states if the continuous variables reach certain thresholds; every such state change can involve non-continuous re-assignment of the continuous variables.

One of the key concerns in the theory of hybrid automata is the algorithmic verification of safety critical properties. This problem is well understood for linear systems, where the trajectories of the continuous variables are linear functions [3] and implemented in the model checker HyTech [17]. The situation for non-linear systems is, not surprisingly, much less satisfactory. While the approximation of non-linear hybrid automata by linear systems is asymptotically complete [18], it results in a huge blow-up in the number of discrete control states and associated state transitions, which limits the possibilities of algorithmic analysis.

This paper presents an alternative approach. Conceptually, we regard a hybrid automaton as the integration of two different types of systems: the evolution of a family of continuous systems, governed by differential equations, and the

Email addresses: ae@doc.ic.ac.uk (Abbas Edalat), dirk@doc.ic.ac.uk (Dirk Pattinson).

¹ Partially supported by the Nuffield Foundation under NAL/32589

dynamics of a discrete system given by a generalised iterated function system (IFS), see [20]. We synthesise the domain-theoretic approach to solving differential equations [9,11,13] and the domain-theoretic approach to obtain the attractor of an iterated function system [8] to develop a domain-theoretic semantics for general hybrid automata. The denotational semantics assigns to every time point t the set $\llbracket H \rrbracket(t)$ of states that the automaton H can reach at time t , starting from one of its initial states. The semantic function $\llbracket H \rrbracket$ is obtained as the least fixpoint in the (continuous) domain of compact-set valued functions of a real variable. Our first main results are correctness and computational adequacy of this denotational semantics w.r.t. the operational semantics, given in terms of a labelled transition system. While this provides a mathematical representation of the states visited at each particular point in time, we can now moreover use standard techniques of domain theory to actually compute this function. The implications are twofold: first, we obtain new results on the computability of trajectories based on the domain theoretic model of computation. Second, our analysis gives rise to a directly implementable algorithm that computes approximations to the semantic function $\llbracket H \rrbracket$ up to an arbitrary degree of accuracy, and hence gives approximations to the set of reachable states up to an arbitrary error bound. As the algorithms induced by our method work on bases of the involved domains, which can be defined in terms of either the rational or the dyadic numbers, this property is moreover guaranteed for implementations of our technique, as with rational arithmetic no rounding of real numbers is required.

Technically, the paper is divided in two parts. In the first part, we focus on flow automata, where the behaviour of the continuous variables in every discrete control state is governed by flow functions, which behave like the solutions of ordinary differential equations. In this setup, we formulate an operator on the domain of compact-set valued functions of a real variable that precisely captures the reachable states at any particular point in time. We impose two conditions on the automata under scrutiny: first, we require that the ingredients of the automaton, i.e. the flow and transition functions, give rise to Scott continuous functions on the respective domains. In order to show that the least fixpoint precisely captures the reachable states, we assume that the automaton is separated, i.e. has no transient states which the automaton can leave immediately (after 0 time units) after entering. We discuss these restrictions by means of examples, and show that the semantic function associated with a flow automaton cannot be computable in absence of these properties.

In the second part of the paper, we transfer the results obtained to hybrid automata, where the trajectories of the continuous variables are given by an ordinary differential equation. By instantiating earlier results on domain theoretic solutions of initial value problems [9,11,13] we show that we can effectively obtain the associated flows, thus reducing the problem of computing the

semantic function of a hybrid automaton to that of a flow automaton. Taken together, the domain theoretic approach provides a new computational model for the analysis of hybrid systems, and gives rise to both new computability results, and directly implementable data types and algorithms for the analysis of non-linear systems. Apart from providing algorithms that are guaranteed to capture the set of reachable states at any particular point in time, the semantic function associated with a hybrid automaton contains many further items of information, as it allows us for example to compute the first point in time where the system can enter into a particular control state; this will be exploited in further work.

Related Work. We have already mentioned symbolic techniques for the analysis of linear hybrid automata [3] and their implementation in the HyTech model checker [17]. Approximating non-linear systems by linear hybrid automata, while being asymptotically complete, results in an explosion of the number of discrete control states and the associated state transition functions [18], which is avoided by the domain theoretic approach. The domain theoretic approach of this paper is related to the interval analysis approach of [19], where interval numerical methods are used to compute over-approximations of the set of reachable states. In contrast to *loc.cit.*, where outward rounding is required if the result of an arithmetic operation is not machine representable, the domain theoretic model of computation actually allows to compute the semantic function up to an arbitrary degree of accuracy.

2 Preliminaries and Notation

We use basic notions of domain theory, see e.g. [1,15]. In particular, our analysis employs the following domains defined over the real numbers: the domain of n -dimensional compact rectangles extended with a least element

$$\mathbf{IR}^n = \{a \subseteq \mathbb{R}^n \mid a \text{ nonempty compact rectangle}\} \cup \{\mathbb{R}^n\},$$

ordered by reverse inclusion, and the *upper space*

$$\mathbf{UR}^n = \{c \subseteq \mathbb{R}^n \mid c \text{ nonempty and compact}\} \cup \{\mathbb{R}^n\}$$

of nonempty compact subsets of \mathbb{R}^n , also ordered by reverse inclusion [7]. A closed *semi rectangle* in \mathbb{R}^n is of the form $a_1 \times \dots \times a_n$, where the a_i are closed (not necessarily bounded) intervals in \mathbb{R} . If A is a semi-rectangle, we write $\mathbf{IA} = \{A \cap r \mid r \in \mathbf{IR}^n\}$ and $\mathbf{UA} = \{A \cap c \mid c \in \mathbf{UR}^n\}$ for the sub-domain of all elements above A . In particular, we will consider the domain $\mathbf{I}[0, \infty)$, whose bottom element is $\perp = [0, \infty)$.

We denote the extension of \mathbf{UA} (resp. \mathbf{IA}) with a top element $\top = \emptyset$ as $\mathbf{U}^\top \mathbf{A}$

(resp. $\mathbf{I}^\top \mathbf{A}$) and refer to them as the *extended upper space* (resp. the *extended interval domain*).

For a semi rectangle \mathbf{A} , \mathbf{IA} and \mathbf{U} are continuous Scott domains and $\mathbf{U}^\top \mathbf{A}$ and $\mathbf{I}^\top \mathbf{A}$ are continuous lattices. We often consider $\mathbf{IA} \subseteq \mathbf{U}^\top \mathbf{A}$ as a sub-domain without making this explicit; similarly, we identify $x \in \mathbb{R}^n$ with the degenerate hyper-rectangle $\{x\} \in \mathbf{IR}^n \subseteq \mathbf{U}^\top \mathbb{R}^n$. We write $\perp = \mathbf{A}$ for the least element of both \mathbf{IA} and \mathbf{UA} , and $\top = \emptyset$ for the top element of $\mathbf{U}^\top \mathbf{A}$ and $\mathbf{I}^\top \mathbf{A}$. Note that the way-below relation in \mathbf{UA} and \mathbf{IA} and their extensions $\mathbf{U}^\top \mathbf{A}$ and $\mathbf{I}^\top \mathbf{A}$ is given by $a \ll b$ iff $b \subseteq a^\circ$, where a° is the interior of a .

If $(C_i)_{i \in I}$ is a family of compact subsets $C_i \subseteq \mathbb{R}^{n_i}$, we identify $(x_i)_{i \in I} \in \prod_{i \in I} \mathbf{U}^\top C_i$ with the set $\{(i, y) \mid i \in I, y \in x_i\}$ for convenience of notation. Note that this induces a membership predicate and subset relation, which are explicitly given by

$$(j, z) \in (x_i)_{i \in I} \iff z \in x_j, \quad (x_i)_{i \in I} \subseteq (y_i)_{i \in I} \iff \forall i \in I. x_i \subseteq y_i$$

where $(x_i)_{i \in I}$ and $(y_i)_{i \in I} \in \prod_{i \in I} \mathbf{U}^\top C_i$, $j \in I$ and $z \in C_j$. Moreover, we obtain two continuous maps \cap, \cup , whose explicit definition reads

$$\diamond : \left(\prod_{i \in I} \mathbf{U}^\top C_i \right)^2 \rightarrow \prod_{i \in I} \mathbf{U}^\top C_i, \quad ((x_i)_{i \in I}, (y_i)_{i \in I}) \mapsto (x_i \diamond y_i)_{i \in I}$$

where $\diamond \in \{\cap, \cup\}$. Note that, domain theoretically, \cap is the least upper bound and \cup gives us the greatest lower bound of two elements of $\prod_{i \in I} \mathbf{U}^\top C_i$. We always consider sub-domains of the extended upper space or the interval domain equipped with the Scott topology.

The symbol \Rightarrow is used for the continuous function space. In particular, for semi rectangles \mathbf{A}, \mathbf{B} , we consider the set $(\mathbf{A} \Rightarrow \mathbf{U}^\top \mathbf{B})$ of functions $f : \mathbf{A} \rightarrow \mathbf{U}^\top \mathbf{B}$ which are continuous with respect to the Euclidean topology on \mathbf{A} and the Scott topology on \mathbf{B} . Similarly, $(\mathbf{U}^\top \mathbf{A} \Rightarrow \mathbf{U}^\top \mathbf{B})$ denotes the set of functions that are continuous w.r.t. the Scott topology on $\mathbf{U}^\top \mathbf{A}$ and $\mathbf{U}^\top \mathbf{B}$; the same conventions apply to the interval domain.

We extend the ordinary arithmetical operations to the extended upper space without further mention. In particular, we write $a \diamond b = \{x \diamond y \mid x \in a, y \in b\}$, where $\diamond \in \{+, -, *, /\}$ and $a, b \in \mathbf{U}^\top \mathbb{R}^n$. (We adopt the standard convention that $a/b = \perp$ if $0 \in b$.)

It is a straightforward exercise to see that Scott continuous functions of type $\mathbf{A} \rightarrow \mathbf{U}^\top \mathbf{B}$ are precisely the semi continuous functions of set-valued analysis [4]. More concretely, we have that $f : \mathbf{A} \rightarrow \mathbf{U}^\top \mathbf{B}$ is Scott continuous, iff

$$\forall x \in \mathbf{A} \forall \epsilon > 0 \exists \delta > 0 \forall x' \in B_\delta(x). f(x') \subseteq f(x) + B_\epsilon$$

where $B_\epsilon(x) = \{x' \in \mathbf{A} \mid \|x - x'\| < \epsilon\}$ and $B_\delta = B_\delta(0)$. Note that we have the Scott continuous *extension mapping*

$$\mathcal{E} : (A \Rightarrow \mathbf{U}^\top B) \rightarrow (\mathbf{U}^\top A \Rightarrow \mathbf{U}^\top B), f \mapsto \lambda x. \bigsqcap_{y \in x} f(y),$$

and it is an easy exercise to show that this greatest lower bound is actually given by direct image, i.e. $\mathcal{E}(f)(x) = \bigcup \{f(y) \mid y \in x\}$.

3 Flows and Flow Automata

We begin our study of hybrid automata by first discussing *flow automata*, where the continuous evolution in every control state is an explicitly given flow function. This will subsequently be shown to be equivalent to the case that the continuous evolution is specified by a vector field in Section 6. For flow automata, every discrete control state comes with a flow function that behaves like the solution of an initial value problem and governs the evolution of the continuous variables in that state. We restrict attention to flows take values in a regular closed set C (i.e. a closed set which is equal to the closure of its interior) that will later correspond to the invariant sets associated with the discrete control states of an automaton.

Definition 1 *If $D \subseteq \mathbb{R}^n \times [0, \infty)$ is a subset, then the support of a vector $x \in \mathbb{R}^n$ is the set $D_x = \{t \in [0, \infty) \mid (x, t) \in D\}$. A flow on a regular closed subset $C \subseteq \mathbb{R}^n$ is a continuous function $f : D \rightarrow C$ defined on a regular closed set D with $C \times \{0\} \subseteq D \subseteq C \times [0, \infty)$ such that for all $s, t \geq 0$ and all $x \in C$*

- (1) $s+t \in D_x$ iff $s \in D_x$ and $t \in D_{f(x,s)}$. In this case $f(x, s+t) = f(f(x, s), t)$.
- (2) the function $f(\cdot, t)$ is injective for all $t \in \mathbb{R}$.
- (3) the partial derivative $\frac{\partial f}{\partial t} : D^\circ \rightarrow \mathbb{R}^n$ exists in the interior D° of D and can be continuously extended to the whole of D .

In the following, we will identify $\frac{\partial f}{\partial t}$ with its continuous extension to D .

That is, a flow $f : D \subseteq \mathbb{R}^n \times [0, \infty) \rightarrow C$ on $C \subseteq \mathbb{R}^n$ behaves like the solution of an initial value problem $f'(t) = v(f(t))$, $f(0) = x$ defined on the support D_x of x , where v is defined on a subset of Euclidean space \mathbb{R}^n . We briefly summarise some of the well known properties of flow functions that we will use later.

Proposition 2 *Suppose $f : D \rightarrow C$ is a flow on $C \subseteq \mathbb{R}^n$.*

- (1) the support $D_x = \{t \in [0, \infty) \mid (x, t) \in D\}$ is a closed interval for every $x \in C$.

- (2) $f(x, 0) = x$ for all $x \in C$
- (3) $\frac{\partial f}{\partial t}(f(x, t), 0) = \frac{\partial f}{\partial t}(x, t)$ whenever $x \in \mathbb{R}^n$ and $t \in D_x$
- (4) The function $f_x = \lambda t.f(x, t) : D_x \rightarrow \mathbb{R}^n$ solves the initial value problem $\dot{f}_x(t) = \frac{\partial f}{\partial t}(f_x(t), 0)$ with initial condition $f_x(0) = x$, where \dot{f}_x denotes taking derivative (w.r.t. time).

For later reference, we note the following corollary, which ensures boundedness of flows on compact rectangles.

Corollary 3 *Suppose $R \subseteq \mathbb{R}^n$ is regular compact and $f : D \subseteq R \times [0, \infty) \rightarrow R$ is a flow on R . Then there exists $K > 0$ s.t. $\|\frac{\partial f}{\partial t}(x, t)\| \leq K$ for all $(x, t) \in D$.*

PROOF. Continuity of $\frac{\partial f}{\partial t}$ and compactness of R implies that $K = \sup\{\frac{\partial f}{\partial t}(x, 0) \mid x \in R\} < \infty$. By Proposition 2, for all $x \in R$, the function $g = f(x, \cdot)$ satisfies $\dot{g}(t) = \frac{\partial f}{\partial t}(g(t), 0)$, hence $\|\dot{g}(t)\| \leq K$ for all $t \in [0, \infty)$. But by definition of g , we have $\dot{g}(t) = \frac{\partial f}{\partial t}(x, t)$ and the result follows, as x was arbitrary.

Flows arise as solutions of initial value problems. In the light of the later developments, we focus on (locally) Lipschitz vector fields defined on a compact subset of \mathbb{R}^n ; note that every locally Lipschitz vector field defined on a compact space is automatically globally Lipschitz.

Lemma 4 *Suppose $v : C \rightarrow \mathbb{R}^n$ is a locally Lipschitz vector field defined on a regular compact subset $C \subseteq \mathbb{R}^n$. If $f(x, \cdot)$ denotes the maximal solution of the initial value problem $f(x, t) = v(f(x, t))$, $f(x, 0) = x$, then f is a flow. We say that f is the flow induced by v .*

PROOF. Follows from the continuous dependence of the solution of an initial value problem on the initial condition and the continuation theorem, see e.g. [5].

We now introduce continuous flow automata.

Definition 5 *A flow automaton in \mathbb{R}^n is a tuple $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{init})$ where*

- Q is a finite set of discrete control states
- $\text{inv} = (\text{inv}(q))_{q \in Q}$ is a family of state invariants where $\text{inv}(q) \subseteq \mathbb{R}^n$ is a regular closed set
- $\text{flow} = (\text{flow}(q))_{q \in Q}$ is a family of flow functions where $\text{flow}(q) : D(q) \subseteq \text{inv}(q) \times [0, \infty) \rightarrow \text{inv}(q)$ is a flow on $\text{inv}(q)$

- $\text{res} = (\text{res}(p, q))_{p, q \in Q}$ is a family of reset relations with $\text{res}(p, q) : \text{inv}(p) \rightarrow \mathcal{P}(\text{inv}(q))$
- $\text{init} = (\text{init}(q))_{q \in Q}$ is a family of initial states with $\text{init}(q) \subseteq \text{inv}(q)$

for all $q \in Q$, resp. $(p, q) \in Q \times Q$. We call a flow automaton compact, if $\text{inv}(q), \text{init}(q) \in \mathbf{U}^\top \mathbb{R}^n$ are compact for all $q \in Q$ and $\text{res}(p, q)(x) \in \mathbf{U}^\top \text{inv}(q)$ is a compact subset of $\text{inv}(q)$ for all $p, q \in Q$ and all $x \in \text{inv}(q)$. A state of a flow automaton is a tuple (q, x) with $q \in Q$ and $x \in \text{inv}(q)$. We write $S_F = \{(q, x) \mid q \in Q, x \in \text{inv}(q)\}$ for the state space of F and $i_F = \{(q, x) \in S \mid x \in \text{init}(q)\}$ for the set of initial states.

Remark 6 The above definition of flow automata, though slightly different, is equivalent to the standard definition given e.g. in [3]. While our control states are in one-to-one correspondence to the control locations of loc.cit., the transitions between control states are modelled in terms of a finite multiset $V \subseteq Q \times Q$ of transitions, and an action predicate $\text{act}(v) \subseteq \mathbb{R}^n \times \mathbb{R}^n$ is assigned to every transition $v \in V$. In this terminology, the automaton can change its state, say from state (q, x) to state (q', x') iff there exists a transition $(q, q') \in V$ with $(x, x') \in \text{act}(v)$. In our terminology, this can be modelled by the reset relation $\text{res}(q, q') = \lambda x. \{y \in \text{inv}(q) \mid \exists (q, q') \in V. (x, y) \in \text{act}(q, q')\}$.

For the remainder of the paper, we assume that all flow automata are compact. Our main interest lies in the comparison of the denotational semantics and the operational semantics of a flow automaton. The latter is given in terms of a labelled transition system, where a label is either a positive real numbers that signifies the duration of a continuous transition or 0, indicating that the automaton is changing its discrete control state.

Definition 7 Suppose $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{init})$ is a flow automaton and let $\Sigma = [0, \infty)$. The transition system T_F associated with F is the tuple (S_F, \rightarrow) , where S_F is the state space of F and $\rightarrow \subseteq S \times \Sigma \times S$ is defined by the following two clauses:

flow transitions $(q, x) \xrightarrow{t} (q', x')$ iff $q = q'$, $t \in D(q)_x$ and $\text{flow}(q)(x, t) = x'$ for $t > 0$

jump transitions $(q, x) \xrightarrow{0} (q', x')$ iff $x' \in \text{res}(q, q')(x)$;

For states $s, s' \in S$, we write $s \xrightarrow{*} s'$ if there is a finite sequence of states s_1, \dots, s_k with $s \xrightarrow{t_1} s_1 \xrightarrow{t_2} \dots \xrightarrow{t_k} s_k = s'$ with $t_1, \dots, t_k \in \Sigma$ and $\sum_{i=1}^k t_k = t$. We write $\text{init} \xrightarrow{*} s$ iff there exists $i \in i_F$ with $i \xrightarrow{*} s$.

An F -trajectory is a finite or infinite sequence $(t_i, q_i, f_i)_{i < N}$ where $N \in \mathbb{N} \cup \{\infty\}$ such that $(t_i)_{i < N}$ is non-decreasing in $[0, \infty)$, $(q_i)_{i < N}$ is a sequence in Q and $f_i : [t_{i-1}, t_i] \rightarrow \text{inv}(q)$ is a function (we use the convention that $t_{-1} = 0$) that, for all $i < N$, satisfies

- $f_0(t_{-1}) \in \text{init}(q_0)$ and $(q_i, f_i(t_{i-1})) \xrightarrow{t} (q_i, f_i(t_{i-1} + t))$ for all $t \in [t_{i-1}, t_i]$
- $(q_i, f_i(t_i)) \xrightarrow{0} (q_{i+1}, f_{i+1}(t_i))$.

We denote the set of possible states of the automaton F at time t by $R_F(t)$ and the set of all states the automaton can visit up to time t by $V_F(t)$, formally defined by

$$R_F(t) = \{s \in S_F \mid \text{init} \xrightarrow{*}^t s\} \quad \text{and} \quad V_F(t) = \bigcup \{R_F(s) \mid s \leq t\}$$

where $t \in [0, \infty)$.

Note that by assumption, $\text{flow}(q_i)(f_i(t_{i-1}), t) = f_i(t_{i-1} + t)$. Compared with the definition of trajectories in [2], it is straightforward to verify that, under the correspondence outlined in Remark 6, our definition of trajectories gives rise to the same semantics.

We now turn to the main issue of the present paper and describe the necessary ingredients needed to perform domain theoretic analysis of a flow automaton F . Our main goal is to define a domain theoretic semantic function $\llbracket F \rrbracket : [0, \infty) \rightarrow \prod_{q \in Q} \mathbf{U}^\top \text{inv}(q)$, where, for a closed semi rectangle $R \subseteq \mathbb{R}^k$, $\mathbf{U}^\top R$ is the *extended upper space* associated with R , that is the dcpo of compact subsets of R , ordered by reverse inclusion. The function $\llbracket F \rrbracket$ associates to every time point $t \in [0, \infty)$ an element of $\prod_{q \in Q} \mathbf{U}^\top \text{inv}(q)$. That is, to every point in time t we associate a family $(s_q)_{q \in Q}$, with $s_q \subseteq \text{inv}(q)$, of compact sets such that $\{(q, x) \mid x \in s_q\} = R_F(t)$. Having computed R_F , it is easy to derive a mechanism for computing the possibly visited states $V_F(t)$ at time t by unfolding the definition of V_F . We demonstrate later that it is also possible to obtain V_F directly as a fixed point.

The goal of the construction is to give a *continuous* semantics of flow automata: if the automaton is *effectively given*, i.e. both flow and res arise as limits of sequences of finitary approximations with $\text{flow} = \bigsqcup_{k \in \mathbb{N}} f_k$ and $\text{res} = \bigsqcup_{k \in \mathbb{N}} r_k$, then we can effectively obtain $\sigma_k : [0, \infty) \rightarrow \prod_{q \in Q} \text{inv}(q)$ such that $\llbracket F \rrbracket = \bigsqcup_{k \in \mathbb{N}} \sigma_k$. This provides us with three important properties:

- (1) The function σ_k is a *conservative approximation* of the semantics of F , for all $k \geq 0$
- (2) The semantics of F can be computed up to an arbitrary degree of accuracy
- (3) The algorithm for computing σ_k can be implemented on a digital computer without loss of precision

Clearly, continuity of the semantics mapping $\llbracket \cdot \rrbracket$ can only be achieved if we restrict attention to flow automata whose components are continuous. This motivates the next definition.

Definition 8 A flow automaton $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{inv})$ is continuous, if

$\text{res}(p, q) : \text{inv}(p) \rightarrow \mathbf{U}^\top \text{inv}(q)$ is Scott continuous for all $p, q \in Q$. We say that F is separated, if

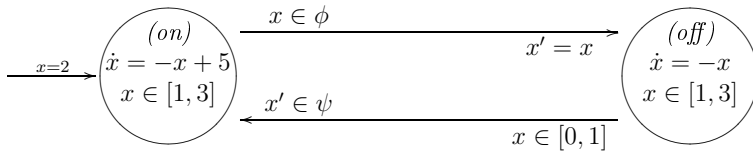
- $x \in \text{res}(p, q)(y)$ implies that $\text{res}(q, r)(x) = \emptyset$ for all $p, q, r \in Q$ and $y \in \text{inv}(p)$
- $x \in \text{init}(q)$ implies that $\text{res}(q, r)(x) = \emptyset$ for all $q, r \in Q$

While the continuity condition on res is clearly enforced by our goal to be able to approximate the semantics of flow automata, the separation condition tells us that there are no transient states, i.e. the automaton cannot perform state changes from q_0 to q_1 , and subsequently from q_1 to q_2 without remaining in state q_1 for a non-zero amount of time.

We will see later that separation and continuity imply that the automaton under scrutiny is non-zeno. While we believe that all of our results can be established even for non-separated automata under the additional assumption that the automata are non-zeno, the main benefit of the separation property is that it is very easy to verify.

For a continuous flow automaton, the family $\text{res}(p, q)_{p, q \in Q}$ induces a generalised IFS on the extended upper spaces of $\text{inv}(p)$, for $p \in Q$, as we will see in Definition 15 later on. The following example discusses the requirements introduced in Definition 8.

Example 9 We consider the following variant F of a thermostat automaton, see e.g. [18]. Let $Q = \{\text{on}, \text{off}\}$ with $\text{inv}(q) = [1, 3]$ for $q = \text{on}, \text{off}$. The flow functions are given by the differential equations $\text{flow}(\text{on})(x_0, \cdot) =$ the unique solution of $\dot{x} = -x + 5, x(0) = x_0$, and similarly, $\text{flow}(\text{off})(x_0, \cdot) =$ the unique solution of $\dot{x} = -x, x(0) = x_0$, with initial state $(\text{on}, 2)$. We fix two subsets $\phi, \psi \subseteq [1, 3]$ and let $\text{res}(\text{on}, \text{off})(x) = \{x\} \cap \phi$. The function $\text{res}(\text{off}, \text{on})$ is given by $x \mapsto \psi$, if $x \in [0, 1]$, and $x \mapsto \emptyset$ otherwise. Graphically, the automaton can be displayed as follows, where x' denotes the value of x after the change of control states.



We now discuss several alternatives for the sets ϕ and ψ , and relate them to continuity of the induced automaton.

- (1) Suppose $\psi = (1, 2)$. Then $\text{res}(\text{off}, \text{on})$ does not take values in $\mathbf{U}^\top [1, 3]$, as $(1, 2)$ is not compact, hence $\text{res}(\text{off}, \text{on})$ is not a well defined function of type $[1, 3] \rightarrow \mathbf{U}^\top [1, 3]$.
- (2) Suppose $\phi = (2, 3]$. Then the F is not continuous, as for $x = 2$ and

$\epsilon > 0$, we fail to find δ s.t. for all $x' \in B_\delta(x)$ we have $\text{res}(\text{on}, \text{off})(x') \in \text{res}(\text{on}, \text{off})(x) + B_\epsilon$.

- (3) If both ϕ and ψ are compact, then F is continuous.
(4) We have that F is separated, iff $\phi \cap [0, 1] = \phi \cap \psi = \emptyset$ and $\phi \cap \{2\} = \emptyset$.

To verify continuity of the reset functions in practise, note that Scott continuity is preserved by function composition, hence all combinations of Scott continuous functions will be Scott continuous. In particular, we note that the following functions are Scott continuous, and thus can be used as building blocks for reset functions.

Proposition 10 Suppose $A, B \in \mathbf{U}^\top \mathbb{R}^n$.

- (1) All step functions

$$a \searrow b : A \rightarrow \mathbf{U}^\top B, \quad x \mapsto \begin{cases} b & x \in a^\circ \\ \perp & \text{otherwise} \end{cases}$$

are continuous for $a \in \mathbf{U}^\top A, b \in \mathbf{U}^\top B$, where a° denotes the interior of a .

- (2) All co-step functions

$$a \swarrow b : A \rightarrow \mathbf{U}^\top B, \quad x \mapsto \begin{cases} b & x \in a \\ \top & \text{otherwise} \end{cases}$$

are continuous for $a \in \mathbf{U}^\top A, b \in \mathbf{U}^\top B$.

- (3) All functions

$$\bowtie b : A \rightarrow \mathbf{U}^\top B, \quad x \mapsto \{x\} \cap b$$

are continuous for $b \in \mathbf{U}^\top B$

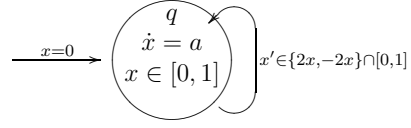
- (4) If $f_1, f_2 : A \rightarrow \mathbf{U}^\top B$ are continuous, then so is $f_1 \cup f_2 : A \rightarrow \mathbf{U}^\top B, x \mapsto f_1(x) \cup f_2(x)$.
(5) If $(f_i)_{i \in I}$ is directed (w.r.t. the pointwise ordering), then $\bigsqcup_{i \in I} f_i : A \rightarrow \mathbf{U}^\top B, x \mapsto \bigsqcup_{i \in I} f_i(x)$ is continuous.

PROOF. Item (3) follows from continuity of the binary join operation on bounded complete domains, for (2) see [14]. All remaining items are standard, see e.g. [15].

The previous proposition gives some general construction principles for continuous hybrid automata, and can be applied to show that a large class of flow automata are actually continuous. We now turn to the separation property. The following example, which is a variation of the bouncing ball automaton [23] shows that the separation property is vital for the computability of the semantic function associated with a flow automaton.

Example 11 Consider the automaton $F = (Q, \text{init}, \text{flow}, \text{res}, \text{inv})$ with

- $Q = \{q\}$
- $\text{inv}(q) = [-1, 1]$
- $\text{flow}(r, t) = r + a \cdot t$
- $\text{res}(x) = \{2x, -2x\} \cap [0, 1]$
- $\text{init}(q) = \{0\}$



where $a \in [-1, 1]$ is a computable real number, as depicted on the right above. Suppose we can effectively find a sequence of functions $R_k : [0, 1] \rightarrow \mathbf{U}^\top[0, 1]$ such that $\bigsqcup_{k \in \mathbb{N}} R_k = R_F$. Then clearly $R(1) = \{0\}$ iff $a = 0$, and $R(1) \cap [1/2, 1] \neq \emptyset$ iff $a \neq 0$. As $R(1) = \bigcap_{k \in \mathbb{N}} R_k(1)$, this implies that we can semi-decide whether $a = 0$. Together with a semi decision procedure for $a \neq 0$, we arrive at a decision procedure for $a = 0$, which is impossible, see e.g. [26].

Recall that a flow automaton is *zeno*, if it admits a trajectory $(t_i, q_i, f_i)_{i < \infty}$ with $\sup_{i < \infty} t_i < \infty$. The key consequence of separation, which makes it possible to compute the semantic function associated with a flow automaton, is that separated automata are non-zeno. This follows from the next proposition.

Proposition 12 Suppose F is separated and continuous. Then there exists $\epsilon > 0$ such that $t_i - t_{i-1} \geq \epsilon$ for all F -trajectories $(t_i, q_i, f_i)_{i < N}$ and all $0 \leq i < N$.

PROOF. We assume $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{init})$. For all $p, q \in Q$, the sets

$$\text{post}_{p,q} = \{x \in \text{inv}(q) \mid \exists y \in \text{inv}(p). x \in \text{res}(p, q)(y)\}$$

and

$$\text{pre}_{p,q} = \text{res}(p, q)^{-1}(\{x \in \mathbf{U}^\top \text{inv}(q) \mid x \neq \top\})$$

are closed, hence compact by boundedness of $\text{inv}_p, \text{inv}_q$. Therefore the sets

$$\text{in}_p = \bigcup_{q \in Q} \text{post}_{q,p} \cup \text{init}_p \quad \text{and} \quad \text{out}_p = \bigcup_{q \in Q} \text{pre}_{p,q}$$

are compact for all $p \in Q$. As F is separated, $\text{in}_p \cap \text{out}_p = \emptyset$ for all $p \in Q$.

Therefore, there exists $\delta > 0$, such that for all $q \in Q$ and all $x, y \in \text{in}_q \times \text{out}_q$ one has $\|x - y\| \geq \delta$.

By Corollary 3, there exists $K > 0$ such that $\frac{\partial \text{flow}(q)}{\partial t}(r, x) \leq K$ for all $q \in Q$ and all $(r, x) \in \text{inv}(q) \times [0, \infty) \cap O(q)$. Put $\epsilon = \delta/K$ and suppose $\rho = (t_i, q_i, f_i)_{i < N}$ is an F -trajectory with $t_{-1} = 0$, as usual. Then, for all $i \geq 0$, we have that $f_i(t_{i-1}) \in \text{in}_{q_i}$ and $f_i(t_i) \in \text{out}_{q_i}$. Hence we have $t_i > t_{i-1}$ and

$$\delta \leq \|f_i(t_{i-1}) - f_i(t_i)\| \leq K \|t_{i-1} - t_i\| \leq K \cdot (t_i - t_{i-1})$$

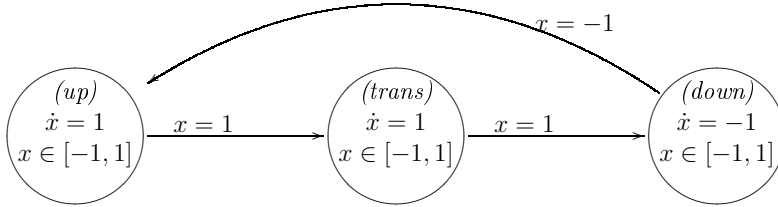
as the function $\text{flow}(q_i)$ is uniformly Lipschitz in t . Therefore $t_i - t_{i-1} \geq \epsilon$ as claimed.

As an immediate corollary, we obtain an easy-to-check sufficient condition for a flow automaton to be non-zeno.

Corollary 13 *Suppose F is separated and continuous. Then F is non zeno.*

PROOF. If F were zeno, then F would have a trajectory $(t_i, q_i, f_i)_{i < \infty}$ where $\sup_i t_i < \infty$, which is impossible, as $t_i - t_{i-1} \geq \epsilon$ with ϵ as in Proposition 12.

Remark 14 *While the fact that an automaton is separated is sufficient for it being non-zeno, the separation property is not necessary. Consider for example the automaton*



with reset relations $\text{res}(\text{up}, \text{trans}) = \text{res}(\text{trans}, \text{down}) = \lambda x. \{x\} \cap \{1\}$ and $\text{res}(\text{down}, \text{up}) = \lambda x. \{x\} \cap \{-1\}$ and initial state $(\text{up}, 0)$. Then clearly F is non-zeno, but F is not separated. This suggests that the separation property can be relaxed, and one just needs to require that there is no finite loop $(q_0, x_0), (q_1, x_1), \dots, (q_l, x_l)$ with $x_{i+1} \in \text{res}_{q_i, q_{i+1}}(x_i)$ and $x_0 \in \text{res}_{q_l, q_0}(x_l)$, but we refrain from doing so, as the technical complications would obscure the techniques at the heart of our analysis.

4 Denotational Semantics of Continuous and Separated Automata

We now turn to the main objective of the present paper and describe a computational method for obtaining the reachable states R_F for a continuous and separated flow automaton F . Our technique will compute the function R_F as least fixpoint of a functional of type $([0, \infty) \Rightarrow \mathcal{U}) \rightarrow ([0, \infty) \Rightarrow \mathcal{U})$, where $\mathcal{U} = \prod_{q \in Q} \mathbf{U}^\top \text{inv}(q)$. We first introduce some terminology to make the notation more readable.

Definition 15 *Suppose $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{init})$ is a flow automaton. The*

function

$$f_F : \mathcal{U} \times [0, \infty) \rightarrow \mathcal{U},$$

$$((x_q)_{q \in Q}, t) \mapsto \{\text{flow}(q)(y_q, t) \mid y_q \in x_q, t \in D(q)_{y_q}\}$$

is called the extended flow function, and

$$r_F : \mathcal{U} \rightarrow \mathcal{U}, (x_q)_{q \in Q} \mapsto \left(\bigcup_{p \in Q} \mathcal{E}(\text{res}(p, q))(x_p) \right)_{q \in Q}$$

is the extended reset function with \mathcal{E} as defined at the end of Section 2. If the automaton F is clear from the context, we omit the corresponding subscript.

Both the extended flow function and the extended reset function collect all flow and reset functions associated with a flow automaton in a single map. It is easy to see that both the extended flow function, and the extended reset function are Scott continuous.

Lemma 16 *If F is a continuous flow automaton, then both f_F and r_F are Scott continuous.*

PROOF. Both are straightforward calculations and follow from the Scott-continuity of the extension mapping, discussed at the end of Section 2. For continuity of the extended reset function, one furthermore needs that of set-theoretic union.

With this notation, we are now ready to introduce the key concept of the present paper: the forward action associated with a flow automaton. As we will see later, the least fixpoint of this operator captures the set of states the automaton can engage in at time t and, moreover, can be effectively computed.

Definition 17 *Suppose F is a flow automaton. The operator*

$$\Phi_F : ([0, \infty) \Rightarrow \mathcal{U}) \rightarrow ([0, \infty) \Rightarrow \mathcal{U}), \rho \mapsto \lambda t. f_F(i_F, t) \cup \bigcup_{s \leq t} f_F(r_F(\rho(s)), t - s)$$

is called the forward action associated with F .

The forward action combines the discrete action and the continuous flow, and can be seen as a generalisation of the fixpoint operator associated with an IFS [8]. Our goal is to show that the least fixpoint of the forward action is precisely the function R_F that computes reachable states. In order to compute this fixpoint effectively, we first have to ensure that Φ_F is compatible with approximations, i.e. Φ_F is well-defined and Scott continuous.

Lemma 18 *Suppose $\rho : [0, \infty) \rightarrow \mathcal{U}$. Then $\Phi_F(\rho) : [0, \infty) \rightarrow \mathcal{U}$ is well defined and Scott continuous.*

PROOF. For well definedness, we have to show that $\Phi_F(\rho)$ actually takes values in \mathcal{U} , that is compact sets. This, and continuity of Φ_F , will follow by representing Φ_F as composition of well-defined and continuous functions.

Now let $\rho \in ([0, \infty) \Rightarrow \mathcal{U})$ and consider $g_\rho : [0, \infty)^2 \rightarrow \mathcal{U}$, defined by $g_\rho(s, t) = f_F(r_F(\rho(s)), t - s)$. Then g_ρ is continuous by the continuity of composition and subtraction. Therefore, also the canonical extension of g_ρ , $\mathcal{E}(g_\rho) : \mathbf{U}^\top [0, \infty)^2 \rightarrow \mathcal{U}$, $S \mapsto \bigcup_{(s,t) \in S} g_\rho(s, t)$ is well defined and Scott continuous. Now consider the continuous function $h : [0, \infty) \rightarrow \mathbf{U}^\top [0, \infty)^2$, given by $h(t) = \{(s, t) \in [0, \infty)^2 \mid s \leq t\}$. It is easy to see that $\Phi_F(\rho) = \mathcal{E}(g_\rho) \circ h$, which shows that $\Phi_F(\rho)$ is well defined and continuous.

Lemma 19 *The operator $\Phi_F : ([0, \infty) \Rightarrow \mathcal{U}) \rightarrow ([0, \infty) \Rightarrow \mathcal{U})$ is continuous.*

PROOF. Suppose $\rho = \bigsqcup_{k \in \mathbb{N}} \rho_k : [0, \infty) \rightarrow \mathcal{U}$. Take g_ρ (resp. g_{ρ_k}) as in the proof of Lemma 18. An easy analysis, using continuity of f_F and r_F , shows that $g_\rho = \bigsqcup_{k \in \mathbb{N}} g_{\rho_k}$. The claim now follows from continuity of the extension function $\mathcal{E} : ([0, \infty)^2 \Rightarrow \mathcal{U}) \rightarrow (\mathbf{U}^\top [0, \infty)^2 \Rightarrow \mathcal{U})$ and the continuity of h .

Continuity of Φ_F now guarantees the existence of a least fixpoint of Φ_F , which we denote by $\llbracket F \rrbracket$ throughout. We now examine this fixpoint and show that it precisely captures the set of all F -trajectories.

In order to show soundness, it is convenient to formulate trajectories as maps into the upper space. In order to turn the trajectories into Scott continuous functions, we let the induced function take a non-singleton set as value whenever the discrete control state changes.

Lemma 20 *Suppose $f : [-1, 0] \rightarrow \mathbb{R}$ and $g : [0, 1] \rightarrow \mathbb{R}$ are continuous. Then the function $f \oplus g : [-1, 1] \rightarrow \mathbf{U}^\top \mathbb{R}$, with*

$$f \oplus g : t \mapsto \begin{cases} \{f(t)\} & \text{if } t < 0 \\ \{f(0), g(0)\} & \text{if } t = 0 \\ \{g(t)\} & \text{if } t > 0 \end{cases}$$

is Scott continuous.

PROOF. Follows immediately from the ϵ - δ characterisation of continuity of maps into the extended upper space.

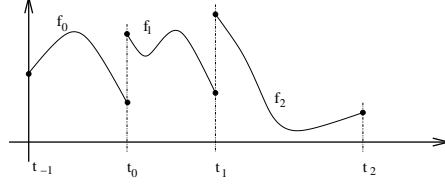


Figure 1. The function ρ^\sharp

For F -trajectories, we have the following corollary. Note that the condition on trajectories is automatic for continuous and separated automata.

Corollary 21 *Suppose F is a flow automaton and $\rho = (t_i, q_i, f_i)_{i < N}$ is a F -trajectory with $\sup_i t_i = \infty$ in case $N = \infty$. Then*

$$\rho^\sharp : [0, \infty) \rightarrow \mathcal{U}, \quad t \mapsto \{(q_i, f_i(t)) \mid i < N, t \in [t_{i-1}, t_i]\}$$

is Scott-continuous. Moreover, $R_F(t) = \bigcup \{\rho^\sharp(t) \mid \rho \text{ is an } F\text{-trajectory}\}$, if F is a flow automaton.

The function ρ^\sharp is visualised in Figure 1. The next statement is a stepping stone for proving the soundness of our approach. We begin by noting that every fixpoint of Φ_F is an over-approximation of the set of all trajectories.

Lemma 22 *Suppose F is separated and continuous, ρ is an F -trajectory and $\sigma = \Phi_F(\sigma)$ is a fixpoint of Φ_F . Then $\Phi_F(\sigma) \sqsubseteq \rho^\sharp$.*

PROOF. Suppose that $\rho = (t_i, q_i, f_i)_{i < N}$ for some $N \in \mathbb{N} \cup \{\infty\}$. We show by induction on i that $\rho^\sharp \upharpoonright [0, t_i] \supseteq \Phi_F(\sigma) \upharpoonright [0, t_i]$. Recall our convention that $t_{-1} = 0$ and note that, by Proposition 12, we have that $t_0 > 0$, hence $\rho^\sharp(0) = \{(q_0, f_0(0))\} \in i_F \subseteq f_F(i_F, 0) \subseteq \Phi_F(\sigma)(0) = \sigma(0)$. Now suppose $i \geq -1$ and let $t \in [t_i, t_{i+1}]$. We show that $\Phi_F(\sigma)(t) \upharpoonright [0, t_{i+1}] \sqsubseteq \rho^\sharp(t) \upharpoonright [0, t_{i+1}]$, i.e. $(q_{i+1}, f_{i+1}(t)) \in \Phi_F(\sigma)(t)$ for all $t \in [t_i, t_{i+1}]$. By induction hypothesis, we have

$$(q_i, f_i(t_i)) \in \sigma(t_i) \tag{1}$$

and the definition of F -trajectories gives

$$f_{i+1}(t_i) \in \text{res}(q_i, q_{i+1})(f_i(t_i)) \quad f_{i+1}(t) = \text{flow}(q_{i+1})(f_{i+1}(t_i), t - t_i) \tag{2}$$

for all $t \in [t_i, t_{i+1}]$. Taken together, Equation (1), combined with the left hand part of (2) give

$$(q_{i+1}, f_{i+1}(t_i)) \in r_F(\sigma(t_i)). \tag{3}$$

In combination with the right hand part of (2) this yields

$$(q_{i+1}, f_{i+1}(t)) \in f_F(r_F(\sigma(t_i)), t - t_i) \subseteq \Phi_F(\sigma)(t) = \sigma(t)$$

which concludes the proof.

Note that the proof of the previous theorem relies on separatedness, as otherwise even the base case of the induction would not work. Using the above result, soundness of the fixpoint construction is immediate:

Corollary 23 (Correctness) *Suppose F is continuous and separated. Then $s \in \llbracket F \rrbracket(t)$ if $\text{init} \rightarrow_*^t s$ for all $s \in S_F$ and all $t \in [0, \infty)$.*

PROOF. Let $s \in S_F$ and assume that $\text{init} \rightarrow_*^t s$. Then there exists an F -trajectory ρ such that $s \in \rho^\sharp(t) \subseteq \llbracket F \rrbracket$ as $\llbracket F \rrbracket$ is a fixpoint of Φ_F .

While the previous result can be read as asserting soundness, we now turn to computational adequacy of the construction, that is we show that $R_F = \llbracket F \rrbracket$, where $\llbracket F \rrbracket$ is the least fixpoint of Φ_F . Moreover, our analysis entails that $\llbracket F \rrbracket$ is the *unique* fixpoint of Φ_F . Both facts are consequences of the following lemma.

Lemma 24 *Suppose σ is a fixpoint of Φ_F . Then $\sigma(t) \subseteq R_F(t)$ for all $t \in [0, \infty)$.*

PROOF. We define, for $q \in Q$, the sets in_q and out_q as in the proof of Proposition 12, and similarly pick δ and K such that for all $q \in Q$

$$\frac{\partial \text{flow}(q)}{\partial t}(r, t) \leq K \text{ and } \inf\{\|x - y\| \mid (x, y) \in \text{in}_q \times \text{out}_q\} \geq \delta$$

for all $r \in \text{inv}(q)$ and all $t \in [0, \infty)$ that satisfy $(r, t) \in O(q)$.

Suppose for a contradiction that there exists $t \in [0, \infty)$ such that $\sigma(t) \not\subseteq R_F(t)$. Let

$$t_0 = \inf\{t \in T \mid \sigma(t) \not\subseteq R_F(t)\}.$$

Let $\epsilon = \frac{\delta}{K}$. By definition of t_0 , we can find $t_1 \in [t_0, t_0 + \epsilon)$ such that $\sigma(t_1) \not\subseteq R_F(t_1)$. Let $(q_1, x_1) \in \sigma(t_1) \setminus R_F(t_1)$. Then, by definition of Φ_F , and the fact that $(q_1, x_1) \notin R_F(t_1)$, we have that

$$(q_1, x_1) \in \bigcup_{s \in [t_0, t_1]} f_F(r_F(\sigma(s)), t_1 - s).$$

Hence we find $s_1 \in [t_0, t_1]$ and $(r_1, y_1) \in \sigma(s_1)$ together with $z_1 \in \text{res}(r_1, q_1)(y_1)$ such that $(q_1, x_1) \in f_F((q_1, z_1), t_1 - s_1)$, i.e. $x_1 = \text{flow}(q_1)(z_1, t_1 - s_1)$. Now $(r_1, y_1) \notin R_F(s_1)$, for otherwise we could construct an F -trajectory that witnesses $(q_1, x_1) \in R_F(t_1)$. By repeating the same argument, we find $s_2 \in [t_0, s_1]$ and $(r_2, y_2) \in \sigma(s_2)$, together with $z_2 \in \text{res}(r_2, r_1)(y_2)$ such that $(r_1, y_1) \in f_F((r_1, z_1), s_1 - s_2)$, i.e. $y_1 = \text{flow}(r_1)(z_2, s_1 - s_2)$. Summing up, we have $|s_1 - s_2| < \epsilon$ and

- $(r_2, y_2) \in \sigma(s_2)$
- $y_1 = \text{flow}(r_1)(z_2, s_1 - s_2)$
- $z_2 \in \text{res}(r_2, r_1)(y_2)$
- $z_1 \in \text{res}(r_1, q_1)(y_1)$

Note that in particular, $z_2 \in \text{in}_{r_1}$ and $y_1 \in \text{out}_{r_1}$, where in and out are as in the proof of Proposition 12. Using the bound K on the derivative of $\text{flow}(q)$ w.r.t. time, we have

$$\delta \leq \|y_1 - z_2\| \leq K|s_1 - s_2|$$

which implies that $\epsilon > |s_1 - s_2| \geq \frac{\delta}{K}$, contradicting our choice of ϵ .

This immediately gives computational adequacy:

Theorem 25 (Computational Adequacy) *Suppose F is separated and continuous. Then $s \in \llbracket F \rrbracket(t)$ iff $\text{init} \xrightarrow{*}^t s$ for all $s \in S_F$ and all $t \geq 0$.*

PROOF. From Corollary 23 we already have $R_F(t) \subseteq \llbracket F \rrbracket(t)$ for all $t \geq 0$, and an application of Lemma 24 yields the converse inclusion.

The proof of the theorem in fact demonstrates that any function $\rho \in ([0, \infty) \Rightarrow \mathcal{U})$ with $\rho \sqsubseteq \llbracket F \rrbracket$ which does not arise as an F -trajectory, necessarily leads to a violation of the separatedness property. As it turns out, the least fixpoint $\llbracket F \rrbracket$ of Φ_F is actually unique.

Corollary 26 *The operator Φ_F has a unique fixpoint.*

PROOF. Suppose $\sigma : [0, \infty) \rightarrow \mathcal{U}$ is a fixpoint of Φ_F . As $\llbracket F \rrbracket$ is the least such, we have $\llbracket F \rrbracket \sqsubseteq \sigma$. Together with Lemma 24, this implies

$$\sigma(t) \subseteq \llbracket F \rrbracket(t) = R_F(t) \subseteq \sigma(t)$$

for all $t \in [0, \infty)$, hence $\sigma = \llbracket F \rrbracket$.

Unfolding the definition of V_F , we also obtain computational means to obtain the states of a flow automaton F that can be visited up to time t in terms of the least fixpoint $\llbracket F \rrbracket$ of the forward action Φ_F associated with F . This then gives $V_F(t) = \bigcup_{s \leq t} R_F(s)$. However, we can also obtain V_F as a fixpoint of an operator in its own right.

Definition 27 *The operator*

$$\Psi_F : ([0, \infty) \Rightarrow \mathcal{U}) \rightarrow ([0, \infty) \Rightarrow \mathcal{U}), \rho \mapsto \overline{f_F}(\text{init}, [0, t]) \cup \bigcup_{s \leq t} f_F(r_F(\rho(s)), [0, t-s])$$

where $\overline{f}_F : \mathcal{U} \times \mathbf{I}[0, \infty) \rightarrow \mathcal{U}$, $(x, \alpha) \mapsto \prod_{t \in \alpha} f_F(x, t)$ is the canonical extension of f_F to time intervals, is the visited states operator associated with F .

The properties of Ψ_F are similar to those of Φ_F , in particular, Ψ_F is Scott continuous, and the least fixpoint captures the set of visited states.

Theorem 28 *Suppose $\rho : [0, \infty) \rightarrow \mathcal{U}$ is the least fixpoint of Ψ_F . Then $\rho = V_F$.*

PROOF. Similar to the proof of Lemma 18 and Lemma 19, one checks that $\Psi_F(\rho)$, for $\rho : [0, \infty) \rightarrow \mathcal{U}$, and Ψ_F itself, are Scott continuous. Lemma 22 remains valid, if we replace Φ_F by Ψ_F , and ρ^\sharp by ρ^\flat , where $\rho^\flat(t) = \bigcup_{s \leq t} \rho^\sharp(s)$. Note that ρ^\flat can be formulated in terms of the extension function \mathcal{E} , and is hence Scott continuous. Finally, the proof of Theorem 25 can be repeated almost verbatim.

While Theorem 25 and Theorem 28 are important on their own, as they allow us to obtain the semantics of hybrid automata as a least fixpoint in a suitable function space, they also allow us to derive new results about the function R_F that yields the states reachable at time t for continuous and separated automata:

Corollary 29 (1) $R_F(t)$ and $V_F(t)$ are compact for every $t \in [0, \infty)$.
 (2) R_F and V_F are Scott continuous.

PROOF. This is because both R_F and V_F arise as least fixpoints of a Scott continuous functional that takes only compact sets as values.

5 Approximation of Flow Automata

In the previous section, we have seen that the semantics $\llbracket F \rrbracket : [0, \infty) \rightarrow \mathcal{U}$ of a flow automaton F can be computed as the least fixpoint of a functional on $([0, \infty) \Rightarrow \mathcal{U})$. While this gives a mathematical means of understanding the semantics, we now show that this also induces a method to compute the semantics up to an arbitrary degree of accuracy.

To do this, we restrict attention to countable bases of the involved domains, that is to finitely representable objects that generate all of the involved domains by means of directed suprema. We show, that we can effectively compute the least fixpoint of the functional up to an arbitrary degree of accuracy, if

we approximate all continuous ingredients of the automaton. We begin by introducing the bases of the domains we are interested in. For the remainder of the section, we fix a countable dense ordered subring $D = \{d_0, d_1, \dots\}$ with decidable equality and order and computable ring operations. We put $D_k = \{d_0, \dots, d_k\}$. We only treat the case of computing R_F as a least fixpoint; the setup can be easily adapted to accommodate also V_F .

Definition 30 *We let, for an arbitrary set $S \subseteq \mathbb{R}$, $\mathbf{IR}_S^n = \{[a_1, b_1] \times \dots \times [a_n, b_n] \in \mathbb{R}^n \mid a_1, \dots, a_n, b_1, \dots, b_n \in S\} \cup \{\mathbb{R}\}$ denote the set of rectangles with endpoints in S , augmented with the least element \mathbb{R} . If $A \subseteq \mathbb{R}^n$ is a semi rectangle, then $\mathbf{IA}_S = \{A \cap b \mid b \in \mathbf{IR}_S^n\}$ denotes the set of rectangles $R \in \mathbf{IR}^n$ that are contained in A and have corners in S , again with a bottom element. We distinguish two different kinds of step functions:*

$$a \searrow^i b : A \rightarrow B, x \mapsto \begin{cases} b & x \in a^\circ \\ \perp & \text{otherwise,} \end{cases} \quad \text{and} \quad a \searrow b : A \rightarrow B, x \mapsto \begin{cases} b & a \ll x \\ \perp & \text{otherwise} \end{cases}$$

where B is a dcpo with $b \in B$ in both cases; $A \subseteq \mathbb{R}^n$ is a semi rectangle with $a \in \mathbf{IA}$ in the case of $a \searrow^i b$, and A is a dcpo with $a \in A$ for $a \searrow b$. We use the following bases:

- (1) If $A \subseteq \mathbb{R}^n$ is a semi-rectangle with corners in $D \cup \{\pm\infty\}$, then the set \mathbf{IA}_D of rectangles contained in A having corners in D together with A itself, is called the standard base of \mathbf{IA} ; the standard base of $\mathbf{I}^\top A$ is $\mathbf{IA}_D \cup \{\top\}$.
- (2) If $A \in \mathbf{IR}_D^n$, then the set $\mathbf{U}^\top A_D = \{\cup_{1 \leq i \leq k} D_i \mid i \in \mathbb{N}, D_i \in \mathbf{IA}_D\}$ of finite unions of rectangles with corners in D is the rectangular base of $\mathbf{U}^\top A$.
- (3) If A_D and B_D are bases of the dcpos A and B , respectively, then $(A \Rightarrow B)_D = \{\sqcup_{1 \leq i \leq k} a_i \searrow b_i \in (A \Rightarrow B) \mid a_1, \dots, a_k \in A_D, b_1, \dots, b_k \in B_D\}$ is the induced base of $(A \Rightarrow B)$.
- (4) Finally, if $A \subseteq \mathbb{R}^n$ is a semi rectangle with corners in $D \cup \{\pm\infty\}$ and B_D is a base of the dcpo B , then $(A \Rightarrow B)_D = \{\sqcup_{1 \leq i \leq k} a_i \searrow^i b_i \in (A \Rightarrow B) \mid a_1, \dots, a_k \in \mathbf{IA}_D, b_1, \dots, b_k \in B_D\}$ is the induced base of $(A \Rightarrow B)$.

where we indicate by $\sqcup_{1 \leq i \leq k} a_i \searrow b_i \in (A \Rightarrow B)$ that we consider only consider consistent step functions [10, Section 2], similarly for $\sqcup_{1 \leq i \leq k} a_i \searrow^i b_i$.

In words, if $A, B \subseteq \mathbb{R}^n$ are semi-rectangles, \mathbf{IA}_D is the set of rectangles contained in A with corners in D and $\mathbf{U}^\top A_D$ is the set of finite unions of rectangles with corners in D . For the function space, $(A \Rightarrow \mathbf{U}^\top B)_D$ is the induced base of the space of functions of one or more real variables; $(\mathbf{U}^\top A \Rightarrow \mathbf{U}^\top B)_D$ is the induced base of the function space of a compact set valued variable.

It is easy to see that the sets introduced above are indeed bases of the corresponding domain. We now use these bases to show that the fixpoint operator Φ_F associated to a flow automaton can be effectively computed, given approximations of the components of the automaton. In order to make assertions

about the computability of functions in the domain theoretic model of computation, we have to fix an enumeration of the base of the involved domains. We do not do this explicitly here, but instead assume that all of the bases $(\cdot)_D$ above come with an effective enumeration $\iota : \mathbb{N} \rightarrow (\cdot)_D$, which we fix throughout the discussion. In particular, the enumeration gives rise to a notion of *effective sequence*: If \mathbf{A} is a depo whose base is enumerated via $\iota : \mathbb{N} \rightarrow \mathbf{A}_D$, then a sequence $(a_k)_{k \in \mathbb{N}}$ in \mathbf{A}_D is *effective*, if $a_k = \iota(f(k))$ for some total recursive function f .

First, note that composition of base functions yields a base function, and that the extension function is effectively computable.

Lemma 31 *Suppose $f \in (\mathbf{A} \Rightarrow \mathbf{U}^\top \mathbf{B})_D$ and $g \in (\mathbf{U}^\top \mathbf{B} \Rightarrow \mathbf{U}^\top \mathbf{C})_D$. Then*

- (1) $g \circ f \in (\mathbf{A} \Rightarrow \mathbf{U}^\top \mathbf{C})_D$ and $g \circ f$ is effectively computable.
- (2) $\mathcal{E}(f) \in (\mathbf{U}^\top \mathbf{A} \Rightarrow \mathbf{U}^\top \mathbf{B})_D$ and $\mathcal{E}(f)$ is effectively computable.

PROOF. The first item is straightforward, as

$$\left(\bigsqcup_{i \in I} a_i \searrow b_i\right) \circ \left(\bigsqcup_{j \in J} c_j \searrow d_j\right) = \bigsqcup_{h \in H} \left\{ \left(\bigsqcup_{h \in H} c_h\right) \searrow b_i \mid H \subseteq I \text{ finite, } \bigsqcup_{h \in H} d_h \ll a_i \right\}.$$

The second item is as in [10, Section 2].

The next lemma gives a basis representation of subtraction, which is needed in the definition of the fixpoint functional Φ_F associated with F , see Definition 17.

Lemma 32 *The functions $M_k : [0, \infty)^2 \rightarrow \mathbf{I}[0, \infty)$, defined by $M_k = \bigsqcup \{a \times b \searrow b - a \mid a, b \in \mathbf{I}[0, \infty)_{D_k}\}$ satisfy $M_k \in ([0, \infty)^2 \Rightarrow \mathbf{I}[0, \infty))_D$ for all $k \in \mathbb{N}$, and $\bigsqcup_{k \in \mathbb{N}} M_k = \lambda(x, y).y - x$.*

Building on these basic facts, we can now show that the least fixpoint of the operator Φ_F associated with a flow automaton is effectively computable. This of course hinges on the fact that the automaton is effectively given. In this context, we identify an interval valued function $f : D \rightarrow \mathbf{IC}$ defined on a closed set $D \subseteq \mathbb{R}^m$ that takes values in a compact set $C \subseteq \mathbb{R}^n$ with the function

$$\bar{f} : \mathbb{R}^n \rightarrow \mathbf{I}^\top C, x \mapsto \begin{cases} f(x) & x \in D \\ \top = \emptyset & x \notin D \end{cases}$$

that takes values in the extended interval domain $\mathbf{I}^\top C = \mathbf{IC} \cup \{\emptyset\}$. The next lemma justifies this identification.

Lemma 33 *Suppose $D \subseteq \mathbb{R}^n$ is closed and $f : D \rightarrow \mathbf{IC}$ is continuous. Then so is \bar{f} .*

The proof uses the fact that D is closed, and is straightforward. In particular, this allows us to view a flow on a compact set C that is defined on a regular closed subset of $\mathbb{R}^n \times [0, \infty)$ as a function of type $\mathbb{R}^n \times [0, \infty) \rightarrow \mathbf{I}^\top C$. We can now define effectively given flow automata as follows.

Definition 34 *Suppose $F = (Q, \text{init}, \text{flow}, \text{res}, \text{inv})$ is a flow automaton. We say that F is effectively given if $\text{inv}(q) \in \mathbf{IR}_D^n$ for all $q \in Q$ and F comes with*

- an effective sequence $(i_k^q)_{k \in \mathbb{N}}$ in $(\mathbf{U}^\top \text{inv}(q))_D$ with $\bigsqcup_{k \in \mathbb{N}} i_k^q = \text{init}(q)$
- an effective sequence $(f_k^q)_{k \in \mathbb{N}}$ in $(\mathbb{R}^n \times [0, \infty) \rightarrow \mathbf{I}^\top (\text{inv}(q)))$ such that $\bigsqcup_{k \in \mathbb{N}} f_k^q = \text{flow}(q)$
- an effective sequence $(r_k^{p,q})_{k \in \mathbb{N}}$ in $(\text{inv}(p) \Rightarrow \mathbf{U}^\top \text{inv}(q))_D$ with $\bigsqcup_k r_k^{p,q} = \text{res}(p, q)$

for all $q \in Q$, resp. all $(p, q) \in Q^2$. The family of sequences $(f_k^q)_{k \in \mathbb{N}}$, $(O_k^q)_{k \in \mathbb{N}}$, $(i_k^q)_{k \in \mathbb{N}}$ (where $q \in Q$) and $(r_k^{p,q})_{k \in \mathbb{N}}$ (where $(p, q) \in Q^2$) are called an effective presentation of F .

That is to say that, for an effectively given flow automaton, the initial states, the flow functions and the reset functions are computable. It is easy to see that every effectively given flow automaton induces a computable extended flow function, and a computable extended reset function. Since the forward action associated with a flow automaton computes differences of time points, which – at each finite step of the computation – are only known approximatively, we need to extend the type of the approximating flow function in such a way that it accepts interval values in the second clause below.

Lemma 35 *Suppose F is an effectively given flow automaton.*

- (1) *We can effectively construct an increasing sequence $(\hat{i}_k) \in \mathcal{U}_D$ with $\bigsqcup_{k \in \mathbb{N}} \hat{i}_k = i_F$.*
- (2) *We can effectively construct an increasing sequence $(\hat{f}_k) \in (\mathcal{U} \times \mathbf{I}[0, \infty) \Rightarrow \mathcal{U})_D$ with $\bigsqcup_{k \in \mathbb{N}} \hat{f}_k(x, \{t\}) = f_F(x, t)$ for all $x \in \mathcal{U}$ and all $t \in [0, \infty)$.*
- (3) *We can effectively construct an increasing sequence $(\hat{r}_k) \in (\prod_{q \in Q} \mathbf{U}^\top \text{inv}(q) \Rightarrow \prod_{q \in Q} \mathbf{U}^\top \text{inv}(q))_D$ with $\bigsqcup_{k \in \mathbb{N}} \hat{r}_k = r_F$.*

PROOF. The first claim is immediate, the second and the third are an application of Lemma 31.

Lemma 36 *Suppose F is effectively given, (\hat{i}_k) , (\hat{r}_k) and (\hat{f}_k) are as in Lemma 35 and $\rho_k = \lambda t. \hat{f}_k(\hat{i}_k, t)$. Then $\rho_k \in ([0, \infty) \Rightarrow \mathcal{U})_D$ is effectively computable and $\bigsqcup_{k \in \mathbb{N}} \rho_k = \lambda t. \text{flow}(\text{init}, t)$.*

PROOF. Suppose that $\hat{f}_k = \bigsqcup_{i \in I} a_i \times b_i \searrow c_i$. We know that \hat{f}_k is effectively computable by Lemma 35. Then $\rho_k = \bigsqcup \{b_i \searrow c_i \mid i \in I, a_i \ll \hat{i}_k\}$.

We now turn to the second part of the fixpoint functional.

Lemma 37 *Suppose F is effectively given and take (\hat{f}_k) and (\hat{r}_k) as in Lemma 35. If $\rho \in ([0, \infty) \Rightarrow \mathcal{U})_D$ and*

$$\Delta_k(\rho) = \lambda(s, t). \hat{f}_k(\hat{r}_k(\rho(s), M_k(s, t))) \text{ and } \Sigma_k(\rho) = \lambda t. \mathcal{E}(\Delta_k)(\{t\}, [0, t]),$$

where M_k is defined as in Lemma 32, then both $\Sigma_k(\rho)$ and $\Delta_k(\rho)$ are effectively computable, and $\Sigma_k(\rho) \in ([0, \infty) \Rightarrow \mathcal{U})_D$.

PROOF. Computability of Δ_k follows from Lemma 31. Now suppose that $\Delta_k(\rho) = \bigsqcup_{i \in I} a_i \times b_i \searrow c_i$. Then $\Sigma_k(\rho) = \bigsqcup \{a_i \searrow c_i \mid b_i \ll [0, \bar{a}_i]\}$, where $a_i = [\underline{a}_i, \bar{a}_i]$.

We can now compute $\llbracket F \rrbracket$ as $\bigsqcup_k \sigma_k$ by putting $\sigma_0 = \lambda t. \perp$, where \perp is the least element of \mathcal{U} , and $\sigma_{k+1} = \lambda t. \rho_k(t) \cup \Sigma_k(\sigma_k)(t)$, where ρ_k is as in Lemma 36. This is the content of the following theorem.

Theorem 38 *Suppose F is an effectively given flow automaton. Then we can effectively obtain a sequence (σ_k) with $\bigsqcup_{k \in \mathbb{N}} \sigma_k = \llbracket F \rrbracket$.*

PROOF. We put $\sigma_0 = \lambda t. \perp$, where \perp is the least element of \mathcal{U} , and

$$\sigma_{k+1} = \lambda t. \rho_k(t) \cup \Sigma_k(\sigma_k)(t)$$

where ρ_k is as in Lemma 36. Then σ_{k+1} can be effectively obtained from σ_k by Lemma 36 and Lemma 37. We still have to show that $\llbracket F \rrbracket = \bigsqcup_{k \in \mathbb{N}} \sigma_k$. By definition of Φ_F , we have, by the domain theoretic fixpoint theorem that $\llbracket F \rrbracket = \bigsqcup_{k \in \mathbb{N}} \Phi_F^k(\lambda t. \perp)$. Recall that

$$\Phi_F(\rho) = \lambda t. f_F(i_F, t) \cup \mathcal{E}(\lambda(s, t). f_F(r_F(\rho(s), t - s)))(\{t\}, [0, t]).$$

Using the fact that i_F , f_F and r_F are effectively approximated by (\hat{i}_k) , (\hat{f}_k) and (\hat{r}_k) , respectively, the above can be re-written as

$$\Phi_F(\rho) = \lambda t. \left(\bigsqcup_k \hat{f}_k(\hat{i}_k, t) \cup \mathcal{E}(\lambda(s, t). \bigsqcup_k \hat{f}_k(\hat{r}_k(\rho(s), M_k(t, s))) \right)(\{t\}, [0, t]),$$

which, by continuity, amounts to $\Phi_F = \bigsqcup_{k \in \mathbb{N}} \Phi_k$, where

$$\Phi_k(\rho) = \lambda t. f_k(\hat{i}_k, t) \cup \mathcal{E}(\lambda(s, t). \hat{f}_k(\hat{r}_k(\rho(s), M_k(t, s))) \right)(\{t\}, [0, t]).$$

But this just means that $\sigma_0 = \perp$ and $\sigma_{k+1} = \Phi_k(\sigma_k)$, which implies, again by continuity, that $\llbracket F \rrbracket = \sqcup_k \sigma_k$, which was what we had to show.

While this puts us into a position to effectively compute an increasing sequence of sets converging to the reachable states of a flow automaton at any time point, the following example shows that the convergence is not effective in general in a natural metric.

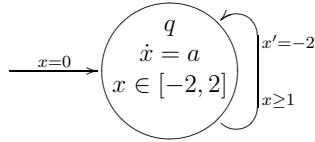
Example 39 *This example shows that, for an effectively given flow automaton F , the convergence $\llbracket F \rrbracket(t) = \sqcup_{k \in \mathbb{N}} \sigma_k(t)$ is not effective in general if we measure the convergence speed in the Hausdorff metric, given by*

$$d_H(C, D) = \max\left\{\sup_{c \in C} \inf_{d \in D} \|c - d\|, \sup_{d \in D} \inf_{c \in C} \|c - d\|\right\}$$

for two compact sets $C, D \subseteq \mathbb{R}^n$, where $d_H(\emptyset, \emptyset) = 0$ and $d_H(C, D) = \infty$ if either C or D (but not both) are empty. Alternatively, the Hausdorff distance between two compact subsets of \mathbb{R}^n is characterised by

$$d_H(C, D) = \max\{\epsilon \geq 0 \mid B_\epsilon(C) \subseteq D \text{ and } B_\epsilon(D) \subseteq C\}$$

where the ϵ -ball around a compact set C is given by $B_\epsilon(C) = \{x \in \mathbb{R}^n \mid \exists y \in C. \|x - y\| \leq \epsilon\}$. Now consider the following automaton, where $a \in [0, 2]$ is a computable real number:



Theorem 38 provides us with computable sequence $(R_k)_{k \in \mathbb{N}}$ such that $\sqcup_{k \in \mathbb{N}} R_k = R(1)$ equals the set of reachable states at time $t = 1$. It is straightforward to verify that $a \geq 1 \iff -1 \in R(1)$. If the convergence $R(1) = \sqcup_{k \in \mathbb{N}} R_k$ were effective in the Hausdorff metric on the space $\mathcal{U} = \mathbf{U}^\top[-2, 2]$, we could computably determine $k \in \mathbb{N}$ such that $d_H(R(1), R_k) \leq \frac{1}{2}$. We obtain

$$a \geq 1 \iff -1 \in R(1) \iff -1 \in R_k$$

and, since $-1 \in R_k$ can be effectively determined, a decision procedure for $a \geq 1$. By the dual construction, we can decide $a \leq 1$, resulting in a decision procedure for $a = 1$, which is impossible, see for example [26].

In other words, we cannot effectively determine the convergence speed, and hence the complexity, of the algorithm underlying Theorem 38.

6 Hybrid Automata

In this section, we transfer our results on flow automata to hybrid systems, where the continuous behaviour of the system in every given control state is described by a vector field. This is achieved by associating the equivalent flow automaton to the hybrid automaton under consideration. If the hybrid automaton is effectively given, we show that the same also holds for the induced flow automaton. We thus obtain an effective framework for the analysis of hybrid automata. The following is a variant of the standard definition of a hybrid automaton [16,2].

Definition 40 *A hybrid automaton is a tuple $H = (Q, \text{inv}, \text{vect}, \text{res}, \text{init})$ where $Q, \text{inv}, \text{res}, \text{init}$ are as in Definition 5, and $\text{vect} = (\text{vect}_q)_{q \in Q}$ is a family of vector fields $\text{vect}(q) : \text{inv}(q) \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ where each $\text{vect}(q)$ is locally Lipschitz, i.e. every $z \in \text{inv}(q)$ has a neighbourhood $N \subseteq V(q)$ such that $\|\text{vect}(q)(x) - \text{vect}(q)(y)\| \leq L\|x - y\|$, for all $x, y \in N$ and some $L \in \mathbb{R}$.*

In contrast to the standard definition, the trajectories of the real variables are described by a differential equation rather than differential inclusion. We require this restriction in view of the domain theoretic treatment of differential equations [12], which in general gives a strict over-approximation to the solution of a differential inclusion.

We recall from Lemma 4 that every Lipschitz vector field $v : C \rightarrow \mathbb{R}^n$ defined on a regular compact subset $C \subseteq \mathbb{R}^n$ induces a flow function $f : O \subseteq C \times [0, \infty) \rightarrow \mathbb{R}^n$. Replacing the vector field by the induced flow function, every hybrid automaton H induces a flow automaton F ; in this case, we write $\llbracket H \rrbracket$ for $\llbracket F \rrbracket$.

Definition 41 *Suppose $H = (Q, \text{inv}, \text{vect}, \text{res}, \text{init})$ is a hybrid automaton and $\text{flow}(q)$ is the flow induced by $\text{vect}(q)$. The automaton $F = (Q, \text{inv}, \text{flow}, \text{res}, \text{init})$ is called the flow automaton induced by H . We say that H is continuous (resp. separated), if the induced flow automaton is continuous (resp. separated). We say that H is effectively given if it comes with*

- an effective sequence $(i_k^q)_{k \in \mathbb{N}}$ in $(\mathbf{U}^\top \text{inv}(q))_D$ with $\sqcup_{k \in \mathbb{N}} i_k^q = \text{init}(q)$
- an effective sequence $(v_k^q)_{k \in \mathbb{N}}$ in $(\text{inv}(q) \Rightarrow \mathbf{I}\mathbb{R}^n)_D$ with $\sqcup_{k \in \mathbb{N}} v_k^q = \text{vect}(q)$
- an effective sequence $(r_k^{p,q})_{k \in \mathbb{N}}$ in $(\text{inv}(p) \Rightarrow \mathbf{U}^\top \text{inv}(q))_D$ with $\sqcup_k r_k^{p,q} = \text{res}(p, q)$

for all $q \in Q$, resp. all $(p, q) \in Q^2$ and $\text{inv}(q) \in \mathbf{U}^\top \mathbb{R}_D^n$ for all $q \in Q$. The family of sequences $(i_k^q)_{k \in \mathbb{N}}$, $(v_k^q)_{k \in \mathbb{N}}$ (where $q \in Q$) and $(r_k^{p,q})_{k \in \mathbb{N}}$ (where $(p, q) \in Q^2$) are called an effective presentation of H .

We have seen in Theorem 38 that the function $\llbracket F \rrbracket$ associated with a flow automaton, which captures the states reachable by F at time $t \in [0, \infty)$, is effectively computable, if F is effectively given. In order to associate an effectively given flow automaton to an effectively given hybrid automaton, we therefore have to produce approximations $f_k \in (\prod_{q \in Q} \text{inv}(q) \times [0, \infty) \Rightarrow \mathcal{U})$ of the flow function induced by a hybrid automaton. In other words, we have to solve the initial value problems defined by the vector field that defines the hybrid automation. This is achieved by instantiating results from [12,13], where it is shown how to solve initial value problems in a domain theoretic framework; however, we have to adapt these results to deal with the upper space. We recall the main result on domain theoretic solutions of initial value problems, formulated for (globally) Lipschitz vector fields defined on the whole of \mathbb{R}^n that we will adapt to the present setting later.

As before, $D \subseteq \mathbb{R}$ is a dense subring with effective ring operations and decidable ordering.

Theorem 42 *For every given $u \in (\mathbb{I}\mathbb{R}^n \rightarrow \mathbb{I}\mathbb{R}^n)_D$, we can effectively compute $y^u \in ([0, \infty) \Rightarrow \mathbb{I}\mathbb{R}^n)_D$ such that the following holds: If $u = \bigsqcup_{k \in \mathbb{N}} u_k$ is an extension of a real valued and Lipschitz vector field $v : \mathbb{R}^n \rightarrow \mathbb{R}^n$, then $y = \bigsqcup_{k \in \mathbb{N}} y^{u_k}$ is real valued and satisfies $\dot{y} = v(y), y(0) = 0$.*

Note that the restriction on the initial value $y(0)$ is not essential, as every initial value problem with initial condition $y(0) = c$ can be translated into an equivalent problem with initial condition $y(0) = 0$. Using this translation technique, we can now show that the flow function associated with a Lipschitz vector field is computable. This generalises the corresponding result in [9] to dimension $n > 1$. In a nutshell, the next proposition shows that the domain theoretic solution of initial value problems also puts us into the position to construct the flows induced by the vector fields. Notationally, we use currying for convenience, i.e. if $f = \bigsqcup_{j \in J} a_j \searrow^i b_j : [0, \infty) \rightarrow \mathbb{I}\mathbb{R}^n$ is a step function, we write $c \searrow^i f$ for the function $\bigsqcup_{j \in J} c \times a_j \searrow^i b_j$. Recall that we use the pointwise extension of arithmetical operations to elements of $\mathbb{I}\mathbb{R}^n$, in particular $c + d = \{x + y \mid (x, y) \in c \times d\}$ for $c, d \in \mathbb{I}\mathbb{R}^n$.

Proposition 43 *Suppose $u = \bigsqcup_{k \in \mathbb{N}} u_k$ is an extension of a Lipschitz vector field $v : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Then*

$$f = \bigsqcup_{k \in \mathbb{N}} f_k \text{ with } f_k = \bigsqcup_{c \in (\mathbb{I}\mathbb{R}^n)_{D_k}} c \searrow^i y^{u_k(+c)} + c : \mathbb{R}^n \times [0, \infty) \rightarrow \mathbb{I}\mathbb{R}^n$$

with $y^{(\cdot)}$ as in Theorem 42 is real valued and equals the flow associated with v . Moreover, if (u_k) is an effective sequence, then so is (f_k) .

PROOF. Clearly (f_k) is monotone, and it will follow from showing that $f = \sqcup_{k \in \mathbb{N}} f_k$ equals the flow associated with v that the f_k are actually well defined.

First suppose that f is the flow associated with v . We show that $c \searrow^i y^{u_k(\cdot+c)} + c \sqsubseteq f$ for any $c \in \mathbb{I}\mathbb{R}^n$; this will establish $\sqcup_{k \in \mathbb{N}} f_k \sqsubseteq f$. Let $r \in \mathbb{R}^n$ and $t \geq 0$. The case $r \notin c^\circ$ is trivial, as $c \searrow^i y^{u_k(\cdot+c)} + c(r)(t) = \perp \sqsubseteq f(r, t)$, so assume $r \in c^\circ$. Pick an increasing sequence $(c_k) \in \mathbb{I}\mathbb{R}^n$ with $r = \sqcup_{k \in \mathbb{N}} c_k$ and $c_0 = c$. Then $\sqcup_{k \in \mathbb{N}} u_k(\cdot + c_k) = v(\cdot + r)$, hence $y = \sqcup_{k \in \mathbb{N}} y^{u_k(\cdot+c_k)}$ is real valued and satisfies $\dot{y} = v(y + c)$ and $y(0) = 0$, whence $z = y + r$ satisfies $z = f(r, \cdot)$, where we recall that f denotes the flow associated with v . Now $y^{u_k(\cdot+c)} + c \sqsubseteq y + r \sqsubseteq f(r, \cdot)$ as claimed.

We now show that $\sqcup_{k \in \mathbb{N}} f_k$ is real valued. Together with $f \sqsubseteq \sqcup_{k \in \mathbb{N}} f_k$ this will imply the overall claim, as f only takes real values. Let $c \in \mathbb{R}^n$ and let $k_0 \geq 0$ be big enough so that $c \in d^\circ$ for some $d \in \mathbb{I}\mathbb{R}_{D_k}^n$. Pick a sequence $(c_k)_{k \geq k_0}$ such that $c_k \in \mathbb{I}\mathbb{R}_{D_k}^n$ and $\sqcup_{k \in \mathbb{N}} c_k = c$. Then $\sqcup_{k \geq k_0} u_k(\cdot + c_k) = v(\cdot + c)$, hence $\sqcup_{k \geq k_0} y^{u_k(\cdot+c_k)} + c_k \sqsubseteq \sqcup_{k \geq k_0} f_k(c, \cdot)$ and the claim follows, as $\sqcup_{k \geq k_0} y^{u_k(\cdot+c_k)} + c_k$ is real valued by construction.

That is to say, if a vector field is effectively given, so is the induced flow. We now adapt this result to vector fields defined on a compact subset of \mathbb{R}^n that will later be instantiated with the (compact) invariant set $\text{inv}(q)$ obtained from a hybrid automaton.

Proposition 44 *Suppose $C \subseteq \mathbb{R}^n$ is compact and $v : C \rightarrow \mathbb{R}^n$ is a Lipschitz vector field. If $(u_k)_{k \in \mathbb{N}}$ is an effective sequence in $(C \Rightarrow \mathbb{I}\mathbb{R}^n)$ with $v = \sqcup_{k \in \mathbb{N}} u_k$ then we can construct an effective sequence $(\bar{u}_k)_{k \in \mathbb{N}}$ in $(\mathbb{R}^n \Rightarrow \mathbb{I}\mathbb{R}^n)$ such that $\sqcup_{k \in \mathbb{N}} \bar{u}_k = \bar{v}$ for an extension $\bar{v} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ of v that is globally Lipschitz.*

PROOF. We adapt the classical extension theorem for Lipschitz functions [6] to the interval valued setting. If L is the Lipschitz constant of v , we can extend every function $w : C \rightarrow \mathbb{I}\mathbb{R}^n$ with $w \sqsubseteq v$ the function $\bar{w} : \mathbb{R} \rightarrow \mathbb{I}\mathbb{R}^n$ whose i -th component is given by

$$\bar{w}_i : \mathbb{R}^n \rightarrow \mathbb{I}\mathbb{R}, x \mapsto [\sup_{y \in C} u_i^-(y) - L\|x - y\|, \inf_{y \in C} u_i^+(y) + L\|x - y\|]$$

where $u_i = [u_i^-, u_i^+]$. It follows from [6] that \bar{v} is a Lipschitz extension of v to \mathbb{R}^n and it is easy to see that for $w \sqsubseteq v$ we have $\bar{w} \sqsubseteq \bar{v}$ so \bar{w} is well defined. We show that \bar{w} is Scott-continuous for $w \sqsubseteq v$, which will amount to showing that \bar{w}_i^+ (resp. \bar{w}_i^-) is upper (resp. lower) semi-continuous. Let $\epsilon > 0$ and $x_1 \in \mathbb{R}^n$. We can find $y \in C$ such that $\bar{w}_i^+(x_1) \geq w_i^+(y) + L\|x_1 - y\| - \epsilon$. Then $\bar{w}_i^+(x_1) - \bar{w}_i^+(x_2) \leq L\|x_1 - x_2\| + \epsilon$ which implies upper semi-continuity of \bar{w}_i^+ ; the case of \bar{w}_i^- is analogous. Similarly one shows that $\sqcup_{k \in \mathbb{N}} \bar{u}_k = \bar{v}$; if

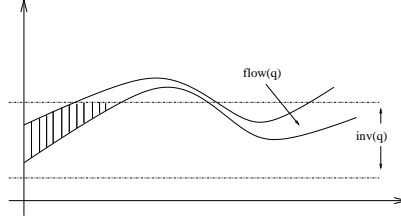


Figure 2. Restriction of the global flow function

$w \in (C \rightarrow \mathbf{IR}^n)_D$ then the sup taken in the definition of \bar{w} ranges over a finite set which implies that the construction of $(\bar{u}_k)_{k \in \mathbf{N}}$ can be made effective.

We are now in the position to turn an effective presentation of a hybrid automaton into an effective presentation of the associated flow automaton, which amounts to computing the maximally defined solution of the initial value problem associated with a vector field v defined on a compact subset $C \subseteq \mathbb{R}^n$. This is achieved in two steps: first, we compute an everywhere defined solution of the initial value problem given by the Lipschitz extension $\bar{v} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and then restrict the ensuing flow to values in C . Graphically speaking, we need to cut out those portions of the flows that leave or re-enter a state invariant so that we are left with the shaded area in Figure 2

Theorem 45 *Suppose H is an effectively given Hybrid automaton. Then so is the associated flow automaton F . Moreover, we can construct an effective presentation of F from an effective presentation of H .*

PROOF. We only have to show that we can effectively construct the flow functions. So suppose $C \in (\mathbf{U}^\top \mathbb{R}^n)_D$, $v : C \rightarrow \mathbb{R}^n$ is a Lipschitz vector field and $(v_k)_{k \in \mathbf{N}}$ is an effective sequence in $(C \Rightarrow \mathbf{IR})$ with $\bigsqcup_{k \in \mathbf{N}} v_k = v$. We are going to construct an effective sequence $(f_k)_{k \in \mathbf{N}}$ such that $f_k \in (C \times [0, \infty) \Rightarrow \mathbf{I}^\top C)$ with $\bigsqcup_{k \in \mathbf{N}} f_k = f$ where f is the flow induced by v ; note that this employs extending f to a function of type $C \times [0, \infty) \rightarrow \mathbf{I}^\top C$ as described in Lemma 33.

Proposition 44 allows us to obtain an effective sequence $(u_k)_{k \in \mathbf{N}}$ in $(\mathbb{R}^n \Rightarrow \mathbf{IR}^n)$ such that $\bigsqcup_{k \in \mathbf{N}} u_k$ is a Lipschitz extension of v to the whole of \mathbb{R}^n . Hence we can apply Proposition 43 to the sequence of maximal extensions of the u_k to obtain an effective sequence $(f_k)_{k \in \mathbf{N}}$ so that each f_k and the supremum $f = \bigsqcup_{k \in \mathbf{N}} f_k$ have type $\mathbb{R}^n \times [0, \infty) \rightarrow \mathbf{IR}^n$. Let $D_k = \{(x, t) \in C \times [0, \infty) \mid f_k(x, t) \cap C \neq \emptyset \text{ for all } 0 \leq s \leq t\}$. Since $f_k \in (\mathbb{R}^n \times [0, \infty) \Rightarrow \mathbf{IR}^n)_D$ is a step function, D_k can be effectively obtained, and moreover monotonicity of the f_k

imply that the D_k are decreasing w.r.t \subseteq . Now define

$$g_k : C \times [0, \infty) \rightarrow \mathbf{I}^\top C, x \mapsto \begin{cases} f_k(x) \cap C & x \in D_k \\ \emptyset & \text{otherwise} \end{cases}$$

It follows that $f = \bigsqcup_{k \in \mathbb{N}} f_k$ is the flow associated with the vector field v .

Together with Theorem 38, we have now shown that the semantic function $\llbracket H \rrbracket$, associated with an effectively given hybrid automaton, is computable.

Theorem 46 *Suppose H is effectively given, continuous and separated. Then the function $\llbracket H \rrbracket : [0, \infty) \rightarrow \mathcal{U}$ is effectively computable.*

Moreover, as all our constructions are based on bases of the domains involved, the algorithms underlying Theorems 46 and 38 are based on proper data types, and can be directly implemented on a digital computer: we choose the dyadic (or rational) numbers for D , and then define data types that directly represent the bases $[0, \infty)_D$ and \mathcal{U}_D , as well as the bases of the function space $([0, \infty) \Rightarrow \mathcal{U})_D$. Computing with dyadic (or rational) numbers then allows us to manipulate elements of the data types without any loss of arithmetical precision. Moreover, we have shown that the fixpoint operator that gives rise to the semantic function $\llbracket H \rrbracket$ of a hybrid automaton, can be effectively computed on the described data types.

7 Conclusions and Future Work.

Of course, much remains to be done. While the presentation in this paper is geared towards demonstrating that domain theory can be used to facilitate the algorithmic analysis of hybrid automata, we anticipate that major improvements will be made on the efficiency of the involved algorithms. In particular, we are working towards conditions that ensure effective convergence and estimates of the convergence speed and the complexity of the described fixpoint algorithms in terms of the Hausdorff distance in \mathcal{U} .

For now, we have concentrated on computing the semantic function $\llbracket H \rrbracket$ associated with a hybrid automaton. Future work will bring a framework for computing the set of reachable states of a hybrid automaton, and a real time logic with associated model checking procedure for the automated verification of hybrid automata.

References

- [1] S. Abramsky and A. Jung. Domain Theory. In S. Abramsky, D. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 3. Clarendon Press, 1994.
- [2] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoret. Comp. Sci.*, 138(1):3–34, 1995.
- [3] R. Alur, T. Henzinger, and P.-H. Ho. Automatic symbolic verification of embedded systems. *IEEE Trans. Softw. Eng.*, 22(3):181–201, 1996.
- [4] J.-P. Aubin. *Viability Theory*. Birkhäuser, 1991.
- [5] E. A. Coddington and N. Levinson. *Theory of Ordinary Differential Equations*. McGraw-Hill, 1955.
- [6] J. Czipser and L. Gehér. Extension of functions satisfying a Lipschitz condition. *Acta Math. Hung.*, 6(1–2), 1955.
- [7] A. Edalat. Dynamical systems, measures and fractals via domain theory. *Inform. and Comput.*, 120(1):32–48, 1995.
- [8] A. Edalat. Power domains and iterated function systems. *Inform. and Comput.*, 124:182–197, 1996.
- [9] A. Edalat, M Krznarić, and A. Lieutier. Domain-theoretic solution of differential equations (scalar fields). In *Proceedings of MFPS XIX*, volume 83 of *Elect. Notes in Theoret. Comput. Sci.*, 2004.
- [10] A. Edalat and A. Lieutier. Domain theory and differential calculus (functions of one variable). *Math. Struct. Comp. Sci.*, 14, 2004.
- [11] A. Edalat and D. Pattinson. A domain theoretic account of Euler’s method for solving initial value problems. In *Proceedings PARA 2004*, number 3732 in *Lect. Notes in Comp. Sci.*, pages 112–121, 2004.
- [12] A. Edalat and D. Pattinson. A domain theoretic account of Picard’s theorem. In *Proc. ICALP 2004*, number 3142 in *Lect. Notes in Comp. Sci.*, pages 494–505, 2004.
- [13] A. Edalat and D. Pattinson. Domain theoretic solutions of initial value problems for unbounded vector fields. In M. Escardó, editor, *Proc. MFPS XXI*, volume 155 of *Electr. Notes in Theoret. Comp. Sci.*, pages 565–581, 2005.
- [14] T. Erker, M. Escardò, and K. Keimel. The way-below relation of function spaces over semantic domains. *Topol. Appl.*, 89(1–2):61–74, 1998.
- [15] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. Scott. *Continuous Lattices and Domains*. Cambridge University Press, 2003.

- [16] T. Henzinger. The theory of hybrid automata. In M. Inan and R. Kurshan, editors, *Verification of Digital and Hybrid Systems*, volume 170 of *NATO ASI Series F: Computer and Systems Sciences*, pages 265–292. Springer Verlag, 2000.
- [17] T. Henzinger, P.-H. Ho, and H. Wong-Toi. HYTECH: A model checker for hybrid systems. *Internat. J. on Softw. Tools for Tech. Transf.*, 1(1–2):110–122, 1997.
- [18] T. Henzinger, P.-H. Ho, and H. Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE T. Automat. Contr.*, 43:540–554, 1998.
- [19] T. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi. Beyond HYTECH: Hybrid systems analysis using interval numerical methods. In *Proc. HSCC 2000*, volume 1790 of *Lect. Notes in Comp. Sci.*, pages 130–144. Springer, 2000.
- [20] J. E. Hutchinson. Fractals and self-similarity. *Indiana U. Math. J.*, 30:713–747, 1981.
- [21] J. Lygeros, D. Godbole, and S. Sastry. Verified hybrid controllers for automated vehicles. *IEEE T. Automat. Contr.*, 43(4):522–539, 1998.
- [22] O. Müller and T. Stauner. Modelling and verification using linear hybrid automata – a case study. *Math. Comp. Model. Dyn.*, 6(1):71–89, 2000.
- [23] S. Simic, K. Johansson, S. Sastry, and J. Lygeros. Towards a geometric theory of hybrid systems. In N. Lynch and B. Krogh, editors, *Proc. HSCC 2000*, volume 1790 of *Lect. Notes in Comp. Sci.*, pages 421–436, 2000.
- [24] C. Tomlin, G. Pappas, and S. Sastry. Conflict resolution for air traffic management : A study in multi-agent hybrid systems. *IEEE T. Automat. Contr.*, 43(4):509–521, 1998.
- [25] P. Varaiya. Smart cars on smart roads: problems of control. *IEEE T. Automat. Contr.*, 38(2), 1993.
- [26] K. Weihrauch. *Computable Analysis*. Springer, 2000.