Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Computability of finite-time reachable sets for hybrid systems

Pieter Collins† and J. Lygeros‡

†Centrum voor Wiskunde en Informatica,
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, *Pieter.Collins@cwi.nl*

‡Department of Electrical and Computer Engineering,
University of Patras, GR-26500, Patras, Greece, *lygeros@ee.upatras.gr*

***Abstract*— In this paper we consider the computability of the evolution of hybrid systems, or equivalently, the computability of finite-time reachable sets. We use the framework of type-two computability theory and computable analysis, which gives a theory of computation for points, sets and maps by Turing machines, and is related to computable approximation. We show that, under suitable hypotheses, the system evolution may be lower or upper semicomputable, but cannot be both in the presence of grazing contact with the guard sets.**

***Index Terms*— Computable analysis; Hybrid system; Reachable set.**

I. Introduction

In the past decade, the study of hybrid systems has attracted considerable attention in the control theory and computer science communities. Hybrid systems are dynamic systems which involve the interaction of discrete and continuous dynamics. Hybrid system models are particularly useful in the area of embedded control, where digital devices are used to control an analogue environment.

Hybrid systems are of considerably higher complexity than the analogous continuous or discrete systems. Even for the relatively innocuous class of piecewise-affine systems, the dynamics exhibits nonlinear characteristics. Worse, the presence of tangential contact with the guard sets governing the reset relations, and the possibility of multiply-enabled discrete transition, means that the system evolution may depend discontinuously on the initial conditions, even on a finite time interval. This discontinuous dependence on initial conditions suggests that the accurate simulation of hybrid systems may be difficult, or even impossible.

In this paper, we consider the computation of reachable sets over finite time intervals, and for finitely many discrete transitions. This includes the problem of computing simulations for deterministic hybrid systems. We show that for correctly formulated problems, the reachable set may be lower-semicomputable or upper-semicomputable, but is not computable in general. Computability of the reachable set for discrete-time systems was considered in [1], and viable and invariant sets in [2]. Computation of reachable and control sets for continuous-time systems have been considered by Puri, Varaiya and Borkar [3] and Szolnoki [4]. Upper-semicontinuity of solutions of *impulse differential inclusions*, was considered in [5].

We consider computations using type-two Turing machines, which work with infinite input and output tapes. The computable analysis used here is based on the work of Weihrauch [6]. The framework is inherently based upon approximation. Positive results on computability show that it is possible to compute approximations to desired output using only approximations to the inputs. Negative results show that it is impossible to compute approximations to the output if the only usable information about the input is approximations. Uncomputability in this framework does not necessarily imply uncomputability in some algebraic framework in which the system and sets of interest can be specified exactly [7]. Related work on computable analysis includes the texts [8], [9], [10]. We stress that all computations are performed on Turing machines (and hence can be implemented using existing computers), unlike the computability theory of [11] in which computations are performed using machines which can store arbitrary real numbers (and cannot be implemented by existing hardware). Many of the set-theoretic operators of computable analysis have been implemented in the software package GAIO [12].

The paper is organised as follows. In Sec. II, we give a brief introduction to the results of computable analysis which we require, including computability results for multivalued mappings. In Sec. III we discuss the computability of solutions of differential inclusions. In Sec. IV, we discuss the computability of reachable sets for hybrid systems in which the guard sets are general open or closed sets. In Sec. V, we consider computability of reachable sets of hybrid systems in which the guard sets are specified as codimension-1 submanifolds. Finally, we give some conclusions and directions for future research in Sec. VI.

II. Computable analysis

A. Computable topological spaces

Throughout this section, Σ will denote a finite alphabet.

Definition 1: A *computable topological space* is a quadruple (M, τ, σ, ν) such that M is a non-empty set, $\tau \subset \mathcal{P}(M)$ is a topology on M, $\sigma \subset \tau$ is a countable sub-base of τ, and $\nu :\subset \Sigma^* \to \sigma$ is a bijective partial function.

Definition 2: A *representation* of a set M is a partial surjective function $\delta :\subset \Sigma^\omega \to M$.

Definition 3: The *standard representation* $\delta_{\mathbf{S}}$ of a computable topological space $\mathbf{S} = (M, \tau, \sigma, \nu)$ is given by

$$\delta_{\mathbf{S}}(p) = x :\Longleftrightarrow \{\nu(w) : w \lhd p\} = \{J \in \sigma : x \in J\}.$$

0-7803-9568-9/05/$20.00 ©2005 IEEE

Here, $w \lhd p$ means that p is a list of words w_i separated by blanks, and $w = w_i$ for some i. Informally, we think of the standard representation δ of (M, τ, σ, ν) as encoding a sequence $(J_i)_{i\in\mathbb{N}}$ containing all sets $J_i \in \sigma$ for which $x \in J_i$.

Definition 4: For $i = 0, \ldots, k$, let X_i be a set, and $\delta_i :\subset \Sigma^\omega \to X_i$ a representation of X_i. Then a function $f : X_1 \times \cdots \times X_k \to X_0$ is $(\delta_1, \ldots, \delta_k; \delta_0)$-computable if there is a Turing machine computing a function $\mathcal{M} :\subset \Sigma^\omega \times \cdots \times \Sigma^\omega \to \Sigma^\omega$ such that $f(\delta_1(w_1), \ldots \delta_k(w_k)) = \delta_0(\mathcal{M}(w_1, \ldots, w_k))$.

The main result of computable analysis is that a computable function is continuous.

Theorem 5: For $i = 0, \ldots, k$ let $\mathbf{S}_i = (M_i, \tau_i, \sigma_i, \nu_i)$ be a computable topological space, and δ_i the standard representation of $\mathbf{S}_i$. Then every $(\delta_1, \ldots, \delta_k; \delta_0)$-computable function $f : M_1 \times \cdots \times M_k \to M_0$ is $(\tau_1, \ldots, \tau_n; \tau_0)$-continuous.

B. Representations of Euclidean space

A computable topological space (X, τ, β, ν) is a *computable Hausdorff space* if X is a locally-compact Hausdorff space, and β is a base for τ. If $J \in \beta$, then J is a *basic (open) set* and $\overline{J}$ is a basic compact set. If $X = \mathbb{R}^n$, then the basic sets can be chosen to fit the application or numerical methods used; typical choices are cuboids, spheres, ellipsoids, parallelepipeds, simplexes or convex polyhedra, and each of these choices leads to the same notion of computability.

The standard representation ρ of a computable Hausdorff space is equivalent to a representation which encodes a sequence of basic sets $(J_i)_{i\in\mathbb{N}}$ such that $\overline{J}_{i+1} \subset J_i$ and $\{x\} = \bigcap_{i=1}^{\infty} J_i$. For the real numbers $\mathbb{R}$, this give the *interval representation*, where a point is represented by a nested sequence of intervals with rational endpoints.

We can describe continuous functions $\mathbb{R}^m \to \mathbb{R}^n$ using the *compact-open representation*. A δ^{co}-name of a continuous function f encodes a sequence of pairs $(\overline{I}, J)$, where I is a basic compact set and J a basic open set, such that $f(\overline{I}) \subset J$.

C. Representations of sets

Let $\mathcal{O}$ denote the open subsets, $\mathcal{A}$ the closed subsets and $\mathcal{K}$ the compact subsets of Euclidean space $\mathbb{R}^n$. The standard representations of these spaces are based on the topologies of lower and upper convergence, and are defined as follows.

Definition 6 (Representations of sets):

1) A $\theta_<$-name for $U \in \mathcal{O}$ encodes a list of all basic compact sets $\overline{I}$ such that $\overline{I} \subset U$.
2) A $\psi_<$-name for $A \in \mathcal{A}$ encodes a list of all basic open sets J such that $J \cap A \neq \emptyset$.
3) A $\psi_>$-name for $A \in \mathcal{A}$ encodes a list of all basic compact sets $\overline{I}$ such that $\overline{I} \cap A = \emptyset$.
4) A $\kappa_>$-name for $C \in \mathcal{K}$ encodes a list of all tuples of basic open sets $(J_1, \ldots, J_k)$ such that $C \subset \bigcup_{i=1}^{k} J_i$.

A ψ-name for $A \in \mathcal{A}$ encodes both a $\psi_<$-name and a $\psi_>$-name of A, and a κ-name for $C \in \mathcal{K}$ encodes both a $\psi_<$-name and a $\kappa_>$-name of C.

We will also need to consider the basic operations of intersection, union and negation on closed and open sets.

Theorem 7:

1) $U \mapsto \mathrm{cl}(U)$ on $\mathcal{O}$ is $(\theta_<; \psi_<)$-computable.
2) $(A, B) \mapsto A \cup B$ on $\mathcal{A}$ is $(\psi_<, \psi_<; \psi_<)$-computable, $(\psi_>, \psi_>; \psi_>)$-computable and $(\psi, \psi; \psi)$-computable.
3) $(U, V) \mapsto U \cap V$ on $\mathcal{O}$ is $(\theta_<, \theta_<; \theta_<)$-computable.
4) $(A, B) \mapsto A \cap B$ on $\mathcal{A}$ is $(\psi_>, \psi_>; \psi_>)$-computable, but not $(\psi, \psi; \psi_<)$-computable.
5) $(C, A) \mapsto C \cap A$ on $\mathcal{K} \times \mathcal{A}$ is $(\kappa_>, \psi_>; \kappa_>)$-computable.
6) $(U, A) \mapsto \mathrm{cl}(U \cap A)$ on $\mathcal{O} \times \mathcal{A}$ is $(\theta_<, \psi_<; \psi_<)$-computable.
7) $(A, U) \mapsto A \setminus U$ on $\mathcal{A} \times \mathcal{O}$ is $(\psi_>, \theta_<; \psi_>)$-computable.

The lack of lower-semicomputability of $A \cap B$ is one of the major sources of difficulty in computing the evolution of a hybrid system, since it makes it impossible to check whether the reachable set has nonempty intersection with a closed guard set at any stage of the computation. To work around this difficulty, we instead take open guard sets, and use the lower-semicomputability of $\mathrm{cl}(A \cap U)$ to obtain lower-semicomputability.

D. Semicontinuous functions

We now consider semicontinuous multivalued functions $F : X \rightrightarrows Y$. There are two natural set-valued preimages of a multivalued function, the *weak preimage* $F^{-1}(B) = \{x \in X : F(x) \cap B \neq \emptyset\}$, and the *strong preimage*, $F^{\Leftarrow}(B) = \{x \in X : F(x) \subset B\}$. We say F is *lower-semicontinuous* if $F^{-1}(U)$ is open whenever U is open, or equivalently, if $F^{\Leftarrow}(A)$ is closed whenever A is closed. F is *upper-semicontinuous* if $F^{-1}(A)$ is closed whenever A is closed, or equivalently, if $F^{\Leftarrow}(U)$ is open whenever U is open. A multivalued function is *continuous* if it is both lower-semicontinuous and upper-semicontinuous. If F is lower-semicontinuous, then so is the map $x \mapsto \mathrm{cl}(F(x))$, and for any set B, $\mathrm{cl}(F(B)) = \mathrm{cl}(F(\mathrm{cl}(B)))$.

We have the following representations for multivalued maps, where I, J and J_i denote basic open sets:

Definition 8:

1) A $\mu_<^\psi$-name of closed-valued lower-semicontinuous F encodes a list of all pairs $(\overline{I}, J)$ such that $\overline{I} \subset F^{-1}(J)$.
2) A $\mu_>^\kappa$-name of compact-valued upper-semicontinuous F encodes a list of all tuples $(\overline{I}, J_1, \ldots, J_k)$ such that $\overline{I} \subset F^{\Leftarrow}(\bigcup_{i=1}^{k} J_i)$.

A μ^κ-name of compact-valued continuous F encodes both a $\mu_<^\psi$-name and a $\mu_>^\kappa$-name of F.

As shown in [1], the set-image operator is computable:

Theorem 9:

1) $(F, A) \mapsto \mathrm{cl}(F(A))$ is $(\mu_<^\psi, \psi_<; \psi_<)$-computable.
2) $(F, C) \mapsto F(C)$ is $(\mu_>^\kappa, \kappa_>; \kappa_>)$-computable and $(\mu^\kappa, \kappa; \kappa)$-computable.

E. Denotable elements and approximation

The standard representations of sets are particularly useful for theoretical analysis, since they are closely related to the topology. For actual computation, we are more interested in computing approximations to the set of interest.

Definition 10: A compact set C is *denotable* if C is a finite union of basic compact sets, $C = \bigcup_{i=1}^{k} \overline{I}_i$. A lower-semicontinuous map F is *denotable* if $\mathrm{Graph}(F) = \bigcup_{j=1}^{k} J_j \times \overline{K}_j$.

As discussed in [1, Section 3], we can compute denotable approximations to sets and maps from the standard representation, and the standard representation from convergent sequences of denotable sets or maps.

If the basic sets are cuboids or rectangles in Euclidean space, we say C is a *rectangular* set, and F a *rectangular* map. It is clear that if C and F are rectangular, then so is $F(C)$, and can be computed exactly. By approximating arbitrary sets and maps by rectangular ones, we can lower- or upper- approximate the set-image.

III. Differential inclusions

A differential inclusion is a generalisation of a differential equation in which the right-hand side may be multivalued. Consider the differential inclusion $\dot{x} \in F(x)$, where $F : X \rightrightarrows \mathrm{T}X$ on $X = \mathbb{R}^n$. A solution to the differential inclusion $\dot{x} \in F(x)$, where $F : X \rightrightarrows \mathrm{T}X$, is an absolutely continuous function $x : [0, T] \to X$ such that $\dot{x}(t) \in F(x(t))$ almost everywhere on $[0, T]$. There is a considerable literature on the existence of solutions for a given initial condition; for an introduction see [13], [14].

The flow of a differential inclusion within a domain D is given by the multivalued map

$$\Phi_T^{F,D}(x_0) = \{x(T) \mid x(\cdot) \text{ is a solution of } \dot{x} \in F(x) \text{ with } x(0) = x_0 \text{ and } x(s) \in D \ \forall s \in [0, T]\},$$

and satisfies $\Phi_{T_1+T_2}^{F,D} = \Phi_{T_1}^{F,D} \circ \Phi_{T_2}^{F,D}$ for all $T_1, T_2 \geqslant 0$.

Computability of solutions of a differential inclusion has been studied by Puri, Varaiya and Borkar [3]. The following theorem is a reformulation of [3, Theorem 3.3] in the language of computable analysis:

Theorem 11: If F is Lipschitz continuous with compact convex values and a known Lipschitz constant L, the set C is compact, then $(F, C, T) \mapsto \Phi_T^{F,X}(C)$ is $(\mu^\kappa, \kappa, \rho; \kappa)$-computable.

The proof is based on computation of Euler steps of the differential inclusion. If $C \subset \mathbb{R}^n$ is compact, $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is lower-semicontinuous and closed-valued, define $C + hF = \{x + hv \mid x \in C \text{ and } v \in F(x)\}$. If C and F are rectangular, then so is $C + hF$, and can be computed exactly. The result follows on taking a sequence of approximations $F_n \to F$, $C_n \to C$, and $T_n \to T$, and considering the convergence of $C_{n,n}$, where $C_{n,0} = C_n$ and $C_{n,i+1} = C_{n,i} + (T_n/n)F_n$.

From Theorem 11, we can compute the reachable set for the time interval $\mathcal{T} = [T_1, T_2]$, since the map $(F, C, T_1, T_2) \mapsto \Phi_{[T_1,T_2]}^{F,X}(C)$ is $(\mu^\kappa, \kappa, \rho, \rho; \kappa)$-computable by Theorem 9.

IV. Hybrid systems

In the case of discrete-time and continuous-time systems, the finite-time evolution is usually computable. Uncomputability only occurs when considering the infinite-time evolution. However, for hybrid-time systems, discontinuities in the solution can occur due, for example, to grazing contact with guard sets. It is therefore important to either find classes of hybrid systems for which the solution varies lower- or upper-semicontinuously with respect to the initial state set, or to give new solution concepts which are lower- or upper-semicontinuous.

A class of hybrid systems for which the solution is an upper-semicontinuous function of the initial state set was given by Aubin et al. [5]. In this section we will show that solutions of this class of system are indeed upper-semicomputable, and find a class of system with lower-semicomputable solutions.

A. Definition of a hybrid system

The following definitions are adapted from [5] and [15].

Definition 12: A hybrid system, H, is a tuple $H = (X, G, D, R, F)$, where $X = \mathbb{R}^n$ is the *state space*, $G \subset \mathrm{dom}(R) \subset X$ is the *guard set*, $D \subset \mathrm{dom}(R) \subset X$ is the *domain* of the flow, $R : \mathrm{dom}(R) \rightrightarrows X$ is the *reset map* and $F : \mathrm{dom}(F) \rightrightarrows \mathrm{T}X$ defines a *flow*.

In this paper we will restrict attention to finite solutions of hybrid systems. A *hybrid time domain* τ is a collection of intervals $\{\tau_i = [t_i, t_{i+1}]\}_{i=0}^{N}$.

Definition 13: A solution x of a hybrid system, (X, G, D, R, F) on a hybrid time domain τ is a sequence of functions $x = \{x_i(\cdot)\}_{i=0}^{N}$, with $x_i(\cdot) : [t_i, t_{i+1}] \to X$ that satisfy:

- *Discrete Evolution*: for $i = 0, 1, \ldots, N-1$, $x_i(t_{i+1}) \in G$ and $x_{i+1}(t_{i+1}) \in R(x_i(t_{i+1}))$
- *Continuous Evolution*: for all i with $t_i < t_{i+1}$, $x_i(\cdot)$ is a solution to the differential inclusion $\dot{x} \in F(x)$ over the interval $[t_i, t_{i+1}]$, with $x_i(t) \in D$ for all $t \in [t_i, t_{i+1}]$.

For a hybrid system $H = (X, G, D, R, F)$, a set of initial states $X_0 \subset X$, a real interval $\mathcal{T} = [T_1, T_2] \subset \mathbb{R}^+$ and an interval of natural numbers $\mathcal{N} = [N_1, N_2] \subset \mathbb{Z}^+$, we define the reachable set by

$$\mathrm{Reach}_{(\mathcal{T},\mathcal{N})}(X_0, H) = \{x_n(t) \mid n \in \mathcal{N},\ t \in \mathcal{T} \text{ and } x \text{ a solution of } H \text{ with } x_0(0) \in X_0\}$$

with closure $\mathrm{clReach}_{(\mathcal{T},\mathcal{N})}(X_0, H)$.

B. Discontinuity of evolution

Hybrid-time dynamics naturally introduces discontinuities of the solution set with respect to initial conditions. Discontinuity can arise from three main sources:

- Grazing contact of the guard set by the continuous flow.
- Grazing contact of the boundary of the domain set by the continuous flow.
- Discontinuities of the reset map R.

Discontinuities of the first type arise when the solution of the differential inclusion comes tangent to the guard set, G. If G is closed, the solution may make a discrete transition at the tangency point, whereas there may exist solutions arbitrarily nearby that do not enter the guard set and therefore cannot make a discrete transition. Similarly, if G is open, then the solution cannot make a discrete transition at the tangency point, whereas arbitrarily close solutions may.

Discontinuities of the second type arise when the solution of the differential inclusion comes tangent to the boundary of the domain D. Continuous evolution is then blocked on one side of the tangency, and allowed to continue on the other.

Discontinuities of the third type arise when the guard sets corresponding to two different discrete transitions intersect. Roughly speaking, in this case, both discrete transitions are possible and one actually occurs nondeterministically [16].

These discontinuities are often intrinsic in the system and can be difficult or impossible to eliminate. Even though the set of initial conditions for which such discontinuities occur is typically a set of zero measure [17], eliminating such discontinuities completely requires one to restrict the class of systems considered quite severely [16]. Discontinuities in the evolution induce a lack of computability of the solution. Fortunately, as we shall show in this paper, it may still be possible to obtain semicontinuous evolution, and semicomputable solutions.

C. *Uncomputability of the* Reach *operator*

To show that Reach is not in general computable, we provide simple counter-examples for which it is not continuous.

Example 14: Consider a hybrid system with a two-dimensional continuous state space. The reset map R takes (x,y) to $(x,y+1)$, and the flow F is given by $(\dot{x},\dot{y})=(1,2x)$, with solution $(x,y)=(x_0+t,t^2+2x_0t+y_0)$. Consider the initial state $(-1,s)$ and the set $Y(s)$ of reachable y-values after time 2. It is clear that elements of $Y(s)$ are of the form $s+n$, where n is the number of discrete transitions.

First consider the open domain $D=\{(x,y)\mid y>0\}$ and the guard set $G=\{(x,y)\mid y<1\}$. If $0<s<1$, then the solution leaves D and is forced to reset, and later re-enters G, so may reset again. Hence $Y(s)=\{s+1,s+2\}$. If $s=1$, the solution leaves D and is forced to reset, but never re-enters G, so $Y(s)=\{s+1\}$. If $1<s<2$, then the solution remains in D, but enters G, so may reset, but is not forced to, so $Y(s)=\{s,s+1\}$. If $2\leqslant s$, then the solution never enters G, and $Y(s)=\{s\}$. In all cases, $Y(s)$, and hence Reach, is not upper-semicontinuous at $s=1,2$.

Now consider the closed domain $D=\{(x,y)\mid y\geqslant 0\}$ and the guard set $G=\{(x,y)\mid y\leqslant 1\}$. It is easy to see that $Y(s)=\{s+1,s+2\}$ for $s<1$, $\{s,s+1,s+2\}$ for $s=1$, $\{s,s+1\}$ for $1<s\leqslant 2$ and $\{s\}$ for $2<s$. Again, $Y(s)$, and hence Reach, not lower-semicontinuous at $s=1,2$.

Combining Example 14 with Theorem 5, we obtain

Theorem 15: Let $H=(X,G,D,R,F)$ be a hybrid system R is continuous with compact values, and F is Lipschitz continuous with compact values, and let X_0 be a compact state set.

1) If G and D are closed, then the operator $(X_0,G,D,R,F)\mapsto \mathrm{Reach}_{(\mathcal{T},\mathcal{N})}(X_0,H)$ is not $(\kappa,\psi,\psi,\mu^\kappa,\mu^\kappa;\psi_<)$-computable.
2) If G and D are open, then the operator $(X_0,G,D,R,F)\mapsto \mathrm{clReach}_{(\mathcal{T},\mathcal{N})}(X_0,H)$ is not $(\kappa,\theta,\theta,\mu^\kappa,\mu^\kappa;\psi_>)$-computable.

D. *Continuous-time evolution in restricted domains*

We need to extend Theorem 11 to the case in which solutions of $\dot{x}\in F(x)$ are restricted to a domain D in the state space. However, we cannot use the lower representation $\psi_<$ for the domain D of a system, since any domain is then approximated by a finite set which prevents any continuous evolution. To obtain lower-semicomputability, we take D to be open, since we need to block solutions which graze the boundary of D.

Theorem 16:

1) If F is upper-semicontinuous with compact values and linear growth at infinity and D is closed, then $\Phi_T^{F,D}$ is a compact-valued upper-semicontinuous map, and $(F,D,C,T)\mapsto\Phi_T^{F,D}(C)$ is $(\mu_>^\kappa,\psi_>,\kappa_>,\rho;\kappa_>)$-computable.
2) If F is Lipschitz continuous with closed values and D is open, then $\Phi_T^{F,D}$ is a lower-semicontinuous map and $(F,D,A,T)\mapsto\mathrm{cl}(\Phi_T^{F,D}(A))$ is $(\mu_<^\psi,\theta_<,\psi_<,\rho;\psi_<)$-computable.

Notice that in (2), we need to take the closure of $\Phi_T^{F,D}(A)$, since $\Phi_T^{F,D}(A)$ need not be closed if the evolution reaches the boundary of D.

The proof follows that of Theorem 11, except that in (1) we intersect at each Euler step with a set $D_n\supset D$, and in (2) we intersect with a set $D_n\subset D$.

E. *Semicomputability of the* Reach *operator*

The aim of this section is to establish the semicomputability of the operator $\mathrm{Reach}_{(\mathcal{T},\mathcal{N})}$ under certain conditions. To evolve the system in continuous time within the domain, we will use the operator $\Phi_T^{F,D}$ to convert flows to maps. Theorem 16 provides conditions under which the states reachable by the hybrid system without taking a discrete transition can be semi-computed. These conditions can then be generalised to conditions for computing the states reachable with N discrete transitions.

To ensure that bounds on the continuous time are not violated when multiple discrete transitions are taken we introduce an auxiliary variable, $y\in\mathbb{R}$ to keep track of continuous time. We define an augmented hybrid system, $\hat{H}=(\hat{X},\hat{G},\hat{D},\hat{R},\hat{F})$, by $\hat{X}=X\times\mathbb{R}$, $\hat{G}=G\times\mathbb{R}$, $\hat{D}=D\times\mathbb{R}$, $\hat{R}(x,y)=(R(x),y)$, and $\hat{F}(x,y)=(F(x),1)$. Given a set of initial conditions X_0, we also define $\hat{X}_0=X_0\times\{0\}$.

From a ρ-name of T, we can semi-compute $\hat{Y}_0=\Phi_{[0,T]}^{\hat{F},\hat{D}}(\hat{X}_i)$, and then $\hat{X}_1=\mathrm{cl}(\hat{R}(\hat{Y}_0\cap\hat{G}))$. Continuing recursively, we compute sets $\hat{Y}_n$ such that $\hat{Y}_n$ contains all points which can be reached at time $s_0+s_1+\cdots s_n$, with transitions occurring at time intervals $s_0,\ldots s_{n-1}$ for $s_0,\ldots,s_n\in[0,T]$. Therefore $\hat{Y}_n$ contains all points which can be reached after n transitions at time t for any $t<T$, and some other points besides. To obtain the reachable set, we take $\hat{Y}=\bigcup_{n\in\mathcal{N}}\hat{Y}_n$, $\hat{Z}=\hat{Y}\cap X\times\mathcal{T}$, and $\mathrm{Reach}_{\mathcal{T},\mathcal{N}}(H,X_0)=\pi(\hat{Z})$, where $\pi:X\times\mathbb{R}\to X$ is given by $\pi(x,y)=x$. In the lower-semicomputable case, we instead need to take $\hat{Z}=\mathrm{cl}(\hat{Y}\cap X\times(T_1,T_2))$ We obtain the following result.

Theorem 17: Let $H = (X, G, D, R, F)$ be a hybrid system, and X_0 an initial state set.

1) Suppose G and D are closed, R is upper-semicontinuous with compact values, F is upper-semicontinuous with compact values and linear growth at infinity, and X_0 is compact. Then the operator $(X_0, G, D, R, F) \mapsto \mathrm{Reach}_{(\mathcal{T},\mathcal{N})}(X_0, G, D, R, F)$ is $(\kappa_>, \psi_>, \psi_>, \mu^\kappa_>, \mu^\kappa_>; \kappa_>)$-computable.
2) Suppose G and D are open, R is lower-semicontinuous with closed values, F is Lipschitz continuous with closed values, and X_0 is closed. Then the operator $\mathrm{clReach}_{(\mathcal{T},\mathcal{N})}(X_0, G, D, R, F)$ is $(\psi_<, \theta_<, \theta_<, \mu^\psi_<, \mu^\psi_<; \psi_<)$-computable.

In other words, the finite-time reachable set is semicomputable for appropriately-defined hybrid systems.

V. Hybrid systems with jump sets

In the previous section, we considered hybrid systems for which the mechanism forcing a discrete transition was the system leaving the domain of definition of the continuous evolution. However, hybrid systems are often defined by specifying *jump sets*, with a discrete transition being forced to occur whenever the trajectory crosses a jump set. The jump sets are typically codimension-one manifolds, which partition the state space into domains on which continuous evolution is possible.

In this section, we shall consider hybrid systems $H = (X, G, J, R, F)$ such that G and J are finite unions of closed, codimension-1 manifolds with $J \subset G$, $R : \mathrm{dom}(R) \rightrightarrows X$ is defined on some neighbourhood of G, and $F : X \rightrightarrows \mathrm{T}X$ is globally defined.

A. Codimension-one manifolds

To consider computability for such systems, we need a representation of a codimension-1 hypersurface M which allows us to determine domains of continuous evolution and crossings of the jump sets. The most convenient such representation is as the zero set of a map $f : X \to \mathbb{R}$.

Definition 18: Let M be a codimension-1 manifold. An η-name of M is a δ^{co}-name of a continuous map $f : X \to \mathbb{R}$ such that $M = f^{-1}(0)$ and f changes sign across M.

By [6, Theorem 6.2.9] we can compute an $\psi_>$-name of M from an η-name.

The jump set is a finite union of manifolds, $J = \bigcup_{i=1}^m M_i$, each of which is defined by a function f_i, so the entire set can be described by the function $f : X \to \mathbb{R}^m$ defined by $f(x) = (f_1(x), \ldots, f_m(x))$. Since $X \setminus M = f^{-1}\{y \in \mathbb{R}^m \mid y_i \neq 0 \; \forall i = 1, \ldots, m\}$, it $\theta_<$-computable from a δ^{co}-name of f. The components of $X \setminus M$ form open domains in which solutions may evolve continuously.

Given a point $x \notin J$, we are interested in finding closure of the component of $X \setminus J$ containing x. Define $Q_x = \{y \in \mathbb{R}^m \mid f_i(x) \cdot y_i > 0 \; \forall i = 1, \ldots, m\}$. Then $D_x = f^{-1}(\mathrm{cl}(Q_x))$ is a closed set containing x, and any path in D_x does not cross J, though it may touch J. Hence the sets D_x form a natural subdivision into closed domains in which solutions may evolve continuously.

B. Transverse crossings

We now consider the problem of computing the set of points at which trajectories cross a jump manifold. The computation depends crucially on the semantics used for defining a "crossing".

Definition 19: Let M be a connected codimension-1 manifold, and let U and V be the components of $X \setminus M$. Let $x(\cdot)$ be a continuous function $x : [0, T] \to X$. Then $x(\cdot)$ *crosses* M from U to V at time t if for all $\epsilon > 0$, there exist u, v with $t - \epsilon < u < t < v < t + \epsilon$ such that $x(u) \in U$ and $x(v) \in V$.

In the lower-semicontinuous case, difficulties arise when trajectories, instead of crossing transversely at a single point, flow along the manifold for some time interval. We therefore restrict the class of systems we consider.

Definition 20: Let M be a connected codimension-1 manifold, U, V the components of $X \setminus M$, and $\Phi : T \times X \rightrightarrows X$ a flow. We say that the crossings of M from U to V are *lower-detectable* if there exists $\delta > 0$ such that for all $x \in M \cap \mathrm{cl}(U)$, either

- $\Phi_t(x) \subset \mathrm{cl}(U)$ for all $t < \delta$, or
- $\forall \epsilon > 0$, $\exists t \in (0, \epsilon)$ such that $\Phi_t(x) \cap V \neq \emptyset$.

In other words, either all trajectories stay in $\mathrm{cl}(U)$ for a known time δ, or there exist trajectories which leave $\mathrm{cl}(U)$ after arbitrary small times. This precludes a situation where all trajectories starting at x remain in M for some unknown nonzero time δ' before leaving $\mathrm{cl}(U)$. Using this condition, the crossing set is lower-semicomputable:

Lemma 21: Let M be a manifold, U, V the components of $X \setminus M$, Φ a lower-semicontinuous flow with detectable crossings from U to V, and $A \subset \mathrm{cl}(U)$ closed. Let $\mathrm{cr}(\Phi, M, A, T)$ be the set of points at which the flow of Φ with initial set A crosses M in time $t < T$. Then $(\Phi, M, A, T) \mapsto \mathrm{cr}(\Phi, M, A, T)$ is $(\mu^\psi_<, \eta, \psi_<\rho; \psi_<)$-computable.

Note that we have not used any differentiability assumptions on the jump manifold in this section. Of course, differentiability of the jump manifold may be used to verify the lower-detectability of the crossing set, and in implementation strategies, but is not necessary for lower-semicomputability.

C. Computing reach for systems crossing guard manifolds

For semicontinuity of the system evolution, and hence semicomputability, we need to use a proper semantics for triggering discrete events. We use crossing semantics for the lower-semicontinuous case, and grazing semantics for upper-semicontinuous.

Definition 22: A solution $\{x_i(\cdot)\}$ of the hybrid system $H = (X, G, J, R, F)$ has

- *grazing semantics*, if $x_i(t) \in \mathrm{cl}(D_i)$ for all $t \in [t_i, t_{i+1}]$, and $x_i(t_{i+1}) \in G$ for all $i < N$.
- *crossing semantics*, if $x_i(t) \in D_i$ for all $t \in (t_i, t_{i+1})$, and $x_i(t_{i+1})$ is a crossing point of G for all $i < N$.

To compute the reachable set in the upper-semicontinuous case, we introduce discrete states corresponding to the quadrants Q of $\mathbb{R}^m$ for the jump set J, and partition elements $D = f^{-1}(Q)$ for each quadrant. Upper-semicomputability of Reach follows directly from Theorem 17(1).

For lower-semicontinuity, we use the crossing semantics, which means that a trajectory which grazes the guard set neither resets nor continues to flow. To compute the reachable set, we separately evolve within $X \setminus D$, and compute the crossing set for each discrete transition. Lower-semicomputability of Reach follows from Theorem 16(2), Lemma 21 using the proof technique of Theorem 17.

Theorem 23: Let $\mathcal{H} = (X, G, J, R, F)$ be a hybrid system where G and J are codimension-1 manifolds with $J \subset G$, and let X_0 be a set of initial states.

1) Suppose R is upper-semicontinuous with compact values and F is upper-semicontinuous with compact values, trajectories of $\mathcal{H}$ are defined using the grazing semantics, $\mathcal{T}$ is compact and X_0 is compact. Then the operator $\mathrm{Reach}_{\mathcal{T},\mathcal{N}}(X_0, G, J, R, F)$ is $(\kappa_>, \eta, \eta, \mu^\kappa_>, \mu^\kappa_>; \kappa_>)$-computable.
2) Suppose R is lower-semicontinuous with closed values, F is Lipschitz with convex closed values, trajectories of $\mathcal{H}$ are defined using the crossing semantics, $\mathcal{T}$ is open and X_0 is closed. Then the operator $\mathrm{clReach}_{\mathcal{T},\mathcal{N}}(X_0, G, J, R, F)$ is $(\psi_<, \eta, \eta, \mu^\psi_<, \mu^\psi_<; \psi_<)$-computable.
3) Suppose R is continuous with compact values, F is Lipschitz with convex compact values and linear growth at infinity, and X_0 is a compact set. Suppose further that all trajectories are transverse to G, and that no discrete events occur at the endpoints of $\mathcal{T}$. Then the operator $\mathrm{clReach}_{\mathcal{T},\mathcal{N}}(X_0, G, J, R, F)$ is $(\psi, \eta, \eta, \mu^\kappa, \mu^\kappa; \kappa)$-computable.

VI. CONCLUSIONS

In this paper we have studied the computability of finite-time reachable sets for hybrid systems from the point of view of type-two computability and computable analysis. We showed that under suitable assumptions, the system evolution may be lower-semicomputable or upper-semicomputable, but not both. The fundamental obstruction to computability is the possibility of grazing contact with guard and/or jump sets, which causes discontinuities in the system evolution. This lack of computability may be avoided by restricting to the case of transverse crossing of guard sets.

Although the results contained here are only for finite-time system evolution, they form a basis for the computability of other system properties. Using results of [2] for discrete-time systems, we can easily semicompute infinite-time reachable sets, and also viable and invariant sets for semicontinuous hybrid systems. This extends the results of [5] for viability kernels of upper-semicontinuous hybrid systems.

Our computability results can be used to produce algorithms for the computations involved. However, naive implementations of these algorithms are likely to be prohibitively expensive. Due to the high level of complexity of general nonlinear systems, we expect even highly-optimised algorithms to only be useful for highly robust systems or in low dimensions. Finally, most existing tools for hybrid systems analysis use floating-point computation and do not control the error, and so do not give algorithms for the computable problems discussed.

Acknowledgement: Work supported by the European Commission through the CC project (IST-2001-33520) and the Network of Excellence HYCON (IST-511368).

REFERENCES

[1] P. Collins, “Continuity and computability of reachable sets,” to appear in *Theor. Comput. Sci.*

[2] ——, “On the computability of reachable and invariant sets,” in *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005.

[3] A. Puri, P. Varaiya, and V. Borkar, “Epsilon-approximation of differential inclusions,” in *Hybrid Systems III*, ser. LNCS, R. Alur, T. A. Henzinger, and E. D. Sontag, Eds., vol. 1066. Berlin: Springer, 1996, pp. 362–376.

[4] D. Szolnoki, “Set oriented methods for computing reachable sets and control sets,” *Discrete Contin. Dyn. Syst. Ser. B*, vol. 3, no. 3, pp. 361–382, 2003.

[5] J.-P. Aubin, J. Lygeros, M. Quincampoix, and S. Sastry, “Impulse differential inclusions: A viability approach to hybrid systems,” *IEEE Trans. Automatic Control*, vol. 47, no. 1, pp. 2–20, 2002.

[6] K. Weihrauch, *Computable analysis*, ser. Texts in Theoretical Computer Science. An EATCS Series. Berlin: Springer-Verlag, 2000, an introduction.

[7] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, “Discrete abstractions of hybrid systems,” *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, July 2000.

[8] O. Aberth, *Computable analysis*. New York: McGraw-Hill, 1980.

[9] M. B. Pour-El and J. I. Richards, *Computability in analysis and physics*, ser. Perspectives in Mathematical Logic. Berlin: Springer-Verlag, 1989.

[10] K.-I. Ko, *Complexity theory of real functions*, ser. Progress in Theoretical Computer Science. Boston, MA: Birkhäuser Boston Inc., 1991.

[11] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and real computation*. New York: Springer-Verlag, 1998, with a foreword by Richard M. Karp.

[12] M. Dellnitz, G. Froyland, and O. Junge, “The algorithms behind GAIO-set oriented numerical methods for dynamical systems,” in *Ergodic theory, analysis, and efficient simulation of dynamical systems*, B. Fiedler, Ed. Berlin: Springer, 2001, pp. 145–174, 805–807.

[13] J.-P. Aubin and A. Cellina, *Differential inclusions*, ser. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Berlin: Springer-Verlag, 1984, vol. 264, set-valued maps and viability theory.

[14] K. Deimling, *Multivalued differential equations*, ser. de Gruyter Series in Nonlinear Analysis and Applications. Berlin: Walter de Gruyter & Co., 1992, vol. 1.

[15] P. Collins, “A trajectory-space approach to hybrid systems,” in *Proceedings of the International Symposium on the Mathematical Theory of Networks and Systems, Katholiek Univ. Leuven, Belgium., August 2004*, 2004.

[16] J. Lygeros, K. Johansson, S. Simić, J. Zhang, and S. Sastry, “Dynamical properties of hybrid automata,” *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 2–17, 2003.

[17] L. Tavernini, “Differential automata and their simulators,” *Nonlinear Analysis, Theory, Methods and Applications*, vol. 11(6), pp. 665–683, 1987.