Computable Types for Dynamic Systems

Pieter Collins[⋆]

Centrum Wiskunde en Informatica
Postbus 94079
1090 GB Amsterdam
Pieter.Collins@cwi.nl

Abstract. In this paper, we develop a theory of computable types suitable for the study of dynamic systems in discrete and continuous time. The theory uses type-two effectivity as the underlying computational model, but we quickly develop a type system which can be manipulated abstractly, but for which all allowable operations are guaranteed to be computable. We apply the theory to the study of differential inclusions, reachable sets and controllability.

1 Introduction

Dynamic systems which are used to model time-varying processes in almost all fields of science. Such systems are studied by means of computer simulation of a mathematical model, using approximate numerical schemes and inexact floating-point arithmetic. While this is usually sufficient to obtain an understanding of the system evolution, such an approach cannot be used to rigorously analyse the system behaviour. In order to perform a rigorous analysis we can either resort to an algebraic approach, which is usually too inefficient to be of practical use, or try to perform numerical computations with bounds on the errors. This naturally leads to questions concerning the best representation of the data, and what is actually possible to compute.

There are a number of existing approaches to continuous mathematics in which computational issues are considered, from informal treatments in contructive analysis [BB85], through more formal approaches using domain theory [GHK$^+$03] or locale theory [Joh02], to the low-level theory of type-two effectivity [Wei00]. Each of these approaches has its advantages and disadvantages. Constructive analysis is closest to "working mathematics", and thus is easiest to work with, but the relationship to computation is not explicit. The formal approaches such as domain and locale theory are traditionally also not set up directly as a theory of computation, but this relation can be formalised. While powerful, these theories have their own terminology and development which make them hard to understand without a background in logic. The theory of type-two effectivity provides a direct connection with digital computation, and has been developed furthest in the direction of dynamic systems, but has a cumbersome notation which makes it unwieldy to work with for higher-level mathematics.

The aim of this paper is to give an exposition of a theory of computation in analysis, leading to results on computability of the evolution of some important classes of dynamic systems. We use type-two effectivity to provide a direct link with real compuation, but aim

⋆ This research was supported by the Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO) Vidi grant 639.032.408.

to hide the low-level details in a theory of data types which can then be manipulated in a natural way using constructive mathematics. We avoid using notation and terminology from domain theory and locale theory as much as possible, though the exposition is guided by results in these fields. We address the concrete problems of computing the finite-time evolution of nondeterministic finite-dimensional systems in both discrete time and continuous time, the latter case being the theory of *differential inclusions*.

The paper is organised as follows. In Section 2 we provide a brief sketch of a topological (non-computable) type theory. In Section 3, we develop a theory of computable types for spaces of points, sets and functions. In Section 4 we apply this type theory to the study of dynamic systems.

2 Type Theory and General Topology

In this section, we review existing results on topological spaces without any explicit computational structure. The main aim is to explore a *topological type theory*, in which the types are topological spaces.

2.1 Type theory

In a standard type theory, we are most interested in products $X \times Y$ and exponentials Y^X. In the category of sets, products to the normal Cartesian product, and exponentials to functions from X to Y. The most important operations on these constructions are projection $\pi_i : X_1 \times X_2 \to X_i$, and isomorphism between the type $Y^{A\times X}$ and the type $(Y^X)^A$ via the bijection $f \mapsto \hat{f}$ defined by $\hat{f}(a)(x) = f(a,x)$. We say a category is *Cartesian closed* if we can form products and exponentials with the required properties.

2.2 Topological type theory

In the topological category, we identify the exponential type Y^X with $C(X;Y)$, the space of continuous functions from X to Y. It remains to provide a topology on $C(X;Y)$, and it is here that problems arise. For it is well-known that it is not always possible to find a topology on $C(X;Y)$ such that the set of continuous function $A \times X \to Y$ is in bijective correspondance with the set of continuous functions $A \to C(X;Y)$ for all spaces A.

We say that a topology on $C(X;Y)$ is *exponential* if for any space A, a function $f : A \times X \to Y$ is continuous if, and only if, $\hat{f} : A \to C(X;Y)$ is continuous. It turns out that X is exponentiable if, and only if, it is *core-compact* (see [EH02]). In this case, the exponential topology on $C(X;Y)$ is the *Isbell* topology, which is generated by the *Scott* topology on the open sets of X. If X is a Hausdorff space, then core-compactness is equivalent to local compactness.

While most spaces which are commonly used as state spaces in dynamical systems theory are locally-compact Hausdorff spaces the space of possible solutions, or *trajectory space* is a function space, typically $X^{\mathbb{N}}$, $X^{\mathbb{Z}}$ or $C(\mathbb{R}^+;X)$. Such function spaces are decidedly not locally-compact; typically no compact set has an open subset. We therefore need a theory which can handle non-locally-compact spaces.

Fortunately, there are subcategories of the category of topological spaces which are Cartesian closed. Most interesting for us are the quotients of second-countable spaces, sometimes known as the *quotients of countably-based* (QCB) spaces. It is known that the full category of QCB spaces is Cartesian closed (see [Sch02,ELS04]), so we can define topological types in this category. Even more promising, these spaces can be given a *computable structure* by means of representations by infinite sequences on some finite alphabet. Further, these spaces include all spaces encountered in the study of finite-dimensional dynamics systems, including Euclidean space, manifolds, continuous functions on these spaces, and hyperspaces of open, closed and compact sets.

2.3 Sets and maps in type theory

Starting from (topological) types X and Y, we can define products $X \times Y$ and exponentials Y^X, the latter corresponding to continuous functions $C(X;Y)$. However, using these operations, we can also define subsets of spaces. The key to this construction is the *Sierpinski* space $S = \{\top, \uparrow\}$ with open sets $\{\{\,\}, \{\top\}, \{\top, \uparrow\}\}$. Here, $\top$ denotes "true", and $\uparrow$ denotes "divergent" or "don't know", and can be thought of as representing a computation which never halts.

It is easy to see that the set of open subsets of X, denoted $O(X)$, is in bijective correspondance with the space of continuous functions $X \to S$, since a set U is open if, and only if, the function $f_U : X \to S$ defined by $f_U(x) = \top \iff x \in U$ is continuous. Hence we should identify $O(X)$ and S^X in a topological type theory. The closed subsets $A(X)$ are in bijective correspondance with their complements.

Similarly, if A and C are sets, we can define functions $g_A : O(X) \to S$ by $g_A(U) = \top \iff A \cap U \neq \emptyset$, and $h_C : O(X) \to S$ by $h_C(U) = \top \iff C \subset U$. We henceforth write $A \bowtie U$ for $A \cap U \neq \emptyset$. It is clear that the function g_A satisfies $g_A(U \cup V) = g_A(U) \vee g_A(V)$, and $h_C(U \cap V) = h_C(U) \cap h_C(V)$. Further, given a "sufficiently reasonable" topology on $O(X)$, the functions g_A and h_C are continuous for A closed and C compact, and further, any such continuous function g_A satisfying the union condition arises in this way from a closed set, and any such continuous function h_C satisfying the intersection condition arises in this way from a compact set. By "sufficiently reasonable", it suffices that inclusion $x \in U$ is continuous in both x and U. Following [Tay08], we call the space of closed sets with the topology defined by g_A as the space of *overt* sets. We can therefore identify overt sets and compact sets with subtypes of $S^{(S^X)}$.

3 Computable Analysis

We now give a computational meaning to our purely topological theory. The theory is based on the theory of type-two effectivity of [Wei00], though the development here is more abstract, similar to that of Schröder [Sch02].

3.1 Machine-computability

We fix a finite alphabet Σ, and give the space Σ^ω the product topology. In particular, Σ^ω is a second-countable, locally-compact zero-dimensional Hausdorff space. By considering *type-two* Turing machines working on streams of data identified with Σ^ω, we define a set of *machine-computable* partial functions $\eta :\subset \Sigma^\omega \times \cdots \times \Sigma^\omega \to \Sigma^\omega$. The domain of a machine-computable function is always a G_δ set. (Recall that a G_δ-set is a countable intersection of open sets.) The set of machine-computable functions is countable, and closed under composition. We also define the uncountable set of *machine-continuous* functions $\eta :\subset \Sigma^\omega \times \cdots \times \Sigma^\omega \to \Sigma^\omega$ to be the continuous functions with G_δ-domain.

In this paper, we do not need to concern ourselves with the exact definition of machine-computable function. However, we will need to use some more abstract results on continuous and computable functions, which can be found in [Wei00, Chapter 3].

Theorem 1.

1. *There exists a surjective function $\delta :\subset \Sigma^\omega \to C(\Sigma^\omega; \Sigma^\omega)$ whose range consist of all machine-continuous partial functions (with G_δ-domain) and a machine-computable function $\varepsilon : \Sigma^\omega \times \Sigma^\omega \to \Sigma^\omega$ such that $\delta(p)(q) = \varepsilon(p, q)$ for all q.*
2. *There exists a computable function $\Sigma^* \to \Sigma^\omega$ whose range consists of all elements of $\mathrm{dom}(\delta)$ corresponding to machine-computable functions.*

3.2 Computable types

In this section, we show how to define computable stuctures on mathematical objects, in particular, on topological spaces.

Definition 1 (Representation). *A* representation *of a set M is a partial surjective function $\delta :\subset \Sigma^\omega \to M$.*

Representations δ_1 and δ_2 of M are equivalent *if there are machine-computable functions η_1, η_2 such that $\delta_1 \circ \eta_1 = \delta_2$ and $\delta_2 \circ \eta_2 = \delta_1$.*

A computable type *is an equivalence class of pairs (M, δ), where (M, δ_1) is equivalent to (M, δ_2) if δ_1 and δ_2 are equivalent. We shall usually denote a computable type with underlying set M by $\mathcal{M}$.*

Note that the domain of a representation is an arbitrary set. Where possible, it is useful to use topologically "nice" sets as domains to help checking whether a sequence is a valid name. The following definition shows how representations induce a computable structure on general sets.

Definition 2 (Computable function). *Let $f : M_1 \times \cdots \times M_k \to Y$ be a function, and $\delta_{M_i} :\subset \Sigma^\omega \to M_i$ and $\delta_Y :\subset \Sigma^\omega \to Y$ be representations. Then a machine-continuous function $\eta :\subset (\Sigma^\omega)^k \to \Sigma^\omega$* realises *$f$ if $f(\delta_{X_1}(p_1), \ldots, \delta_{X_k}(p_k)) = \delta_Y(\eta(p_1, \ldots, p_k))$ for all $p_i \in \mathrm{dom}(\delta_i)$. We say f is* computable *if it is realised by a machine-computable function.*

Recall that a continuous function $\delta : A \to X$ is a *quotient map* if δ is surjective and whenever $\phi : A \to Y$ is continuous and $\phi(p) = \phi(q)$ whenever $\delta(p) = \delta(q)$, then there exists a map $g : A \to Y$ such that $\phi = g \circ \delta$. A continuous function $\delta : A \to X$ is *universal*, if whenever $\phi : A \to X$ is continuous, there exists a continuous function $\eta : A \to A$ such that $\phi = \delta \circ \eta$.

Definition 3 (Admissible representation). *An representation δ of a topological space (X,τ) is* admissible *if it is a universal quotient map.*

We require an admissible representation of a topological space to be a universal so that it captures all the topological information about a space. We require it to be a quotient map so that continuity on (X,τ) is reflected in Σ^ω. The *standard representations* of a second-countable Kolmogorov (T_0) space as defined in [Wei00, Chapter 3] are always admissible in the above sense. The following lemma (see [Sch02]) shows that we can extract the topology of X from an admissible representation δ.

Lemma 1. *There is at most one topology τ on X making $\delta :\subset \Sigma^\omega \to X$ an admissible representation of (X,τ).*

In general there are many non-equivalent representations admissible for a given topological space. Not all spaces have admissible representations with "nice" names; there is no admissible representation of $C(C(\mathbb{R};\mathbb{R});\mathbb{R})$ with a G_δ domain.

Up to equivalence there is only one admissible representation on $\mathbb{R}$ making arithmetic and comparison computable, so we can talk of a canonical type $\mathcal{R}$ of real numbers [Bau00, Theorem 5.5.18]; see also [Bra98]. Most other types used in analysis can be built from $\mathcal{R}$ in a natural way.

Theorem 2. *Let δ_X, δ_Y be admissible representations of spaces X and Y.*
1. *If $f : X \to Y$ be continuous, then there exists a machine-continuous function η such that $\mathrm{dom}(\eta) \supset \mathrm{dom}(\delta_X)$ and $\delta_Y \circ \eta = f \circ \delta_X$ on $\mathrm{dom}(\delta)$.*
2. *Suppose $f : X \to Y$, and there exists a machine-continuous function η with $\mathrm{dom}(\eta) \supset \mathrm{dom}(\delta)$ such that $f \circ \delta_X = \delta_Y \circ \eta$ on $\mathrm{dom}(\delta)$. Then f is continuous.*

The first part of the theorem says that any continuous function is realised by a machine-continuous function. The second part of the theorem says that any function realised by a machine-continuous function (in particular, by a machine-computable function) is continuous.

3.3 Types of continuous functions

We now give the main results of computability of operations on continuous functions. The next result asserts that function evaluation is computable.

Theorem 3. *Let X and Y be spaces with admissible representations δ_X and δ_Y. Then there is a canonical representation $\delta_{[X\to Y]}$ of $C(X;Y)$ such that evalution $C(X;Y) \times X \to Y$ is computable. This representation defines a canonical type $\mathcal{C}(\mathcal{X};\mathcal{Y})$.*

The next theorem asserts computability of the fundamental operations on types.

Theorem 4. *Let W, X and Y be spaces with admissible representations δ_A, δ_X and δ_Y respectively. Let $\mathcal{W}$, $\mathcal{X}$ and $\mathcal{Y}$ denote the types of (W,δ_W), (X,δ_X) and (Y,δ_Y). Then there is a computable bijection between the types $\mathcal{Y}^{\mathcal{W}\times\mathcal{X}}$ and $\mathcal{Y}^{\mathcal{X}^{\mathcal{W}}}$ taking f to $\hat{f}$.*

The following theorem shows that if we can effectively evaluate a function $f : X \to Y$, then we can compute a name. We will repeatedly use this result to prove computability of a function type by showing that it can be effectively evaluated.

Theorem 5. *Suppose $\mathcal{X}$ and $\mathcal{Y}$ are computable types, and $f : X \to Y$. Then if we can effectively evaluate $f(x)$ for all $x \in X$, we can effectively compute f in $\mathcal{C}(\mathcal{X};\mathcal{Y})$*

3.4 Point and set types

We now use the computable function types to derive the set types we need to study systems.

Definition 4 (Types of open, closed and compact sets).
1. *We identify the type of* open sets, *denoted $\mathcal{O}(\mathcal{X})$ with the space of continuous function $\mathcal{X} \to \mathcal{S}$ by $f_U(x) = \top \iff x \in U$.*
2. *We identify the type of* closed sets, *denoted $\mathcal{A}(\mathcal{X})$ with the space of continuous function $\mathcal{X} \to \mathcal{S}$ by $f_A(x) = \top \iff x \notin U$.*
3. *We identify the type of* overt sets, *denoted $\mathcal{V}(\mathcal{X})$ with the space of continuous functions $f_A : \mathcal{O}(\mathcal{X}) \to \mathcal{S}$ satisfying $f_A(U \cup V) = f_A(U) \vee f_A(V)$.*
4. *We identify the type of* compact sets, *denoted $\mathcal{K}(\mathcal{X})$ with the space of continuous functions $f_C : \mathcal{O}(\mathcal{X}) \to \mathcal{S}$ satisfying $f_C(U \cap V) = f_C(U) \wedge f_C(V)$.*

Concrete representations for these types in the case of Euclidean/metric spaces are given in [BW99,BP03]. Most of the basic set-theoretic operations, including union and intersection, are computable without additional assumptions on the space X. However, we will sometimes need to work in Hausdorff spaces in which the apartness relation is computable.

Definition 5. *We say $\mathcal{X}$ is a* effectively separated *if the function $s : \mathcal{X} \times \mathcal{X} \to \mathcal{S}$ defined by $s(x,y) = \top \iff x \neq y$ is computable. We say $\mathcal{X}$ is a* effectively separable *if there is a computable function $r : \mathcal{N} \to \mathcal{X}$ whose range is dense in X.*

Note that although any singleton set in a T_1 space is closed, singleton function is only continuous in a Hausdorff (T_2) space.

We have the following computability results on types. Note that having developed the basic theory, and shown that computing a function type is equivalent to being able to evaluate it, the rest of the theory is almost trivial.

Theorem 6. *The following operations on sets are computable:*
1. *Finite intersection $\mathcal{O} \times \mathcal{O} \to \mathcal{O}$ and arbitrary union $\mathcal{O}^{\mathbb{N}} \to \mathcal{O}$.*
2. *Finite union $\mathcal{A} \times \mathcal{A} \to \mathcal{A}$ and arbitrary intersection $\mathcal{A}^{\mathbb{N}} \to \mathcal{A}$.*
3. *Complement $\mathcal{O} \to \mathcal{A}$ and $\mathcal{A} \to \mathcal{O}$.*
4. *Countable union $\mathcal{V}^{\mathbb{N}} \to \mathcal{V}$ and finite union $\mathcal{K} \times \mathcal{K} \to \mathcal{K}$.*
5. *Finite intersection $\mathcal{V} \times \mathcal{O} \to \mathcal{V}$ and $\mathcal{K} \times \mathcal{A} \to \mathcal{K}$.*
6. *Singleton $\mathcal{X} \to \mathcal{V}$, $\mathcal{X} \to \mathcal{K}$.*
7. *Closure $\mathcal{O} \to \mathcal{V}$ if $\mathcal{X}$ is effectively separable, and identity $\mathcal{K} \to \mathcal{A}$ if $\mathcal{X}$ is effectively separated.*
8. *Preimage $\mathcal{C}(\mathcal{X};\mathcal{Y}) \times \mathcal{O}(\mathcal{Y}) \to \mathcal{O}(\mathcal{X})$.*
9. *Image $\mathcal{C}(\mathcal{X};\mathcal{Y}) \times \mathcal{V}(\mathcal{X}) \to \mathcal{V}(\mathcal{Y})$ and $\mathcal{C}(\mathcal{X};\mathcal{Y}) \times \mathcal{K}(\mathcal{X}) \to \mathcal{K}(\mathcal{Y})$.*

Due to the strong conditions on admissible representations, and application of Theorem 5, the proofs are almost deceptively straightforward.

4 Computability for Dynamic Systems

We now use the computable type theory developed in the previous section to give some results on computable properties of dynamic systems.

4.1 Spaces of multifunctions

When considering solutions of nondeterministic systems, we are often interested in function spaces with set-valued types, or hyperspaces of functions.

The set of solutions of a dynamic system is the space of continuous functions $\xi : T \to X$, where T is the time domain, and X is the state space. For an autonomous system, we require *time-invariance*, that if ξ is a solution and $s \in T$, then the function defined by $\eta(t) = \xi(t + s)$ is also a solution. We also require the *property of state*, that if ξ and η are solutions with $\xi(s) = \eta(s)$, then there is a solution ζ with $\zeta(t) = \xi(t)$ for $t \leqslant s$, and $\zeta(t) = \eta(t)$ for $t \geqslant s$.

For a deterministic system, there is only one trajectory through a given initial state. The solution space may be represented either as a function $\phi : X \times T \to X$, or as a function $\hat{\phi} : X \to C(T; X)$. By the exponentiation property, these representations are equivalent.

In a *nondeterministic system* there may be may different trajectories with the same initial state. If the time domain is $\mathbb{R}$, we call the resulting dynamics a *multiflow*. In this case, there are many different ways of representing the solution space. The simplest way of representing the solution space is as the *behaviour* of the system, which is simply the set of all solutions, $\Phi \in \mathcal{P}(C(T; X))$. However, we can also represent the solution space as a function $\hat{\Phi} : X \to \mathcal{P}(C(T; X))$ such that $\xi(0) = x$ for all $\xi \in \hat{\Phi}(x)$. Another useful representation is in terms of the finite *reachability* operator, $\tilde{\Phi} : X \times T \to \mathcal{P}(X)$. We shall see that while Φ and $\hat{\Phi}$ are classically equivalent, in a computational setting Φ may be hard to define and contains less information. Further, Φ and $\tilde{\Phi}$ are classically and computationally equivalent for compactly-generated systems, but not for others.

The following result gives relationships between multiflow representations:

Lemma 2. *The function $B : \mathcal{K}(\mathcal{X}) \times \mathcal{O}(\mathcal{Y}) \to \mathcal{O}(\mathcal{C}(\mathcal{X}; \mathcal{Y}))$ defined by $B(K, V) = \{f \mid f(K) \subset V\}$ is computable, as is the function $B : \mathcal{K}(\mathcal{X}) \times \mathcal{A}(\mathcal{Y}) \to \mathcal{A}(\mathcal{C}(\mathcal{X}; \mathcal{Y}))$ defined by $B(K, A) = \{f \mid f(K) \subset A\}$.*

We now consider multiflows taking values in the class of overt and compact sets.

Lemma 3.

1. *The types $\hat{\Phi} : \mathcal{X} \to \mathcal{K}(\mathcal{C}(\mathcal{T}; \mathcal{X}))$ satisfying $\xi(0) = x$ for all $\xi \in \hat{\Phi}(x)$, and the types $\tilde{\Phi} : \mathcal{X} \times \mathcal{T} \to \mathcal{K}(\mathcal{X})$ are equivalent. We can also compute $\hat{\Phi}$ from Φ, and Φ from $\hat{\Phi}$ if X is compact.*
2. *if $\mathcal{T} = \mathbb{R}$, the type $\hat{\Phi} : \mathcal{X} \to \mathcal{V}(\mathcal{C}(\mathcal{T}; \mathcal{X}))$ is strictly stronger than both $\Phi \in \mathcal{V}(\mathcal{C}(\mathcal{T}; \mathcal{X}))$ and $\Phi : \mathcal{X} \times \mathcal{T} \to \mathcal{V}(\mathcal{X})$. The latter are uncomparible.*

The proofs of the above results are straightforward.

4.2 Computability theory for multivalued maps

A *multivalued map* is a function $F : X \to \mathcal{P}(Y)$. For a set $A \subset X$, we define $F(A) = \bigcup\{F(x) \mid x \in A\}$. For $B \subset Y$, we define $F^{-1}(B) = \{x \in X \mid F(x) \cap B \neq \emptyset\}$ and $F^{\Leftarrow}(B) = \{x \in X \mid F(x) \subset B\}$. Note that $F^{\Leftarrow}(B) = X \setminus F^{-1}(Y \setminus B)$. For $F : X \to \mathcal{P}(Y)$ and $G : Y \to \mathcal{P}(Z)$, we define $G \circ F : X \to \mathcal{P}(Z)$ by $G \circ F(x) = G(F(x))$. We say that F is *lower-semicontinuous* if $F^{-1}(V)$ is open whenever V is open, and *upper-semicontinuous* if $F^{-1}(B)$ is closed whenever B is closed.

We are interested in the case that F is a continuous function from X to a hyperspace of subsets of Y; in particular, for $F : X \to \mathcal{V}(Y)$ and $F : X \to \mathcal{K}(Y)$. In this case, we have the following properties. The proof is in [Col05].

Theorem 7. *The following types are computabably equivalent:*
1. $F : \mathcal{X} \to \mathcal{V}(\mathcal{Y})$*,* $F^{-1} : \mathcal{O}(\mathcal{Y}) \to \mathcal{O}(\mathcal{X})$ *and* $F : \mathcal{V}(\mathcal{X}) \to \mathcal{V}(\mathcal{Y})$*.*
2. $F : \mathcal{X} \to \mathcal{K}(\mathcal{Y})$*,* $F^{-1} : \mathcal{A}(\mathcal{Y}) \to \mathcal{A}(\mathcal{X})$ *and* $F : \mathcal{K}(\mathcal{X}) \to \mathcal{K}(\mathcal{Y})$*.*

We can therefore compute the forward-time evolution of discrete-time multivalued systems. We denote the type of non-negative integers by $\mathcal{N}$.

Corollary 1. *The behaviour of a discrete-time system F is computable in the following cases:*
1. *If* $F : \mathcal{X} \to \mathcal{X}$*, then* $\hat{\Phi} : \mathcal{X} \to \mathcal{C}(\mathcal{N}, \mathcal{X})$ *is computable from* F*.*
2. *If* $F : \mathcal{X} \to \mathcal{V}(\mathcal{X})$*, then* $\hat{\Phi} : \mathcal{X} \to \mathcal{V}(\mathcal{C}(\mathcal{N}, \mathcal{X}))$ *is computable from* F*.*
3. *If* $F : \mathcal{X} \to \mathcal{K}(\mathcal{X})$*, then* $\hat{\Phi} : \mathcal{X} \to \mathcal{K}(\mathcal{C}(\mathcal{N}, \mathcal{X}))$ *is computable from* F*.*

4.3 Computability theory for differential systems

In this section we consider the computability of systems defined by differential equations or differential inclusions. For simplicity, we assume that X is a Euclidean space $\mathbb{R}^n$, though these results also extend to differential manifolds and locally-compact Banach spaces. We state the results in this section without proof, as we need to go back to first principles to solve the differential systems. In particular, we need to resort to the classical Arzela-Ascoli theorem to assert the existence of solutions. We denote the type of real numbers by $\mathcal{R}$.

Theorem 8. *Let $f : X \to X$ be locally-Lipschitz continuous. Then the solution operator of $\dot{x} = f(x)$ is computable $\mathcal{C}(\mathcal{X}; \mathcal{X}) \times \mathcal{X} \to \mathcal{C}(\mathcal{R}, \mathcal{X})$.*

The proof is essentially standard [DM70], though is too long to include here. A simple proof can be found in [CG08]. Note that we can weaken the locally-Lipschitz condition to simply requiring uniqueness of solutions [Ruo96].

We now turn to nondeterministic differential systems as defined by differential inclusions $\dot{x} \in F(x)$. For an introduction to differential inclusions, see [AC84]. Following the well-known Filippov solution concept, we may first need to compute the convex hull of the right-hand side.

Lemma 4. *Closed convex hull is a computable operator $\mathcal{V} \to \mathcal{V}$ and $\mathcal{K} \to \mathcal{K}$.*

We can now state the main theorems on computability of solutions of differential inclusions. The continuous case was first proved in [PVB96], but easily splits into the lower- and upper-semicontinuous cases. The one-sided Lipschitz condition was developed in [Gab07].

Theorem 9.

1. *Let F be one-side locally-Lipschiz closed-convex-valued lower-semicontinuous. Then the solution operator of $\dot{x} \in F(x)$ is computable $\mathcal{C}(\mathcal{X};\mathcal{V}(\mathcal{X})) \times \mathcal{X} \to \mathcal{V}(\mathcal{C}(\mathcal{R},\mathcal{X}))$.*
2. *Let F be compact-convex-valued upper-semicontinuous. Then the solution operator of $\dot{x} \in F(x)$ is computable $C(X;\mathcal{K}(X)) \times \mathcal{X} \to \mathcal{K}(\mathcal{C}(\mathcal{R},\mathcal{X}))$.*

4.4 Computability theory for infinite-time properties

We now apply the results of Section 4.2 to prove computability of some infinite-time operators in discrete-time dynamical systems. The results can be found in [Col05]. We will need the following result, which shows that we can separate compact and closed sets.

Lemma 5. *There is a recursively enumerable set D of pairs $(A,B) \in \mathcal{O} \times \mathcal{K}$ such that $A \subset B$ such that for any compact K and open U, there exist (A_i,B_i) such that $K \subset A_i$ and $B_i \subset U$.*

We define the *reachable set* of a system F with initial state set X_0 as

$$\mathrm{reach}(F,X_0) = \{x \in X \mid \exists \text{ solution } \xi \text{ and } t \in T \text{ with } \xi(0) \in X_0 \text{ and } \xi(t) = x\}.$$

Theorem 10. *The reachable set operator* reach *is computable as a function $\mathcal{C}(\mathcal{X};\mathcal{V}(\mathcal{X})) \times \mathcal{V}(\mathcal{X}) \to \mathcal{V}(\mathcal{X})$, but not as a function $\mathcal{C}(\mathcal{X};\mathcal{K}(\mathcal{X})) \times \mathcal{K}(\mathcal{X}) \to \mathcal{K}(\mathcal{X})$.*

We define the *chain-reachable set* of F as limit of all ϵ-orbits, or equivalently as

$$\mathrm{chain\,reach}(F,X_0) = \bigcap\{U \in \mathcal{O}(X) | \mathrm{cl}(U) \text{ is compact, and } X_0 \cup F(\mathrm{cl}(U)) \subset U\}.$$

Theorem 11. *If* $\mathrm{chain\,reach}(F,X_0)$ *is bounded, then* $\mathrm{chain\,reach} : \mathcal{C}(\mathcal{X};\mathcal{K}(\mathcal{X})) \times \mathcal{K}(\mathcal{X}) \to \mathcal{K}(\mathcal{X})$ *is computable, and is the optimal $\mathcal{K}(\mathcal{X})$-computable over-approximation to* reach.

4.5 Computability theory for control systems

A *noisy control system* with state space X, input space U and noise space V is a function $f : X \times U \times V \to X$. We assume that U is an overt space and V a compact space, and define $F_U : X \to X \times U$, $F_U(x) = \{(x,u) \mid u \in U\}$, and $F_V : X \times U \to X$ by $F_V(x,u) = \{f(x,u,v) \mid v \in V\}$.

The *controllable set* of $\mathrm{ctrl}(f,T,S)$ with target set T and safe set S is determined recursively by $T_0 = T \cap S$ and $T_{i+1} = T_i \cup \{x \in X \mid \exists u \in U,\ \forall v \in V,\ f(x,u,v) \in T_i\} \cap S$.

The following result was proved in [Col08]:

Theorem 12. *The controllable set operator* $\mathrm{ctrl} : \mathcal{C}(\mathcal{X},\mathcal{U},\mathcal{V};\mathcal{X}) \times \mathcal{O}(\mathcal{X}) \times \mathcal{O}(\mathcal{X}) \to \mathcal{O}(\mathcal{X})$ *is computable.*

5 Conclusions

In this paper, we have developed a computable type theory sufficient for allowing the analysis of dynamic systems. Further extensions involve to infinite-dimensional systems and stochastic systems, which require additional work to prove computability of the evolution.

Additional work is also required to understand the function space topologies involved, especially the topology on $O(C(X;Y))$.

Acknowledgement The author would like to thank Martín Escardó for useful advice and suggestions regarding function space topologies, and the anonymous referees.

References

[AC84] Jean-Pierre Aubin and Arrigo Cellina. *Differential inclusions*, volume 264 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, Berlin, 1984.

[Bau00] Andrej Bauer. *The Realizability Approach to Computable Analysis and Topology*. PhD thesis, Carnegie Mellon University, 2000.

[BB85] Errett Bishop and Douglas Bridges. *Constructive analysis*, volume 279 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1985.

[BP03] Vasco Brattka and Gero Presser. Computability on subsets of metric spaces. *Theoretical Comp. Sci.*, 305:43–76, 2003.

[Bra98] Vasco Brattka. *Recursive and Computable Operations over Topological Structures*. PhD thesis, FernUniversität Hagen, 1998.

[BW99] Vasco Brattka and Klaus Weihrauch. Computability on subsets of Euclidean space. I. Closed and compact subsets. *Theoret. Comput. Sci.*, 219(1-2):65–93, 1999. Computability and complexity in analysis (Castle Dagstuhl, 1997).

[CG08] Pieter Collins and Daniel Graça. Effective computability of solutions of ordinary differential equations — the thousand monkeys approach. In *Proceedings of the 5th International Conference on Computability and Complexity in Analysis (CCA'08)*, Electronic Notes in Theoretical Computer Science, pages 53–64. Elsevier, Amsterdam, The Netherlands, 2008.

[Col05] Pieter Collins. Continuity and computability of reachable sets. *Theoret. Comput. Sci.*, 341(1-3):162–195, 2005.

[Col08] Pieter Collins. Computability of controllers for discrete-time semicontinuous systems. In *Proceedings of the 18th International Symposium on the Mathematical Theory of Networks and Systems, Blacksburg, Virginia*, 2008.

[DM70] J. W. Daniel and R. E. Moore. *Computation and Theory in Ordinary Differential Equations*. W. H. Freeman & Co Ltd, 1970.

[EH02] Martín Escardó and Reinhold Heckmann. Topologies on spaces of continuous functions. In *Proceedings of the 16th Summer Conference on General Topology and its Applications (New York)*, volume 26, pages 545–564, 2001/02.

[ELS04] Martín Escardó, Jimmie Lawson, and Alex Simpson. Comparing Cartesian closed categories of (core) compactly generated spaces. *Topology Appl.*, 143(1-3):105–145, 2004.

[Gab07] Grzegorz Gabor. Continuous selection of the solution map for one-sided Lipschitz differential inclusions. *Nonlinear Anal.*, 66(5):1185–1197, 2007.

[GHK$^+$03] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, and D.S. Scott. *Continuous lattices and domains*. Cambridge University Press, 2003.

[Joh02] Peter T. Johnstone. *Sketches of an elephant: a topos theory compendium. Vol. 1*, volume 43 of *Oxford Logic Guides*. The Clarendon Press Oxford University Press, New York, 2002.

[PVB96] Anuj Puri, Pravin Varaiya, and Vivek Borkar. Epsilon-approximation of differential inclusions. In Rajeev Alur, Thomas A. Henzinger, and Eduardo D. Sontag, editors, *Hybrid Systems III*, volume 1066 of *LNCS*, pages 362–376, Berlin, 1996. Springer.

[Ruo96] K. Ruohonen. An effective Cauchy-Peano existence theorem for unique solutions. *Internat. J. Found. Comput. Sci.*, 7(2):151–160, 1996.

[Sch02] Matthias Schröder. *Admissible Representations for Continuous Computations*. PhD thesis, Fachbereich Informatik, FernUniversität Hagen, 2002.

[Tay08] Paul Taylor. A lambda calculus for real analysis. http://www.monad.me.uk/, 2008.

[Wei00] Klaus Weihrauch. *Computable analysis*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2000. An introduction.