Streams, stream transformers and domain representations

Jens Blanck[1], Viggo Stoltenberg-Hansen[1], and John V. Tucker[2]

[1] Dept. of Mathematics, Uppsala University, Box 480, SE-751 06 Uppsala, Sweden
[2] Dept. of Computer Science, University of Wales Swansea, Swansea SA2 8PP, Wales

Abstract. We present a general theory for the computation of stream transformers of the form $F\colon (R \to B) \to (T \to A)$, where time T and R, and data A and B, are discrete or continuous. We show how methods for representing topological algebras by algebraic domains can be applied to transformations of continuous streams. A stream transformer is continuous in the compact-open topology on continuous streams if and only if it has a continuous lifting to a standard algebraic domain representation of such streams. We also examine the important problem of representing discontinuous streams, such as signals $T \to A$, where time T is continuous and data A is discrete.

1 Introduction

1.1 Background

Computing systems are implemented in physical systems, and physical systems are simulated by computing systems. Because of the importance of the applications, there is a need for theoretically sound methods for modelling computations involving the interface between the continuous models of physical processes and the discrete models of algorithmic processes on digital computers. The theory of hardware design can reveal a number of common features between certain classes of computing and physical systems (see [36]). Perhaps the simplest common feature is that computing and physical systems both process streams.

A stream is simply a sequence

$$\ldots, a_t, \ldots$$

of data $a_t \in A$ indexed by time $t \in T$, i.e., a function from T to A. Time may be discrete, when typically it is modelled by the set $\mathbb{N}$ of natural numbers; or time may be continuous, when typically it is modelled by the set $\mathbb{R}_+$ of non-negative real numbers. Time may also be modelled more abstractly.

Most computing systems are designed to operate in discrete time. Their underlying algorithms and architectures are designed to process infinite streams of data such as streams of bits and bytes. Numerous examples of such systems can be found among computers, application specific chips, operating systems and networks. The theoretical development of systolic arrays, dataflow architectures and distributed parallel systems in computing have motivated a great deal of research on stream processing with discrete time. See the recent survey [41], for example. It should also be noted that discrete time streams are basic for computer modelling methods such as neural networks [1], cellular automata [59], and coupled map lattices [8, 26, 28].

Most physical systems operate in continuous time. Their mathematical models can also be viewed as processing continuous streams. For example, partial differential equations specify a function that describes the behaviour of a system from an initial state under streams of input, such as wave forms, boundary conditions, etc. There has been relatively little research on computing with continuous streams though recently the subject has been studied as part of the design of hybrid systems (see e.g. [16]).

In numerical modelling both discrete and continuous stream processing are fundamental. Solution techniques for solving differential equations, such as finite difference and finite element methods, are based on the discretisation of time and space. They involve methods for approximating continuous streams by discrete streams. Both discrete and continuous streams are needed in modelling hybrid systems for control and instrumentation.

In the light of these observations the following problem is important:

> *To create a computability theory for stream processing over abstract data types that can be applied to continuous and discrete models of physical and computing systems.*

1.2 Domain representations of streams and stream transformers

In this paper we present a unified semantic treatment of discrete and continuous streams, and stream transformers. We will focus on problems concerning the continuity and computability of streams and stream transformers, and we will emphasise continuous streams. The framework is based on domain representation theory, which is a theory of representing topological algebras using domains [47]. The topological algebras are used to model data types, and the domains are used to model implementations of data types.

Domain theory is an abstract theory of approximation of spaces and functions, aimed at isolating the structures underlying computation. A domain is an ordered set of approximations on which functions that are continuous in an order-theoretic sense are defined. Of special importance is the fact that domain theory possesses elegant theories of (i) constructions of spaces of continuous functions; and (ii) effectively computable domains. At the heart of the subject is an intimate relation between computability and continuity.

Domain representation theory allows domains to model concrete representations of topological algebras. To a topological algebra A is associated an algebraic domain D_A from which a subset D_A^R is selected to make a representation of A via a continuous map $\nu\colon D_A^R \to A$. On choosing a domain representation, problems about the computability and continuity of functions on topological algebras are translated to corresponding problems about domains (for which there is an excellent theory).

Our unified semantic framework for stream computation is as follows. Let R and T be topological algebras of time, A and B topological algebras of data, and $(R \to B)$ and $(T \to A)$ sets of streams. A *stream transformer* is a function

$$F: (R \to B) \to (T \to A).$$

We will allow T and R to be both discrete and continuous models of time, and allow A and B to be discrete or continuous data types. There are 16 cases in total. Discrete time is identified with the natural numbers $\mathbb{N}$ or the integers $\mathbb{Z}$ (with the discrete topology) and continuous time with the real numbers $\mathbb{R}_+$ or $\mathbb{R}$ (with the Euclidean topology) depending on whether or not there is a starting time. In particular, the framework can be applied to a complete range of computing applications, including analogue-digital transformations of the form

$$F: (\mathbb{R} \to B) \to (\mathbb{N} \to A) \quad \text{and} \quad F: (\mathbb{N} \to B) \to (\mathbb{R} \to A).$$

First, to each of the four algebras R, T, A and B we associate a domain representation D_R, D_T, D_A and D_B respectively. The domains of continuous functions $[D_R \to D_B]$ and $[D_T \to D_A]$ naturally represent stream spaces $(R \to B)$ and $(T \to A)$. Most often we will consider spaces of continuous streams, denoted $\mathrm{C}(R \to B)$ and $\mathrm{C}(T \to A)$. We will also consider some spaces of non-continuous streams, in particular non-zeno signals (see Section 2.7).

Next, for a stream transformer F we show how to construct a continuous function on the domains

$$\bar{F}: [D_R \to D_B] \to [D_T \to D_A]$$

representing F. For example, we will show (Corollary 6.9):

> *A stream transformer* $F: \mathrm{C}(\mathbb{R} \to \mathbb{R}) \to \mathrm{C}(\mathbb{R} \to \mathbb{R})$ *is continuous with respect to the compact-open topology if, and only if,* F *has a continuous representation or lifting* $\bar{F}: [\mathcal{R} \to \mathcal{R}] \to [\mathcal{R} \to \mathcal{R}]$, *where* $\mathcal{R}$ *is the standard interval domain representation of* $\mathbb{R}$.

The study of properties of F, such as computability, is thus reduced to the study of properties of $\bar{F}$.

Despite the smooth theory of domains, this process is not straight-forward because of the relation:

computability implies continuity.

One problem we must deal with is that our unified framework must cope with computing with non-continuous streams.

For example, consider the sets of continuous and discrete data streams in continuous time:

$$(\mathbb{R} \to [0,1]) \text{ and } (\mathbb{R} \to \{0,1\}).$$

In the first case, many applications require models based on the subset of continuous streams (e.g., wave functions). However, in the second case every non-trivial application involving discrete signals will need discontinuous functions (since the only continuous functions $\mathbb{R} \to \{0,1\}$ are constant functions). Thus, to explicate computability, we must find some reasonable notion of representation of non-continuous streams. In particular we consider non-zeno signals $\mathrm{NZ}(\mathbb{R}_+ \to A)$ consisting of step functions, where A is a discrete space. We show that, for certain classes of stream transformers, $F\colon \mathrm{NZ}(\mathbb{R}_+ \to B) \to \mathrm{NZ}(\mathbb{R}_+ \to A)$ has a continuous *approximate* representation $\bar{F}\colon [\mathcal{R}_+ \to B_\perp] \to [\mathcal{R}_+ \to A_\perp]$.

In this paper we will explain the above mathematical framework. In Section 2 the basic ideas about streams and stream transformers will be defined and several examples of transformations of time and data in streams will be presented to motivate the technical development. In Section 3 we summarise the essential ideas about algebraic domains, function spaces and effective domains that we will use. In Section 4 we explain the method of domain representation for topological algebra, including the problem of approximate representations of discontinuous functions. In Section 5 we consider domain representations of stream spaces. Finally, in Section 6 we show how domain representations of stream transformations for continuous streams and for non-zeno signals are chosen and applied.

This paper is part of a series of articles on the theory of domain representations of topological algebras [43–47]. We first considered discrete time stream computation in [46]. We thank our colleagues in the NADA Working Group for the several stimulating debates on the problems of stream computation starting with the debate led by Jan Bergstra at the first NADA meeting at Munich 1994 and the NADA stream workshop 1995 hosted by Helmut Schwichtenberg and Hans Leiss at Elmau. We also have benefitted from discussions on the subject with Jeff Zucker (McMaster), and Neal Harman and Matthew Poole (Swansea).

2 Basic properties of streams and stream transformers

2.1 General definitions

There are several notions of streams and stream transformers in the literature. In this paper we will restrict ourselves to the simplest and, in our view, the most natural notion.

Definition 2.1. Let T be a set of data modelling time and let A be any non-empty set of data. A *stream* is a total function $\varphi\colon T \to A$. The set of all streams from T to A is called the *complete stream space* from T to A. A *stream space* from T to A is simply a subset of the complete stream space from T to A and is usually denoted $(T \to A)$.

This definition is quite open as to what requirements one should put on time T. There are many philosophical, physical and mathematical models of time [58]. Time can be discrete or continuous. To model discrete time we choose the ordered structure of natural numbers $\mathbb{N}$, in the case time has an initial starting point, or $\mathbb{Z}$, in case there is no starting point. For simplicity in the exposition we will usually model discrete time by $\mathbb{N}$. Thus a stream with discrete time is for us simply an infinite sequence of data.

To model continuous time we choose the ordered structure of non-negative real numbers $\mathbb{R}_+$, in the case time has an initial starting point, or $\mathbb{R}$, in case there is no starting point. For simplicity we will usually model continuous time by $\mathbb{R}$, except when an initial starting point is essential. A stream with continuous time is often called a *signal*.

There are well-established notions of computability on $\mathbb{N}$ and $\mathbb{R}$ and hence on our chosen models of discrete and continuous time. Classical models of computability on the natural numbers are well-known [9, 15, 40]. Computability models on $\mathbb{R}$ have been developed since the 1950's [7, 17, 29, 37].

Our data set A is in general simply a set or an algebra. In order to discuss computability of streams and stream transformers we need to have notions of computability on our data algebras as well.

For discrete algebras we have the usual notion of computability induced by the computability on $\mathbb{N}$ in the sense of computable algebra [12, 30, 38, 47, 49].

For uncountable algebras we need a notion of computability in terms of concrete approximations. Topology can be seen as an abstract theory of approximation where an open set approximates all its elements. Usually, but not always, our data type is a metric algebra. Of course, every discrete space can be given a discrete metric inducing the discrete topology.

Assumption 2.2. *All our data types are topological algebras.*

There are several approaches to computability on topological algebras including Type-2 computability by Weihrauch [57], recursive metric spaces by Moschovakis [34], and the approach chosen in this paper, domain representation. See [48] for a discussion of the relationship between these and other approaches.

Having topologies on both time and data allows us to talk about continuity of streams. Note that each discrete time stream $\varphi\colon \mathbb{N} \to A$ is continuous for any space A. Complete stream spaces with continuous time T and non-trivial data set A will contain non-continuous streams. When A is a continuous data set, such as $\mathbb{R}$, it is natural to consider the stream space $\mathrm{C}(T \to A)$ of continuous streams. The stream space $\mathrm{C}(T \to A)$ has a natural topology, viz. the the compact-open topology. Note that there is no obvious or canonical topology for the complete stream space $(T \to A)$ for continuous time T.

Consider streams from continuous time into a discrete data set, i.e., signals. Then the only continuous streams are the constant streams. Non-continuous streams from continuous time to discrete data exist in abundance in models used in computing. Thus our theory of streams and stream transformations must accommodate them. In order to model non-continuous streams we use *approximate* representations in the function space domains. In this way stream spaces containing non-continuous streams (more precisely quotients of such) obtain an induced topology from the representing domains. Thus we may, via the use of domains, speak about continuity of stream transformers also in this case.

We now turn to the notion of a stream transformer.

Definition 2.3. Let $(R_i \to B_i)$ and $(T_j \to A_j)$ be stream spaces for $1 \le i \le m$ and $1 \le j \le n$. A *stream transformer* is a functional $F: \prod_{1 \le i \le m}(R_i \to B_i) \to \prod_{1 \le j \le n}(T_j \to A_j)$.

Thus a stream transformer is a function which takes finitely many streams as input and gives finitely many streams as output. Usually, for the simplicity of the exposition, we will restrict ourselves to the case $m = n = 1$. This is not as restrictive as it first may appear. It is often the case that all input times R_i can be identified with some common input time R and similarly for T_j and T. In this case we have

$$\prod_{1 \le i \le m} (R_i \to B_i) \cong (R \to B_1 \times \cdots \times B_m), \text{ and}$$
$$\prod_{1 \le j \le n} (T_j \to A_j) \cong (T \to A_1 \times \cdots \times A_n)$$

and we may consider F to be a stream transformer having one stream as input and giving one stream as output.

The value $F(\varphi)(t)$ of a stream transformer $F: (R \to B) \to (T \to A)$ at time t may in general depend on the entire input stream φ. This is not reasonable for stream transformers modelling physical devices. Also from a computability point of view it is unreasonable to require infinite information in order to compute a finite object.

Assume that time R has an initial point 0. Then a stream transformer $F: (R \to B) \to (T \to A)$ is said to satisfy "causality", or is "finitely determined", if for each $t \in T$ there is an $r \in R$ such that the value of $F(\varphi)$ at t is determined by φ restricted to the interval between 0 and r.

It is clear that causality of a stream transformer F is intimately connected with the "continuity" of F. The latter will be defined via the domain representation of stream transformers.

2.2 Some examples of streams and stream transformers

First we consider some examples of streams. Then we give a few general forms of stream transformers in which the main idea is to separate time conversions from data conversions. We start with a simple model and then give some extensions. In each case the models will first be motivated by an example.

Discrete time streams. Streams of the form $f: \mathbb{N} \to A$, where A is a non-empty set, occur *throughout* computing. Hardware systems are modelled using streams and stream transformers: at low levels of digital design, the data sets of bits and k-bit words,

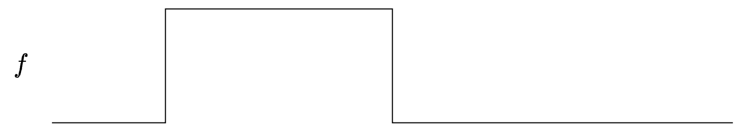
$$\text{Bit} = \{0, 1\} \text{ and } \text{Word}_k = \text{Bit}^k,$$

are used to represent integers, addresses, flags, reals, pixels, etc. Devices operate in time using one or more discrete clocks, each represented by $\mathbb{N}$. There input-output behaviours are modelled by stream transformations of the form

$$F: [\mathbb{N} \to \mathrm{Bit}]^n \to [\mathbb{N} \to \mathrm{Bit}]^m.$$

For example, a thorough study of modelling a digital correlator is given in [19]. Other examples are in [18, 22]. Discrete time streams and stream transformers are widely used in modelling software. One comprehensive approach, FOCUS, has been developed by Broy and his co-workers [6]. There also finite partial streams are used.

Signals. Streams of the form $f: \mathbb{R} \to \{0, 1\}$ are used in digital signal processing and are drawn as square waves



These streams do not exist in the physics of devices but are abstractions or specifications of certain streams of the form $\varphi: \mathbb{R} \to \mathbb{R}$. Transitions in voltage (or current) are "modelled" by discontinuities.

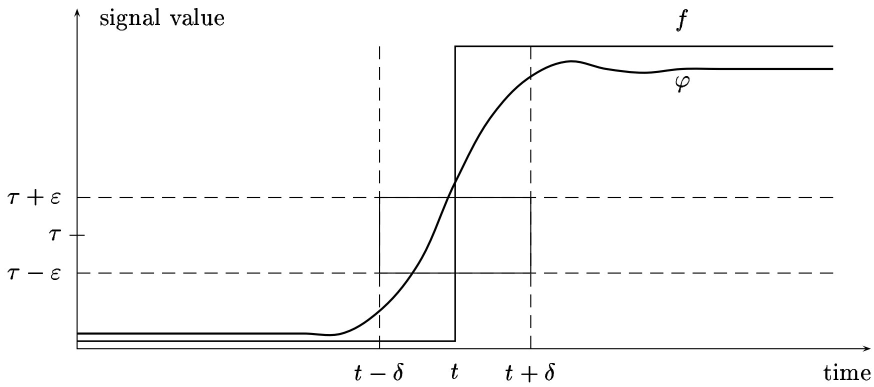


Fig. 1. A digital signal φ is modelled by a square wave form f.

A signal transition from low to high in f takes place at time t when the continuous wave φ passes a threshold value τ. Furthermore, the threshold value τ is a physical quantity that is not known exactly but to within a range

$$[\tau - \varepsilon, \tau + \varepsilon];$$

and the time t is not known exactly but within a range

$$[t - \delta, t + \delta].$$

Any waveform φ that enters the threshold range during the time range has the same square wave abstraction, see Figure 1. A device technology determines some ε and δ such that it is technically impossible to distinguish two (reasonable, steep) wave forms that pass through the box.

Single access stream transformers. Let us look at a very simple type of stream transformation. Suppose that a stream transformer simply transforms its input data at some point in time and outputs the result at some later time.

Example 2.4. Consider $G: (\mathbb{N} \to \mathbb{R}) \to (\mathbb{N} \to \mathbb{R})$, given by

$$G(\varphi)(t) = \begin{cases} 0, & \text{if } t = 0; \\ 2\varphi(t-1), & \text{otherwise.} \end{cases}$$

Notice that we are confronted by a boundary problem at $t = 0$, which is handled by giving an arbitrary value. So let us now suppose that $t > 0$. The first thing to do when given a stream and a time at which to compute the new stream is to calculate the time of interest in the input stream by a function $\tau(t) = t - 1$. Secondly, use the input stream φ to get the input value at the appropriate time. Thirdly, calculate the output data from the input data by means of the function $\pi(n) = 2n$.

The following definition gives us a way of constructing stream transformers of this form, which we call *single access stream transformers.*

Definition 2.5. (i) Let the functional

$$\Phi: (R \to B) \times (T \to R) \times (B \to A) \to (T \to A),$$

be defined by

$$\Phi(\varphi, \tau, \pi)(t) = \pi(\varphi(\tau(t))).$$

(ii) A stream transformer $F: (R \to B) \to (T \to A)$ is of *single access* type if there exist a *time transformation* τ and a *data transformation* π such that F is given by

$$F(\varphi) = \Phi(\varphi, \tau, \pi).$$

The G of Example 2.4 can now be expressed as

$$G(\varphi) = \Phi(\varphi, \tau, \pi),$$

where τ and π are the transformations extracted in the example.

By varying the functions τ and π we get a whole family of stream transformers. Of course, not all stream transformers are of this form.

The important feature of the single access model is that it allows us to discuss time and data transformations independently.

Multiple access stream transformers. We model stream transformers that depend on the input data at several distinct times.

Example 2.6. Consider the Fibonacci stream transformer $G: (\mathbb{N} \to \mathbb{N}) \to (\mathbb{N} \to \mathbb{N})$, defined by

$$G(\varphi)(t) = \begin{cases} t, & \text{if } t \leq 1; \\ \varphi(t-1) + \varphi(t-2), & \text{if } t \geq 2. \end{cases}$$

Clearly, G does not fit into the single access model described above since it accesses the input stream at two different times. Thus, we have two time transformations, τ_1 and τ_2, and a data transformation π taking two input data. The time and data transformations are

$$\begin{aligned} \tau_1(t) &= t - 1, \\ \tau_2(t) &= t - 2, \text{and} \\ \pi(x, y) &= x + y. \end{aligned}$$

Here is a generalisation of Definition 2.5 to a finite number of accesses to the input stream.

Definition 2.7. (i) Let the functional

$$\Phi_n: (R \to B) \times (T \to R)^n \times (B^n \to A) \to (T \to A),$$

be defined by

$$\Phi_n(\varphi, \tau_1, \ldots, \tau_n, \pi)(t) = \pi(\varphi(\tau_1(t)), \ldots, \varphi(\tau_n(t))).$$

(ii) A stream transformer $F: (R \to B) \to (T \to A)$ is of *multiple access* type if there exist *time transformations* $\tau_1, \ldots, \tau_n$ and a *data transformation* π such that F is given by

$$F(\varphi) = \Phi_n(\varphi, \tau_1, \ldots, \tau_n, \pi).$$

Note that $\Phi = \Phi_1$ for Φ as in Definition 2.5.

The G of Example 2.6 can now be expressed as

$$G(\varphi) = \Phi_2(\varphi, \tau_1, \tau_2, \pi),$$

where τ_1, τ_2 and π are the transformations extracted in the example.

Accessing an interval of the input stream.

Example 2.8. Let $G: (\mathbb{R} \to \mathbb{R}) \to (\mathbb{R} \to \mathbb{R})$ be defined by

$$G(f)(t) = \max_{x \in [0,t]} f(x).$$

The stream transformer G depends on a continuum of values of the input stream so there is no possibility of modifying the single access model in the way done above for a finite number of accesses to the input stream.

If we generalise from the example above we could allow both endpoints of the interval to depend on t, making the data transformation to be of the following kind

$$\pi\colon (R \to B) \times R \times R \to A,$$

where π is defined by

$$\pi(f, a, b) = \max_{x \in [a,b]} f(x).$$

Instead of max we could have any of a number of common operations, e.g., the integral

$$\pi(f, a, b) = \int_b^a f(x)dx.$$

Definition 2.9. (i) Let the functional

$$\Psi\colon (R \to B) \times (T \to R)^2 \times ((R \to B) \times R^2 \to A) \to (T \to A)$$

be defined by

$$\Psi(f, \tau_1, \tau_2, \pi)(t) = \pi(f, \tau_1(t), \tau_2(t)).$$

(ii) A stream transformer $F\colon (R \to B) \to (T \to A)$ is of *interval access* type if there exist *time transformations* τ_1, τ_2, and *data transformation* π with the property that $f = g$ on $[a, b]$ implies $\pi(f, a, b) = \pi(g, a, b)$, such that F is given by

$$F(f) = \Psi(f, \tau_1, \tau_2, \pi).$$

The stream transformer G of Example 2.8 can now be expressed as

$$G(f) = \Psi(f, \tau_1, \tau_2, \pi),$$

where τ_1, τ_2 and π are the transformations extracted in the example.

2.3 Time transformations

In the models exhibited so far we have extracted time and data transformations as parts of stream transformers. The data transformations considered are arbitrary functions on data. However, time transformations are of special interest and we will now look at some of the most fundamental time transformations.

Constant delay is modelled by a time transformation of the form

$$\tau(t) = t - d,$$

where d is the constant delay. Hence a simple delay node is

$$\Phi(\varphi, \tau, \mathrm{id}).$$

We can adjust for two clocks running at different speeds by a time transformation of the form

$$\tau(t) = kt,$$

where k is the number of time units that passes on the input clock during one time unit on the output clock. Hence, using the single access functional Φ of Definition 2.5, a stream transformer which outputs every other datum of a discrete stream is given by

$$\Phi(\varphi, \tau, \mathrm{id}),$$

where $\tau(t) = 2t$.

If the input and output clocks run at the same speed but are out of phase, then this can be modelled by

$$\tau(t) = t - m,$$

where m is the time offset. The offset is positive if the output time is ahead of the input time. There is a difference between a delay and the offset considered here since a delay cannot be negative whereas the time offset may be negative.

We have exhibited three different time transformations which are linear and can be used regardless of time being discrete or continuous. However, sometimes it is desirable to consider non-linear time transformations.

Here is a useful general class of time transformations.

Definition 2.10. A mapping $\tau : T \to R$ is a *retiming* if $\tau(0) = 0$ and τ is surjective and monotonic with respect to the orderings on T and R.

In the case that T and R are discrete clocks this notion of retiming is easy to understand and useful in both theoretical investigations and design exercises. It was introduced in a study of the design of digital correlators and further developed through applications to UARTs and micro processors, see [18, 19, 22, 23, 25].

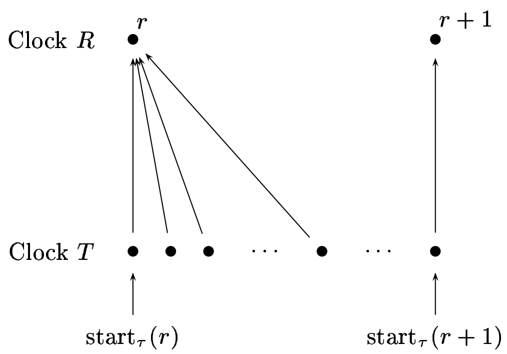


Fig. 2. A retiming τ.

A retiming $\tau{:}T \to R$ relates discrete clocks as in Figure 2. The figure illustrates the set

$$\tau^{-1}(r) = \{t \in T : \mathrm{start}_\tau(r) \leq t < \mathrm{start}_\tau(r+1)\}$$

where $\mathrm{start}_\tau(r) = (\text{least } t)[\tau(t) = r]$.

2.4 State transformations

Consider a server responding to requests. We assume that the requests appear at discrete times. The requests and the responses are easily seen to be streams, leaving us with the conclusion that the server should be modelled as a stream transformer.

The server typically has an internal state which governs the responses to the requests. We consider a server that has an internal state and is working in discrete time, i.e., $T = \mathbb{N}$.

Let I be a set of requests, O a set of responses and S a set of internal states. Suppose the server's transitions are governed by the functions

$$\begin{aligned} \mathrm{out} &: S \times I \to O, \text{ and} \\ \mathrm{next} &: S \times I \to S, \end{aligned}$$

which determine the output and next state from the input and current state.

In responding to a stream of requests in discrete time, starting from some initial state, the dynamical behaviour of the server Φ is governed by

$$\Phi_{\mathrm{state}}{:}\, S \times (T \to I) \to (T \to S)$$

$$\Phi_{\mathrm{state}}(s,i)(t) = \begin{cases} s, & \text{if } t = 0; \\ \mathrm{next}(\Phi_{\mathrm{state}}(s,i)(t-1), i(t-1)), & \text{if } t > 0, \end{cases}$$

and

$$\Phi_{\mathrm{out}}{:}\, S \times (T \to I) \to (T \to O)$$

$$\Phi_{\mathrm{out}}(s,i)(t) = \begin{cases} \text{undefined}, & \text{if } t = 0; \\ \mathrm{out}(\Phi_{\mathrm{state}}(s,i)(t-1), i(t-1)), & \text{if } t > 0. \end{cases}$$

The typing of the functions Φ_{state} and Φ_{out} does not fall within our definition of stream transformers since we have allowed them to take arguments other than streams as parameters.

2.5 The analogue to digital converter

The analogue to digital converter (AD-converter) is a device which samples an analogue electrical signal at discrete times and gives discrete approximations of the analogue signal at those times.

We start by giving a very simple model of an idealistic AD-converter.

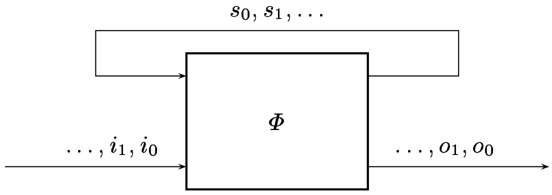


Fig. 3. Stream transformer with an internal state.

Example 2.11. Combining the use of the floor function both as a time transformation and as a data transformation in the single access model will give us a very simple model of an AD-converter. Let $G\colon (\mathbb{R}_+ \to \mathbb{R}) \to (\mathbb{N} \to \mathbb{Z})$ be defined by

$$G(\varphi) = \Phi(\varphi, \tau, \pi),$$

where $\tau(x) = \pi(x) = \lfloor x \rfloor$. Then G is a simple AD-converter.

We will now try to model a hardware AD-converter more faithfully. A hardware AD-converter will first bound the input signal to some closed interval. It will then sample the input stream at discrete points in time. More advanced converters will sample the input stream at several time points for every value it outputs, this is called *oversampling.* The output value is then calculated from the sampled values by some filtering algorithm.

Note that we have not made any effort to model the anomalous behaviour that hardware can exhibit if the input is widely out of range.

Example 2.12. We will model a 16-bit AD-converter with an output rate of 10 kHz, an oversampling factor of 100, and which converts the interval 0 V to 5 V.

The bounding step now amounts to bounding the input signal (stream) to the interval 0 V to 5 V. Let $B\colon (\mathbb{R}_+ \to \mathbb{R}) \to (\mathbb{R}_+ \to \mathbb{R})$ be defined by

$$B(\varphi) = \Phi(\varphi, \mathrm{id}, \pi),$$

where π is given by

$$\pi(x) = \max(0, \min(x, 5)).$$

Clearly, B is a stream transformer bounding the input stream in the desired way.

The sampling step can now be modelled similar to our trivial AD-converter example above. We model the sampling step by $S\colon (\mathbb{R}_+ \to \mathbb{R}) \to (\mathbb{N} \to \mathbb{Z})$ given by

$$S(\varphi) = \Phi(\varphi, \tau, \pi),$$

where $\tau(t) = \lfloor 10^6 \cdot t \rfloor$ and $\pi(x) = \lfloor \frac{2^{16} \cdot x}{5} \rfloor$. The constant 10^6 corresponds to the output frequency times the oversampling factor. The constant 2^{16} corresponds to 16 bit conversion. The constant 5 is the adjustment for the voltage interval.

The filtering step will become a multiple access stream transformer. We model the filtering by $F\colon (\mathbb{N} \to \mathbb{Z}) \to (\mathbb{N} \to \mathbb{Z})$ given by

$$F(\varphi) = \Phi_{100}(\varphi, \tau_0, \ldots, \tau_{99}, \pi),$$

where $\tau_i(t) = 100t + i$ and $\pi(x_0, \ldots, x_{99})$ is the filtering function. The filtering function will typically remove values that are far from the average value and then compute the average of the remaining values.

Our hardware AD-converter can now simply be modelled by composing the bounding, sampling and filtering stream tranformers given above. Thus we define our AD-converter $AD\colon (\mathbb{R}_+ \to \mathbb{R}) \to (\mathbb{N} \to \mathbb{Z})$ by

$$AD = F \circ S \circ B.$$

2.6 DA-conversion or curve fitting

A digital signal φ can be seen as a function from $\mathbb{N}$ to some discrete set (usually a finite set). We assume that the values of the digital signal form a finite subset of $\mathbb{R}$. We also assume that the discrete time is embedded into continuous time by an increasing function τ. Hence every pair $(\tau(n), \varphi(n))$ is a point in the space $\mathbb{R}_+ \times \mathbb{R}$. To convert a digital signal to an analogue signal we need to extend the enumerable set of points in $\mathbb{R}_+ \times \mathbb{R}$ to a function in $(\mathbb{R}_+ \to \mathbb{R})$. Figure 4 indicates a few alternatives of how a digital signal φ may be converted into a continuous signal. The first is to extend the point set to a step function (this is the normal description of a hardware DA-converter), the second makes a linear approximation and the third uses polynomials of degree three.

It is easy to describe the DA-converting stream transformers above. This is done for the linear case in the example below.

Example 2.13. The following stream transformer converts a digital signal to a linear analogue signal. Let $G\colon (\mathbb{N} \to \mathbb{Z}) \to (\mathbb{R} \to \mathbb{R})$ be given by

$$G(\varphi)(t) = \begin{cases} t\varphi(0), & \text{if } t < 1; \\ \varphi(\lfloor t-1 \rfloor) + (t - \lfloor t \rfloor)(\varphi(\lfloor t \rfloor) - \varphi(\lfloor t-1 \rfloor)), & \text{otherwise.} \end{cases}$$

It is easy to see that there are two time transformations in operation here, namely, $\tau_1(t) = \lfloor t-1 \rfloor$ and $\tau_2(t) = \lfloor t \rfloor$. However, the data transformation does not depend only on the input data at both of these times, but also on the time t. Hence this stream transformer does not fall into any of the previously discussed general forms of stream transformers.

2.7 Signal transformations

A *signal* is a stream with continuous time and discrete data. Normally, when considering signals, we consider time with a starting point, i.e. time is $\mathbb{R}_+$. Thus a signal is a function $\varphi\colon \mathbb{R}_+ \to A$ where A is a discrete data set. A *signal transformer* is a stream transformer taking signals to signals. Signal transformers are sometimes called *signal operators.*

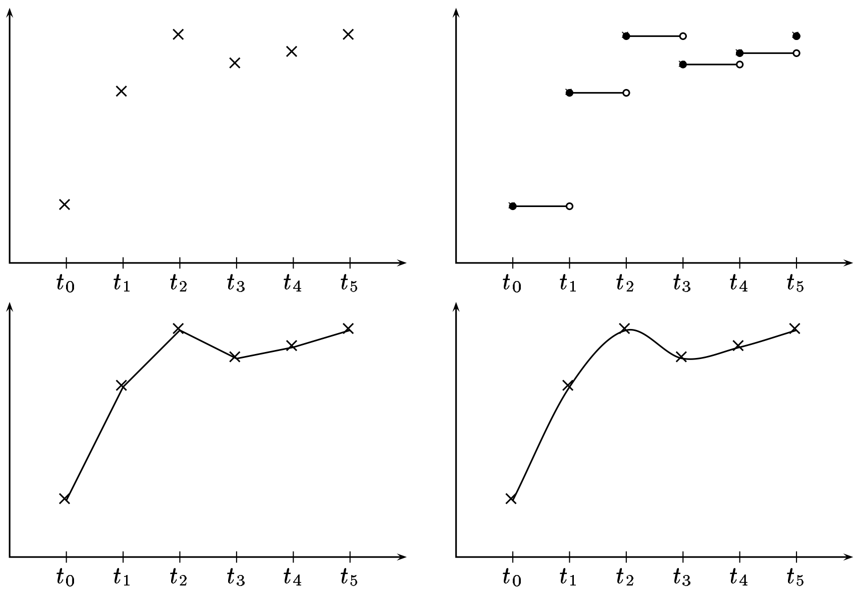


Fig. 4. DA-conversion of a digital signal φ.

Signals and operators on signals have been much studied in the literature from a computer science point of view, in particular in connection with hybrid systems [16]. Automata over continuous time is considered in [51, 39].

As mentioned earlier, the only continuous functions from $\mathbb{R}_+$ into a discrete data set A are the constant functions. Thus the class of continuous signals is no more interesting than the data set A. On the other hand it should be clear, and will be apparent later, that the class of all signals is too wide when considering signal operators. For example, the signal taking the value 0 at rational time points and the value 1 at irrational time points is too irregular to be distinguished by a reasonable operator.

¿From the discussion of signals in Section 2.2 we see that in order to model digital systems it suffices to consider the following subclass of signals.

Definition 2.14. Let A be a discrete countable set. A *non-zeno signal* over A is a stream $\varphi: \mathbb{R}_+ \to A$ such that φ is continuous at 0 and for each $t > 0$, φ has only finitely many discontinuities in $[0, t]$.

Thus a signal is a step function with finitely many jumps on $[0, t]$.

We now consider single and multiple access signal operators. We need to slightly strengthen the notion of retiming (Definition 2.10).

Definition 2.15. A mapping $\tau: \mathbb{R}_+ \to \mathbb{R}_+$ is a *strict retiming* if

(i) τ is surjective and monotonic;

(ii) $\tau(0) = 0$; and
(iii) $\tau(t) < \tau(t')$ whenever $\text{Init}(\tau) \leq t < t'$, where $\text{Init}(\tau) = \sup\{t \in \mathbb{R}_+ : \tau(t) = 0\}$.

Note that a strict retiming is continuous.

Theorem 2.16. *Let $F: (\mathbb{R}_+ \to B) \to (\mathbb{R}_+ \to A)$ be a single access signal operator with repect to $\pi: B \to A$ and a strict retiming $\tau: \mathbb{R}_+ \to \mathbb{R}_+$. Then F takes non-zeno signals to non-zeno signals.*

Proof. Let $\varphi: \mathbb{R}_+ \to B$ be non-zeno. Then

$$F(\varphi)(t) = \pi(\varphi(\tau(t))).$$

Assume that φ is discontinuous at $t_0 < t_1 < \dots$. Then $\varphi\tau$ is discontinuous at precisely $\tau^{-1}(t_0) < \tau^{-1}(t_1) < \dots$, and hence the set of discontinuities of $F(\varphi)$ is contained in $\{\tau^{-1}(t_0), \tau^{-1}(t_1), \dots\}$. Now, $t_0 > 0$ since φ is non-zeno, so $\tau^{-1}(t_0) > 0$, i.e. $F(\varphi)$ is continuous at 0. Similarly, by the monotonicity of τ, $\tau^{-1}(t_0) < \tau^{-1}(t_1) < \dots$ is unbounded in $\mathbb{R}_+$, unless finite. □

The theorem clearly extends to multiple access signal operators.

3 Domains

In this section we will briefly review some basic and relevant parts of domain theory. We concentrate on giving the notions and some results that are needed for our analysis. All proofs are omitted and can be found in the basic reference [42].

3.1 Preliminaries on domains

Let $D = (D, \sqsubseteq)$ be a partial order and let $A \subseteq D$. We will use the notation $\uparrow A$ to denote the set $\{y \in D : \exists x \in A(x \sqsubseteq y)\}$. The set $\uparrow\{x\}$ is abbreviated by $\uparrow x$. We define $\downarrow A$ and $\downarrow x$ dually. A is *directed* if $A \neq \emptyset$ and whenever $x, y \in A$ then there is $z \in A$ such that $x \sqsubseteq z$ and $y \sqsubseteq z$. The supremum, or least upper bound, of A (if it exists) is denoted by $\bigsqcup A$. As usual we write $x \sqcup y$ instead of $\bigsqcup\{x, y\}$.

A *complete partial order*, abbreviated *cpo*, is a partial order, $D = (D; \sqsubseteq, \bot)$, such that $\bot$ is the least element in D and any directed set $A \subseteq D$ has a supremum $\bigsqcup A$ in D.

Let D be a cpo. Then an element $a \in D$ is *compact* if whenever $A \subseteq D$ is a directed set and $a \sqsubseteq \bigsqcup A$, then $a \in \downarrow A$. The set of compact elements of D is denoted by D_c.

A cpo D is *algebraic* if for each $x \in D$, the set

$$\text{approx}(x) = \downarrow x \cap D_c$$

is directed and $x = \bigsqcup \text{approx}(x)$. A cpo D is *consistently complete* if $\bigsqcup A$ exists in D whenever $A \subseteq D$ is a consistent set, i.e., has an upper bound.

The domains we consider in this paper are of the following kind.

Definition 3.1. A *Scott–Ershov domain*, or simply *domain*, is a consistently complete algebraic cpo.

The topology normally used on domains is called the *Scott topology*. Let D be an algebraic cpo. Then $U \subseteq D$ is *open* if

(i) $x \in U$ and $x \sqsubseteq y$ implies $y \in U$, and
(ii) $x \in U$ implies that there exists $a \in \text{approx}(x)$ such that $a \in U$.

An easy observation is that the Scott topology on a non-trivial domain is T_0 but not T_1. Furthermore, the sets $\uparrow a$ for $a \in D_c$ constitute a base for the Scott topology on D. We will also write B_a for $\uparrow a$.

Let D and E be cpo's. A function $f: D \to E$ is continuous with respect to the Scott topology if, and only if, f is monotone and

$$f(\bigsqcup A) = \bigsqcup f[A],$$

for any directed set $A \subseteq D$.

Any continuous function between domains is determined by its values on the compact elements. In fact, let D and E be domains. Then a monotone function $f: D_c \to E$ has a unique extension to a continuous function $g: D \to E$ such that $f = g|_{D_c}$.

A *conditional upper semi-lattice with least element*, abbreviated *cusl*, is a partially ordered set where each finite bounded set has a least upper bound. The set of compact elements D_c of a domain D forms a cusl. Every domain is obtained as a completion of a cusl in the following way.

Definition 3.2. Let P be a cusl. Then $I \subseteq P$ is an *ideal* if

(i) $I \neq \emptyset$,
(ii) $a \in I$ and $b \sqsubseteq a$ implies $b \in I$, and
(iii) $a, b \in I$ implies $a \sqcup b$ exists and $a \sqcup b \in I$.

For $a \in P$ we let $[a]$ denote the principal ideal generated by a. The ideal completion of a cusl P is the set of all ideals over P, denoted $\text{Idl}(P)$. When ordered by set inclusion the ideal completion of a cusl is a domain. The compact elements of $\text{Idl}(P)$ are the principal ideals $[a]$, for $a \in P$.

The representation theorem for Scott–Ershov domains tells us that any domain is the ideal completion of a cusl.

Theorem 3.3. *Let D be a Scott–Ershov domain. Then* $\text{Idl}(D_c) \cong D$.

We clearly have the following equivalence, for $I \in \text{Idl}(P)$

$$[a] \subseteq I \iff a \in I.$$

Thus the basic open sets of $\text{Idl}(P)$ in the Scott topology are of the form $B_a = \{I \in \text{Idl}(P) : a \in I\}$ for $a \in P$.

The class of domains has pleasing closure properties. In particular, what is essential for our purposes, the category of domains is cartesian closed. Most importantly, this means that for any domains D and E, the function space

$$[D \to E] = \{f: D \to E \mid f \text{ is continuous}\}$$

is a domain, where the ordering $\sqsubseteq$ on $[D \to E]$ is given by $f \sqsubseteq g \iff (\forall x \in D)(f(x) \sqsubseteq g(x))$.

We recall some basic facts, and notations, of the compact elements in $[D \to E]$. For $a \in D_c$ and $b \in E_c$ we consider the "step function" $\langle a; b\rangle: D \to E$ defined by

$$\langle a; b\rangle(x) = \begin{cases} b, & \text{if } a \sqsubseteq x; \\ \bot, & \text{otherwise.} \end{cases}$$

Then $\langle a; b\rangle$ is continuous and compact. Furthermore, for any $f \in [D \to E]$ we have

$$\langle a; b\rangle \sqsubseteq f \iff b \sqsubseteq f(a).$$

The compact elements of $[D \to E]$ are those of the form

$$\bigsqcup_{i=1}^{n} \langle a_i; b_i\rangle,$$

where $\{\langle a_i; b_i\rangle : i = 1, \ldots, n\}$ is *consistent* (i.e. bounded) in $[D \to E]$. And the latter holds if, and only if, for each $I \subseteq \{1, \ldots, n\}$,

$$\{a_i : i \in I\} \text{ consistent} \implies \{b_i : i \in I\} \text{ consistent.}$$

Recall that consistent completeness is used to prove the above properties. In fact, the class of algebraic cpo's is *not* closed under the function space construction.

3.2 Effective domains

In this section we briefly recall some basic notions of effectivity or computability on domains. We start by recalling some general notions.

A *structure* A is a tuple $A = (A; R_1, \ldots, R_p; \sigma_1, \ldots, \sigma_q)$, where A is a non-empty set, $R_j \subseteq A^{n_j}$ is an n_j-ary relation and $\sigma_i: A^{n_i} \to A$ is an n_i-ary operation on A. A *numbering* of a structure A is a surjective function $\alpha: \Omega_\alpha \to A$, where $\Omega_\alpha \subseteq \omega$. Let $\equiv_\alpha$ denote the equivalence relation defined on Ω_α by

$$m \equiv_\alpha n \iff \alpha(m) = \alpha(n).$$

Definition 3.4. Let $\alpha: \Omega_\alpha \to A$ be a numbering of the structure $A = (A; R_1, \ldots, R_p; \sigma_1, \ldots, \sigma_q)$. Then α is a *weakly effective numbering* of A and the pair (A, α) is a *weakly effective structure* if (i) and (ii) below hold.

(i) For each $i = 1, \dots, q$ there is an n_i-ary partial recursive function $\hat{\sigma}_i$ such that for each $m_1, \dots m_{n_i} \in \Omega_\alpha$, $\hat{\sigma}_i(m_1, \dots, m_{n_i})\downarrow$ (where $\downarrow$ means defined) and

$$\sigma_i\big(\alpha(m_1), \dots, \alpha(m_{n_i})\big) = \alpha\big(\hat{\sigma}_i(m_1, \dots, m_{n_i})\big).$$

(ii) For each $j = 1, \dots, p$ there is an n_j-ary recursive relation $\hat{R}_j$ such that for each $m_1, \dots, m_{n_j} \in \Omega_\alpha$,

$$R_j\big(\alpha(m_1), \dots, \alpha(m_{n_j})\big) \iff \hat{R}_j(m_1, \dots, m_{n_j}).$$

We say that $\hat{\sigma}_i$ and $\hat{R}_j$ *track* σ_i and R_j respectively.

Definition 3.5. A weakly effective numbering α of a structure A is *computable* if Ω_α is a recursive set and $\equiv_\alpha$ is a recursive relation. If α is computable then the pair (A, α) is a *computable structure*.

Let (A, α) and (B, β) be weakly effective structures. A function $f: A \to B$ is (α, β)-*computable* if there is a partial recursive function $\hat{f}$ such that $\Omega_\alpha \subseteq \mathrm{dom}(\hat{f})$ and for each $m \in \Omega_\alpha$, $f(\alpha(m)) = \beta(\hat{f}(m))$.

A partial function $g: A \to B$ is (α, β)-*computable* if there is a partial recursive function $\hat{g}$ such that $\mathrm{dom}(g \circ \alpha) \subseteq \mathrm{dom}(\hat{g})$ and $g(\alpha(m)) = \beta(\hat{g}(m))$, for $m \in \mathrm{dom}(g \circ \alpha)$.

A set $C \subseteq A$ is α-*decidable* (α-*semidecidable*) if there is a recursive (r.e.) set W such that $\alpha^{-1}[C] = W \cap \Omega_\alpha$. A recursive (r.e.) index for W, in the usual sense of recursion theory, is called a recursive (r.e.) α-index of C.

When regarding computability on a cusl we are often not only interested in having a decidable ordering but also having a decidable consistency relation and the ability to compute suprema of finite consistent sets. Therefore we consider a cusl to be a structure of the form

$$P = (P; \sqsubseteq, \mathrm{Cons}, \sqcup, \perp).$$

Definition 3.6. Let P be a cusl. Then (P, α) is a *computable cusl* if α is a computable numbering of the structure $P = (P; \sqsubseteq, \mathrm{Cons}, \sqcup, \perp)$. A domain D is an *effective domain* if there is α such that (D_c, α) is a computable cusl. We denote this effective domain by (D, α).

It is clear from earlier remarks about the function space that if D and E are effective domains then so is $[D \to E]$ with a numbering obtained uniformly from the numberings of D and E.

We now extend computability from the cusl of computable elements to the whole domain.

Definition 3.7. Let (D, α) be an effective domain. Then $x \in D$ is α-*computable* if $\mathrm{approx}(x)$ is α-semidecidable. An r.e. index of the set $\alpha^{-1}[\mathrm{approx}(x)]$ is an α-*index* of the computable element x.

The prefix α will be dropped when the numbering is clear from the context. Let $D_k = \{x \in D : x \text{ is computable}\}$. Note that $D_c \subseteq D_k$.

Definition 3.8. Let (D, α) and (E, β) be effective domains. A continuous function $f: D \to E$ is (α, β)*-effective* if the relation $R \subseteq D_c \times E_c$ defined by

$$R(a, b) \iff b \sqsubseteq f(a)$$

is (α, β)-semidecidable, that is the relation

$$\hat{R}(m, n) \iff R\big(\alpha(m), \alpha(n)\big)$$

is r.e. An r.e. index for $\hat{R}$ is an *effective index for* f with respect to α and β.

Lemma 3.9. *Let (D, α), (E, β) and (F, γ) be effective domains and let $f: D \to E$ and $g: E \to F$ be continuous and (α, β)-effective and (β, γ)-effective respectively.*

(i) *If $x \in D$ is α-computable then $f(x) \in E$ is β-computable.*
(ii) *The composition $h = g \circ f$ is (α, γ)-effective.*

We observe that the standard proof, see [42], is uniform. That is, we can uniformly obtain an index for $f(x)$ from indices for f and x. Similarly an index for h is obtained uniformly from indices of f and g.

We can extend computability via a numbering from D_c to the computable elements D_k in the following way.

Theorem 3.10. *Let (D, α) be an effective domain. Then there is a numbering $\bar{\alpha}: \omega \to D_k$ such that*

(i) *the inclusion mapping $\iota: D_c \to D_k$ is $(\alpha, \bar{\alpha})$-computable and,*
(ii) *the relation $R(n, m) \iff \alpha(n) \sqsubseteq \bar{\alpha}(m)$ is r.e., i.e., $\mathrm{approx}(\bar{\alpha}(m))$ is α-semidecidable uniformly in m.*

It can be shown that the numbering $\bar{\alpha}$ is the unique one satisfying (i) and (ii), up to recursive equivalence, see [42]. This numbering is the "correct" one since it identifies effectiveness and computability for functions.

Theorem 3.11. *Let (D, α) and (E, β) be effective domains. Then a continuous function $f: D \to E$ is (α, β)-effective if, and only if, $f|_{D_k}: D_k \to E_k$ is $(\bar{\alpha}, \bar{\beta})$-computable.*

In fact, the theorem in its stronger form says that any $(\bar{\alpha}, \bar{\beta})$-computable function $\bar{f}: D_k \to E_k$ extends to a continuous (α, β)-effective $f: D \to E$.

4 Domain representability

Our overall aim is to model streams and stream transformers using domains in a uniform way covering both discrete and continuous time and discrete and continuous data. The method we choose is *domain representation* in which time T and data A are represented by domains D_T and D_A, and a stream space is represented by the domain $[D_T \to D_A]$. First we recall the basic notions of domain representability and then we describe standard domain representations of metric spaces. Finally, in order to model non-continuous streams, we generalise the notion of domain representation to approximate representations. This is related to domain representations of structures with relations.

4.1 Basic definitions

We briefly recall some basic definitions and facts about domain representability. For motivation and details we refer to [47].

Let X and Y be topological spaces. Recall that function $\nu: X \to Y$ is a *quotient mapping* if $U \subseteq Y$ is open if, and only if, $\nu^{-1}[U]$ is open in X. In case ν is surjective we then have that $X/\!\sim$ and Y are homeomorphic spaces when the former is given the quotient topology and where $\sim$ is the equivalence relation induced on X by ν, i.e., $x \sim y \iff \nu(x) = \nu(y)$. Here is the basic definition.

Definition 4.1. Let X be a topological space, let D be a domain and D^R a subset of D. Then (D, D^R, ν) is a *domain representation* of X if $\nu: D^R \to X$ is a surjective quotient map when D^R is given the (relativised) Scott topology. In case (D, α) is an effective domain then (D, D^R, ν, α) is an *effective domain representation* of X.

Thus a domain representation (D, D^R, ν) of X contains both concrete and proper approximations of elements of X, the compact elements in D_c, and "total" elements in D^R containing sufficient information to represent elements of X exactly via ν. Since the function ν in the definition above is a quotient map we have

$$D^R/\!\sim \;\cong\; X.$$

Definition 4.2. A domain representation (D, D^R, ν) of a space X is

(i) *upwards closed* if whenever $x \in D^R$ and $x \sqsubseteq y$ then $y \in D^R$ and $\nu(x) = \nu(y)$;
(ii) *dense* if for each $a \in D_c$, $\uparrow a \cap D^R \neq \emptyset$;
(iii) *local* if $(\forall x, y \in D^R)(\nu(x) = \nu(y) \implies x$ and y are consistent).

Upwards closed domain representations (D, D^R, ν) are natural when regarding the ordering $\sqsubseteq$ on D as an information ordering. If $x \in D^R$ completely determines $\nu(x) \in X$ and $x \sqsubseteq y$ then y contains all the information of x and hence also completely determines $\nu(x)$. All domain representations considered in this paper are upwards closed.

The usual way to construct a domain representation of a space X is to consider an *approximation structure* of concrete approximations of X. An approximation structure most often takes the form of a cusl $P = (P; \sqsubseteq, \bot)$. Then the representing domain is the ideal completion $\mathrm{Idl}(P)$ of P. A space X will have many domain representations. It is up to the "user" to choose a representation appropriate for his or her purposes. We refer to [47] for a discussion of approximation structures.

The next step is to represent continuous functions between topological spaces.

Definition 4.3. Let (D, D^R, ν) and (E, E^R, μ) be domain representations of X and Y respectively. A function $f: X \to Y$ is *represented* by (or *lifts* to) a continuous function $\bar{f}: D \to E$ if $\bar{f}(D^R) \subseteq E^R$ and $\mu(\bar{f}(x)) = f(\nu(x))$, for all $x \in D^R$.

This means that the following diagram commutes.

$$\begin{array}{ccc} D & \xrightarrow{\bar{f}} & E \\ \uparrow & & \uparrow \\ D^R & \xrightarrow{\bar{f}|_{D^R}} & E^R \\ \nu\downarrow & & \downarrow\mu \\ X & \xrightarrow{f} & Y \end{array}$$

Let (D, D^R, ν) and (E, E^R, μ) be domain representations of X and Y respectively. Suppose $\bar{f}: D \to E$ is such that $\bar{f}[D^R] \subseteq E^R$ and such that $\nu(x) = \nu(y) \implies \mu(\bar{f}(x)) = \mu(\bar{f}(y))$. Then $\bar{f}$ induces a unique function $f: X \to Y$ defined by $f(\nu(x)) = \mu(\bar{f}(x))$. In the terminology above, f is represented by $\bar{f}$.

Proposition 4.4. *If $f: X \to Y$ is represented by a continuous $\bar{f}: D \to E$ then f is continuous.*

The proof is simple and depends on the fact that ν is assumed to be a quotient (only continuity of μ is required).

Recall that D_k is the set of computable elements of the effective domain (D, α). Suppose the topological space X is represented by (D, D^R, ν, α). Then the set X_k of (D, D^R, ν, α)-*computable elements* of X is the set

$$X_k = \{x \in X : \nu^{-1}(x) \cap D_k \neq \emptyset\}.$$

Let $\bar{\alpha}$ be the canonical numbering of D_k obtained from α as in Theorem 3.10 and let $\Omega = \{n \in \omega : \nu(n) \in D^R\}$. Define $\tilde{\alpha}: \Omega \to X_k$ by

$$\tilde{\alpha}(n) = \nu(\bar{\alpha}(n)).$$

The numbering $\tilde{\alpha}$ is the *canonical numbering of X_k obtained from the domain representation* (D, D^R, ν, α).

Now suppose (E, E^R, μ, β) is a domain representation of Y and suppose $f: X \to Y$ has an (α, β)-*effective* representation $\bar{f}: D \to E$. Then we say that f is (α, β)-*effective*. It follows from the results for effective domains that $f[X_k] \subseteq Y_k$ and $f|_{X_k}: X_k \to Y_k$ is $(\tilde{\alpha}, \tilde{\beta})$-computable. Sufficient conditions for when an $(\tilde{\alpha}, \tilde{\beta})$-computable function $g: X_k \to Y_k$ can be extended to an $(\tilde{\alpha}, \tilde{\beta})$-effective function $f: X \to Y$ is established in [2, Theorem 2.27]. These conditions are met by the real numbers $\mathbb{R}$, as well as by many other recursive metric spaces.

We see that an effective domain representation of a topological space X induces effectivity on X. Thus the effectivity of X in the sense described here is dependent on the effective domain representation chosen. It is shown in [48] that other notions of effectivity considered in the literature for topological spaces and algebras, such as the algebra of real numbers $\mathbb{R}$, are obtainable from effective domain representations, showing that the method of domain representation is not only flexible and general but also has sufficient strength.

For a discrete space X we have a domain representation $(X_\perp, X, \mathrm{id})$. Here is another representation providing more information in the sense of having a richer set of approximations.

Example 4.5. Let X be a discrete topological space and let $E = \{X\} \cup \mathcal{P}_f(X) \setminus \{\emptyset\}$ be the domain of finite subsets of X ordered by reverse inclusion. Let E^R consist of all singleton sets in E. Then (E, E^R, ν) is a domain representation of X where $\nu(\{x\}) = x$, in fact,

$$E^R \cong X.$$

We will denote the domain E by $\mathcal{P}_f(X)$.

4.2 Standard representations of metric spaces

We will briefly describe how to construct a domain representation of a metric space X. The domain representation constructed here will be referred to as a *standard domain representation.*

Definition 4.6. Let X be a metric space and let P be a family of non-empty closed subsets of X, containing X but not $\emptyset$. Then P is a *closed neighbourhood system* for X if the following are satisfied:

(i) if $F, F' \in P$ and $F \cap F' \neq \emptyset$ then $F \cap F' \in P$, and
(ii) if $x \in U$, where U is open, then $(\exists F \in P)(x \in F^\circ \ \wedge \ F \subseteq U)$.

Here F° denotes the interior of F.

A closed neighbourhood system P is a cusl when ordered by reverse inclusion. Clearly each metric space has a closed neighbourhood system since metric spaces are regular.

Let X be a metric space with metric d and choose a closed neighbourhood system P for X. Let D be the ideal completion of the cusl $(P; \sqsubseteq, X)$, where $\sqsubseteq$ is reverse inclusion. For $F \in P$ we let

$$\mathrm{diam}(F) = \sup_{x,y \in F} d(x, y).$$

Definition 4.7. An ideal $I \in D$ is *converging* if

$$(\forall \varepsilon > 0)(\exists F \in I)(\mathrm{diam}(F) < \varepsilon).$$

It is easy to see that the intersection of a converging ideal is a singleton set. Let D^R be the set of converging ideals and define $\nu: D^R \to X$ by

$$\nu(I) = x \iff \bigcap I = \{x\}.$$

Theorem 4.8. *Let X be a metric space. Then (D, D^R, ν) constructed as above is a domain representation of X, which is upwards closed, dense and local.*

We say that (D, D^R, ν) is a *standard domain representation* of X.

Theorem 4.9. *Let (D, D^R, ν) and (E, E^R, μ) be standard domain representations of metric spaces X and Y respectively. Then every continuous function $f: X \to Y$ can be represented by a continuous function $\bar{f}: D \to E$.*

The proof of the theorems above can be found in [2].

For the real numbers $\mathbb{R}$ we choose in this paper the closed neighbourhood system

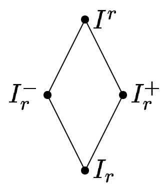
$$P = \{[a,b] : a \leq b \text{ and } a,b \in \mathbb{Q}\} \cup \{\mathbb{R}\}$$

and denote its ideal completion Idl(P) by $\mathcal{R}$. Clearly, P is a computable cusl and hence $\mathcal{R}$ is an effective domain. Thus $\mathcal{R} = (\mathcal{R}, \mathcal{R}^R, \nu)$ is an effective domain representation of $\mathbb{R}$. It is shown in [47] that the computability on $\mathbb{R}$ induced by $\mathcal{R}$ coincides with the notions of computability usually considered in recursive analysis (e.g. in [37]).

It is easy to see that for an irrational point $x \in \mathbb{R}$ there exists only one converging ideal, namely the ideal $I_x = \{[a,b] : a < x < b\}$. However, for a rational point $r \in \mathbb{R}$ there exist four different converging ideals representing r. The existence of several ideals representing the same point in $\mathbb{R}$ is necessary for topological reasons. The converging ideals representing a rational point r are:

$$\begin{aligned} I_r &= \{[a,b] : a < r < b\}, \\ I_r^+ &= \{[a,b] : a \leq r < b\}, \\ I_r^- &= \{[a,b] : a < r \leq b\}, and \\ I^r &= \{[a,b] : a \leq r \leq b\}. \end{aligned}$$

These are ordered as indicated in the following diagram.



Effective representations exist for a large class of metric spaces. However, often a more intricate construction than the one above is needed; see [2].

A discrete topological space X can be given a discrete metric. Then $(X_\perp, X,$ id) is (isomorphic to) a standard representation of X in the sense above. Similarly, the representation in Example 4.5 is a standard representation of X.

4.3 Representing relations and non-continuous functions

Streams need not be continuous. Thus, in order to represent such a stream space by a function space domain, we need a way to represent a non-continuous function by a continuous function between domains. By Proposition 4.4 we know that this is impossible. We have to settle for representing non-continuous functions *approximately.*

An analogous problem is how to represent a relation on a space. A canonical example is the space of real numbers $\mathbb{R}$. How do we represent the $\leq$ relation? The problem is that relations in terms of their characteristic functions are often not continuous.

Let X be a topological space and let $\mathbb{B} = \{\text{true}, \text{false}\}$ be the discrete boolean space. An n-ary relation P on X can be identified with its *characteristic function* $c_P\colon X \to \mathbb{B}$ defined by

$$c_P(a_1, \ldots, a_n) = \begin{cases} \text{true, if } P(a_1, \ldots, a_n); \\ \text{false, if } \neg P(a_1, \ldots, a_n). \end{cases}$$

The idea is to represent the possibly non-continuous characteristic function continuously in such a way that it gives exact values at points of continuity and only proper approximations at points of discontinuity. We know that this is the best possible.

Let (D, D^R, ν) be a domain representation of X and let P be an n-ary relation on X. Define $\bar{c}_P : D_c^n \to \mathbb{B}_\perp$ by

$$\bar{c}_P(\boldsymbol{a}) = \begin{cases} \text{true, if } (\forall \boldsymbol{x} \in (D^R)^n)(\boldsymbol{a} \sqsubseteq \boldsymbol{x} \implies P(\nu(\boldsymbol{x}))); \\ \text{false, if } (\forall \boldsymbol{x} \in (D^R)^n)(\boldsymbol{a} \sqsubseteq \boldsymbol{x} \implies \neg P(\nu(\boldsymbol{x}))); \\ \perp, \quad \text{otherwise.} \end{cases}$$

$\bar{c}_P$ is clearly monotone and hence extends uniquely to a continuous function

$$\bar{c}_P\colon D^n \to \mathbb{B}_\perp .$$

We say that $\bar{c}_P$ *represents* c_P or P *approximately.*

Example 4.10. Consider the standard interval representation $\mathcal{R}$ of the reals and the relation $\leq$. Then

$$\bar{c}_{\leq}([a, b], [c, d]) = \begin{cases} \text{true, if } b \leq c; \\ \text{false, if } d < a; \\ \perp, \quad \text{otherwise.} \end{cases}$$

Note that $\bar{c}_{\leq}$ is effective. If $x < y$ in $\mathbb{R}$ then $\bar{c}_{\leq}(I_x, I_y) = \text{true}$ and if $y < x$ in $\mathbb{R}$ then $\bar{c}_{\leq}(I_x, I_y) = \text{false}$. In case $x = y$ then $\bar{c}_{\leq}(I_x, I_x) = \perp$.

The function $c_{\leq}\colon \mathbb{R}^2 \to \mathbb{B}$ is continuous on

$$\{(x, y) : x \neq y\} \subseteq \mathbb{R}^2 ,$$

and discontinuous on the diagonal. Thus $\bar{c}_{\leq}$ represents $c_{\leq}$ exactly on points of continuity. At points of discontinuity $\bar{c}_{\leq}$ only provides the trivial approximation of the value of $c_{\leq}$.

It is well-known that $\leq$ is not decidable or even semidecidable on the recursive reals $\mathbb{R}_k$, the problem being that equality on $\mathbb{R}_k$ is not semidecidable. (Equality is cosemidecidable, i.e., $\neq$ is semidecidable.) This is reflected by the discontinuity of $c_{\leq}$.

We now want to generalise the continuous representation of relations to continuous representations of non-continuous functions. The idea is the same. We want our representation to be exact at points of continuity and as good as possible in terms of approximations at points of discontinuity. Here is the definition.

Definition 4.11. Let (D, D^R, ν_X) and (E, E^R, ν_Y) be domain representations of the topological spaces X and Y, respectively. Then a function $f: X \to Y$ (not necessarily continuous) is said to be *represented approximately* by (or *lifts approximately* to) $\bar{f}: D \to E$ if

(i) $\bar{f}$ is continuous,
(ii) $(\forall x \in D^R)(f \text{ continuous at } \nu_X(x) \implies \bar{f}(x) \in E^R \text{ and } f\nu_X(x) = \nu_Y \bar{f}(x))$, and
(iii) $(\forall x \in D^R)(f \text{ not continuous at } \nu_X(x) \implies (\exists y \in \nu_Y^{-1}[f\nu_X(x)])(\bar{f}(x) \sqsubseteq y))$.

The following example illustrates the notion above.

Example 4.12. The floor function $\lfloor\cdot\rfloor: \mathbb{R} \to \mathbb{Z}$ is discontinuous at precisely the integer points. We shall represent the floor function by a continuous domain function in the above precise sense.

Let $\mathcal{R}$ be the standard interval domain representation of $\mathbb{R}$. We could choose $\mathbb{Z}_\perp$ as a representation of $\mathbb{Z}$ and proceed as in the case of representing relations. However we can do much better if we choose a domain representation with more care. Let $(\mathcal{P}_f(\mathbb{Z}), \mathcal{P}_f(\mathbb{Z})^R, \mu)$ be the domain representation of $\mathbb{Z}$ described in Example 4.5. Define $f: \mathcal{R}_c \to \mathcal{P}_f(\mathbb{Z})$ by

$$f([a,b]) = \{m \in \mathbb{Z} : \lfloor a\rfloor \leq m \leq \lfloor b\rfloor\}$$

and $f(\mathbb{R}) = \mathbb{Z}$ (i.e., f is strict). Clearly, f is monotone and hence extends uniquely to a continuous function $f: \mathcal{R} \to \mathcal{P}_f(\mathbb{Z})$.

Let $x \in \mathbb{R}$ and let $I_x \in \mathcal{R}^R$ be the smallest ideal representing x. If x is not an integer then $f(I_x) = \{\lfloor x\rfloor\}$. Thus f represents the floor function exactly for all points of continuity. Now consider an integer m. Recall the four different representations of m described in Section 4.2. It is easily seen that

$$\begin{aligned} f(I_m) &= \{m-1, m\}, \\ f(I_m^-) &= \{m-1, m\}, \\ f(I_m^+) &= \{m\}, \text{ and} \\ f(I^m) &= \{m\}. \end{aligned}$$

It follows that f represents the floor function approximately in the sense of Definition 4.11. However, thanks to our choice of representation for $\mathbb{Z}$ we are able to obtain much information also at points of discontinuity. This illustrates the importance of choosing appropriate representations of the data types. Had we chosen $\mathbb{Z}_\perp$ to represent $\mathbb{Z}$ then the representation of the floor function would provide no information at points of discontinuity.

We close this section by showing that under rather general conditions, satisfied by the representations considered in this paper, there is a best continuous approximate representation of an arbitrary function, if there is one at all.

Theorem 4.13. *Let (D, D^R, ν_X) and (E, E^R, ν_Y) be domain representations of X and Y, respectively. Assume that D^R is dense in D, and that (E, E^R, ν_Y) is upwards closed and local. Let $f: X \to Y$ be a function and assume that f has one approximate representation in $[D \to E]$. Then there is a best approximate representation $\bar{f} \in [D \to E]$ in the sense of the domain ordering.*

It should be remarked that in the cases of streams that we consider we always have an approximate representation (see Theorem 5.2) and hence a best approximate representation. However, this representation is best only in the sense of the domain ordering. It may not be the best in the sense of computability, i.e. there may be a computable representation even though the best representation is not computable. Of course, the latter only affects the values at points of discontinuity.

Proof. Let $\mathcal{A}_f = \{\bar{f} \in [D \to E]: \bar{f}$ represents f approximately$\}$. We show that $\mathcal{A}_f$ is directed. $\mathcal{A}_f \neq \emptyset$ by assumption. Suppose $\bar{f}_1, \bar{f}_2 \in \mathcal{A}_f$. We first show that $\bar{f}_1$ and $\bar{f}_2$ are consistent. For this it suffices by the density of D^R to show that $\bar{f}_1(x)$ and $\bar{f}_2(x)$ are consistent for each $x \in D^R$.

Fix $x \in D^R$ and assume f is continuous at $\nu_X(x)$. Thus $\bar{f}_1(x), \bar{f}_2(x) \in E^R$ and

$$\nu_Y(\bar{f}_1(x)) = \nu_Y(\bar{f}_2(x)) = f(\nu_X(x)).$$

But by the hypotheses on (E, E^R, ν_Y) the supremum $\bar{f}_1(x) \sqcup \bar{f}_2(x)$ exists and

$$\nu_Y(\bar{f}_1(x) \sqcup \bar{f}_2(x)) = f(\nu_X(x)).$$

Now assume f is discontinuous at $\nu_X(x)$. Then there are $y_1, y_2 \in E^R$ such that $\nu_Y(y_1) = \nu_Y(y_2) = f(\nu_X(x))$ and $\bar{f}_1(x) \sqsubseteq y_1$ and $\bar{f}_2(x) \sqsubseteq y_2$. Again, by the assumptions on (E, E^R, ν_Y),

$$\bar{f}_1(x) \sqcup \bar{f}_2(x) \sqsubseteq y_1 \sqcup y_2 \in E^R$$

and $\nu(y_1 \sqcup y_2) = f(\nu_X(x))$. We conclude that $\bar{f}_1$ and $\bar{f}_2$ are consistent and $\bar{f}_1 \sqcup \bar{f}_2 \in \mathcal{A}_f$, that is, $\mathcal{A}_f$ is directed.

Let $\bar{f} = \bigsqcup \mathcal{A}_f$. We need to show that $\bar{f} \in \mathcal{A}_f$. Let $x \in D^R$ be such that f is continuous at $\nu_X(x)$ and let $\bar{g} \in \mathcal{A}_f$. Then $\bar{g}(x) \sqsubseteq \bar{f}(x)$ and $\bar{g}(x) \in E^R$ so $\bar{f}(x) \in E^R$ and

$$\nu_Y(\bar{f}(x)) = \nu_Y(\bar{g}(x)) = f(\nu_X(x)).$$

Now suppose f is discontinuous at $\nu_X(x)$. Let $\bar{y} \in E^R$ be the maximal element such that $\nu_Y(\bar{y}) = f(\nu_X(x))$. The assumptions on (E, E^R, ν_Y) imply that such $\bar{y}$ exists. Thus $\bar{g}(x) \sqsubseteq \bar{y}$ and hence

$$\bar{f}(x) = \bigsqcup\{\bar{g}(x) : \bar{g} \in \mathcal{A}_f\} \sqsubseteq \bar{y},$$

proving that $\bar{f} \in \mathcal{A}_f$. □

5 Modelling streams

In this section we give a semantic model for a set of streams $T \to A$ using the function space of the representing domains.

Let (D_T, D_T^R, ν_T) and (D_A, D_A^R, ν_A) be domain representations of T and A respectively. Then, as described in Section 4.1, D_T and D_A capture the original topologies of T and A, or, alternatively, induce topologies on T and A. The domain $[D_T \to D_A]$ contains only continuous functions. Thus, by Proposition 4.4, any function $f: T \to A$ (totally) represented by $\bar{f} \in [D_T \to D_A]$ as in Definition 4.3 is continuous.

When time T is discrete it is natural to model T by $\mathbb{N}$ or $\mathbb{Z}$ (depending on the existence of an initial time) with the discrete topology. In this case every function from T to A is continuous, i.e., all streams from T to A are continuous.

Let us now consider continuous time T. In this case we model T by $\mathbb{R}_+$ or $\mathbb{R}$ with their usual topologies. Assume that A is a discrete set of data, i.e., A has the discrete topology. Then the continuous streams from T to A are precisely the constant streams since T is a connected space. Thus all interesting streams from continuous time to discrete data are non-continuous. One way to deal with such streams, necessary from a view of computability, is apparent via the use of domains. A stream f in the complete stream space $(T \to A)$ is represented by a continuous function $\bar{f}: D_T \to D_A$ which gives correct values at (the representation of) points of continuity of f and approximate values at points of discontinuity (see Definition 4.11).

Here is the formal definition.

Definition 5.1. Let time T and data A have domain representations (D_T, D_T^R, ν_T) and (D_A, D_A^R, ν_A) respectively and let $(T \to A)$ be a stream space. Then $[D_T \to D_A]$ is an *approximate domain representation* of $(T \to A)$ if each stream $\varphi \in (T \to A)$ has an approximate representation $\bar{\varphi}$ in $[D_T \to D_A]$.

To simplify the presentation we make the assumption that discrete time is modelled by $\mathbb{N}$ and continuous time is modelled by $\mathbb{R}$. The reader can easily modify all arguments for the mentioned variants of models for time.

As domain representation for time T we choose $(\mathbb{N}_\perp, \mathbb{N}, \mathrm{id})$ in the discrete case and the standard interval domain representation $\mathcal{R}$ of $\mathbb{R}$ for the continuous case. We denote either representation by D_T.

Theorem 5.2. *Let T be time, discrete or continuous, and let A be a data type. Assume that A is a metric space and let (D_A, D_A^R, ν_A) be a standard domain representation of A. Then each stream*

$$\varphi: T \to A$$

has an approximate representation $\bar{\varphi}$ in $[D_T \to D_A]$.

Thus the complete stream space from T to A has an approximate domain representation using the function space obtained from standard domain representations of time and data. Of course, a discrete space is a metric space when given a discrete metric.

Recall that $\mathcal{R}_c$, the cusl of compact elements of $\mathcal{R}$, consists of all closed intervals with rational endpoints and $\mathbb{R}$, ordered by reverse inclusion.

Proof. If T is discrete then each stream $\varphi: T \to A$ is continuous. By Theorem 4.9, φ lifts to a continuous function $\bar{\varphi}: D_T \to D_A$.

Now we assume that T is continuous time and is modelled by $\mathbb{R}$ with the standard interval domain representation $\mathcal{R}$.

Let $\varphi: T \to A$ be a stream. We define $\bar{\varphi}: \mathcal{R}_c \to D_A$ by

$$\bar{\varphi}([a,b]) = \{F \in (D_A)_c : \varphi([a,b]) \subseteq F^\circ\},$$

where F° denotes the interior of F. Then $\bar{\varphi}$ is monotone and extends uniquely to a continuous function $\bar{\varphi}: \mathcal{R} \to D_A$. In fact,

$$\bar{\varphi}(I) = \{F \in (D_A)_c : (\exists [a,b] \in I)(\varphi([a,b]) \subseteq F^\circ)\}.$$

Suppose φ is continuous at the point $x \in T$ and consider the ideal $I_x = \{[a,b] : a < x < b\} \in \mathcal{R}^R$. Recall that I_x is the smallest ideal representing x. Let $J = \bar{\varphi}(I_x)$. We need to show that $J \in D_A^R$ and that $\nu_A(J) = \varphi(x)$. Put differently, we need to show that $\bigcap J = \{\varphi(x)\}$. Obviously $\varphi(x) \in \bigcap J$. Given $\varepsilon > 0$, take $F \in (D_A)_c$ such that

$$\varphi(x) \in F^\circ \subseteq F \subseteq B(\varphi(x), \varepsilon),$$

where the latter is the open ε-ball around $\varphi(x)$. Since φ is continuous at x and F° is open there are $a', b' \in \mathbb{Q}$, $a' < x < b'$ and $\varphi[(a', b')] \subseteq F^\circ$. But then there is $[a,b] \in I_x$ such that

$$\varphi[\,[a,b]\,] \subseteq \varphi[(a',b')] \subseteq F^\circ,$$

i.e., $F \in J$. But $\operatorname{diam}(F) < \varepsilon$ and ε was arbitrary. It follows that $\bigcap J = \{\varphi(x)\}$.

Now suppose $x \in T$ is a point of discontinuity of φ. It suffices to show

$$I \supseteq I_x \implies \bar{\varphi}(I) \sqsubseteq I_{\varphi(x)}.$$

But this follows directly from the definition of $\bar{\varphi}$. □

The theorem shows that the domain of continuous functions $[D_T \to D_A]$ contains representations of all streams from T to A. The representations are, however, only approximate on points of discontinuity. From a computational point of view this is quite reasonable. We cannot compute exactly on continuous data types, including continuous time, we can only compute on approximations of data. At points of continuity we obtain approximations of arbitrary precision. At points of discontinuity we can only expect proper approximations. However, with an appropriate choice of domain representation this may nonetheless produce important information.

Corollary 5.3. *Each stream $\varphi: T \to A$ has a best representation $\bar{\varphi}$ in $[D_T \to D_A]$.*

Proof. The hypotheses of Theorem 4.13 are satisfied for standard representations. □

¿From an approximate domain representation of the complete stream space $(T \to A)$ we define an equivalence relation $\sim$ on $(T \to A)$ by saying that $\varphi \sim \psi$ if they have the same best approximation. Thus we obtain a domain representation in the sense of Definition 4.1 of $(T \to A)/\!\sim$.

Finally we consider computability of streams. Note that by Ceitin's theorem [7] each computable stream $\varphi: T \to A$, where A is a computable metric space in the sense of [2], is continuous. Thus in order to consider "computability" of non-continuous streams it is necessary to consider approximate representations.

Definition 5.4. Let $(D_T, D_T^R, \nu_T, \alpha)$ and $(D_A, D_A^R, \nu_A, \beta)$ be effective domain representations of time T and data A, respectively. Then a stream $\varphi: T \to A$ is *computable* if there is an (α, β)-effective $\bar{\varphi} \in [D_T \to D_A]$ representing φ approximately.

As an example we consider non-zeno signals into a discrete space A. (Recall Definition 2.14.)

We choose the domain representation $\mathcal{P}_f(A)$ for A. This is clearly an effective domain when A is countable, and the set of maximal elements is decidable.

Let $\mathcal{R}_+$ be the standard interval domain representation of $\mathbb{R}_+$ obtained as the ideal completion of the cusl

$$\{[a, b] : 0 \leq a \leq b \text{ and } a, b \in \mathbb{Q}\} \cup \{\mathbb{R}_+\}$$

ordered by reverse inclusion.

Proposition 5.5. *If φ is a non-zeno signal represented by $\bar{\varphi} \in [\mathcal{R}_+ \to \mathcal{P}_f(A)]$ then*

$$Z \sqsubseteq \bar{\varphi}([a, b]) \implies \varphi[[a, b]] \subseteq Z.$$

Proof. Assume $Z \sqsubseteq \bar{\varphi}([a, b])$ and let $x \in [a, b]$. Note that $[a, b] \in I^x$, the largest ideal representing x. If φ is continuous at x then $\bar{\varphi}(I^x) = \{\varphi(x)\}$. But $[a, b] \in I^x$ so $Z \supseteq \bar{\varphi}([a, b]) \supseteq \{\varphi(x)\}$. If φ is not continuous at x then, by definition, we have $\bar{\varphi}(I^x) \sqsubseteq \{\varphi(x)\}$, i.e., again $\varphi(x) \in Z$. □

Proposition 5.6. *Suppose φ is a computable non-zeno signal represented by the effective $\bar{\varphi} \in [\mathcal{R}_+ \to \mathcal{P}_f(A)]$. Then each discontinuity of φ is a recursive real.*

Proof. Suppose $x \in \mathbb{R}_+$ is a point of discontinuity. Let $a < x < b$ be rational so that φ is constant on (a, x) and on (x, b). Let

$$\begin{aligned} I = \{[c, d] : (\exists c', d')(\exists m, n \in A)(a < c < c' < d' < d < b, \\ \bar{\varphi}([c, c']) = \{m\}, \bar{\varphi}([d', d]) = \{n\} \text{ and } m \neq n)\}. \end{aligned}$$

Note that I is semidecidable. If $[c, d] \in I$ then $\varphi[(a, x)] = \{m\}$ and $\varphi[(x, b)] = \{n\}$ so $x \in (c, d)$. Let y be such that $a < y < x$. Then, by the continuity of φ at y, there is $[c, c'] \in I_y$ such that $\bar{\varphi}([c, c']) = \{m\}$. Similarly, we obtain appropriate $[d', d]$. That is, I generates the ideal I_x. We have shown that I_x is a computable element of D and hence that x is a recursive real. □

Finally we note that for a computable non-zeno signal φ, the set of recursive reals at which φ is continuous is semidecidable (recall the exact definition in Section 3.2). For φ is continuous at x if, and only if,

$$(\exists [a,b] \in I_x)(\bar{\varphi}([a,b]) \in \mathcal{P}_f(A)^R).$$

6 Modelling stream transformers

Now we consider stream transformers $F: (R \to B) \to (T \to A)$. We have already seen that some stream spaces naturally include non-continuous streams. It is equally clear that there are natural stream transformers which take continuous streams to non-continuous streams. For example, let $A = B$ be a non-trivial discrete data type, R discrete time and T continuous time. Let $\tau: T \to R$ and define $F: (R \to B) \to (T \to A)$ by

$$F(\varphi)(t) = \varphi(\tau(t)).$$

Then every stream in $(R \to B)$ is continuous and hence a non-trivial F takes some continuous streams to non-continuous streams, that is, natural stream transformers do not necessarily preserve continuity.

On the other hand there is an intuitive feeling that a stream transformer that we would want to model should be continuous in the sense that approximations to the value of an output stream at a specific time should only depend on a "finite" part or approximation of the input stream. In order to make this precise we need to model approximations of streams, and for this our method of domain representability is natural.

6.1 Transformations of continuous streams

In this subsection we consider the following question. When does a continuous stream transformer

$$F: \mathrm{C}(R \to B) \to \mathrm{C}(T \to A)$$

taking continuous streams to continuous streams have a continuous lifting

$$\bar{F}: [D_R \to D_B] \to [D_T \to D_A]?$$

More precisely, let $\mathrm{C}(R \to B)$ and $\mathrm{C}(T \to A)$ denote the spaces of *continuous* streams, where B and A are metric spaces and where R and T is any pair of our usual models of time. The spaces of continuous streams $\mathrm{C}(R \to B)$ and $\mathrm{C}(T \to A)$ are given the *compact-open topology*. On $\mathrm{C}(T \to A)$ this is the topology generated by the subbasic open sets

$$\mathrm{W}(K, U) = \{f \in (T \to A) \mid f[K] \subseteq U\},$$

where $K \subseteq T$ is compact and $U \subseteq A$ is open. By a continuous stream transformer $F: \mathrm{C}(R \to B) \to \mathrm{C}(T \to A)$ we mean that F is continuous with respect to the compact-open topologies.

Let (D_A, D_A^R, ν_A) and (D_B, D_B^R, ν_B) be standard domain representations of A and B respectively. For simplicity in the exposition we choose $\mathbb{N}$ to model discrete time and $\mathbb{R}$ to model continuous time. The domain representations we consider for time are $(\mathbb{N}_\perp, \mathbb{N}, \mathrm{id})$ and $(\mathcal{R}, \mathcal{R}^R, \mu)$, respectively, where $\mathcal{R}$ is the standard interval domain representing $\mathbb{R}$.

Given time T and data set A, with the chosen domain representations as above, let

$$[D_T \to D_A]^R = \{f \in [D_T \to D_A] : f[D_T^R] \subseteq D_A^R\}.$$

We know by Proposition 4.4 that each $f \in [D_T \to D_A]^R$ induces a unique continuous function $\tilde{f}: T \to A$. Denoting $\tilde{f}$ by $\nu(f)$ we obtain the following theorem.

Theorem 6.1. *$([D_T \to D_A], [D_T \to D_A]^R, \nu)$ is an upwards closed domain representation of $\mathrm{C}(T \to A)$, when the latter is given the compact-open topology.*

The theorem follows from the fact that T is locally compact and that standard domain representations of T and A are used. The proof of the general fact appears in Blanck [3]. The special case of $T = A = \mathbb{R}$ appears in di Gianantonio [14].

In order to answer the question initially posed in this section, we first prove the following general lifting theorem.

Theorem 6.2. *Let X be a topological space with a dense domain representation (E, E^R, ν). Let T and A be time and data as above with domain representations D_T and D_A. Then each continuous*

$$\varphi: X \to \mathrm{C}(T \to A)$$

has a continuous lifting

$$\bar{\varphi}: E \to [D_T \to D_A].$$

Proof. We consider the case $T = \mathbb{R}$ and $D_T = \mathcal{R}$. The proof for $T = \mathbb{N}$ is similar but simpler.

For $a \in E_c$ let $B_a = \{x \in E : a \sqsubseteq x\}$, the basic open set determined by a. Define $\bar{\varphi}: E_c \to [\mathcal{R} \to D_A]$ by

$$\bar{\varphi}(a) = \bigsqcup\{\textstyle\bigsqcup_{i=1}^n \langle [c_i, d_i]; G_i\rangle : n \geq 1 \text{ and} \\ (\forall i = 1, \ldots, n)(\varphi\nu[B_a \cap E^R] \subseteq \mathrm{W}([c_i, d_i], G_i^\circ))\}.$$

To see that $\bar{\varphi}(a)$ is well-defined consider $\langle [c_i, d_i]; G_i\rangle$ for $i = 1$ and 2 where $\varphi\nu[B_a \cap E^R] \subseteq \mathrm{W}([c_i, d_i], G_i^\circ)$. We must show that $\langle [c_1, d_1]; G_1\rangle$ and $\langle [c_2, d_2]; G_2\rangle$ are consistent. Suppose $[c_1, d_1] \cap [c_2, d_2] \neq \emptyset$ with $y \in \mathbb{R}$ as a witness. Let $x \in B_a \cap E^R$, which exists by the density of E^R. Then $\varphi\nu(x)(y) \in G_1 \cap G_2$ so $\langle [c_1, d_1]; G_1\rangle$ and $\langle [c_2, d_2]; G_2\rangle$ are consistent. Generalising the argument to arbitrary elements in $[\mathcal{R} \to D_A]_c$ proves that φ is well-defined.

It is clear from its definition that φ is monotone and hence extends uniquely to a continuous function $\bar{\varphi}: E \to [\mathcal{R} \to D_A]$. We now show that $\bar{\varphi}$ represents φ. Let $\bar{x} \in E^R$ and suppose $\nu(\bar{x}) = x \in X$. We need to show that $v(\bar{\varphi}(\bar{x})) = \varphi(x)$

where $([\mathcal{R} \to D_A], [\mathcal{R} \to D_A]^R, \upsilon)$ is the domain representation of $\mathrm{C}(\mathbb{R} \to A)$. For this it suffices to show, for $y \in \mathbb{R}$,

$$I_{\varphi(x)(y)} \sqsubseteq \bar{\varphi}(\bar{x})(I_y).$$

Let $G \in I_{\varphi(x)(y)}$, that is, $\varphi(x)(y) \in G^\circ$. By the continuity of $\varphi(x)$ there is $[c,d] \in I_y$ such that $\varphi(x) \in \mathrm{W}([c,d], G^\circ)$. By the continuity of φ the set $\nu^{-1}\varphi^{-1}[\mathrm{W}([c,d], G^\circ)]$ is open. Choose $a \in E_c$ such that

$$\bar{x} \in B_a \cap E^R \subseteq \nu^{-1}\varphi^{-1}[\mathrm{W}([c,d], G^\circ)].$$

Thus $a \sqsubseteq \bar{x}$ and

$$\varphi(x) = \varphi\nu(\bar{x}) \in \varphi\nu[B_a \cap E^R] \subseteq \mathrm{W}([c,d], G^\circ).$$

But then

$$\langle [c,d]; G\rangle \sqsubseteq \bar{\varphi}(a) \sqsubseteq \bar{\varphi}(\bar{x})$$

and hence, since $[c,d] \in I_y$, $G \sqsubseteq \bar{\varphi}(\bar{x})(I_y)$. This shows that $I_{\varphi(x)(y)} \sqsubseteq \bar{\varphi}(\bar{x})(I_y)$. □

In view of the above theorem it suffices, for the problem at hand, to consider sufficient conditions for density of the domain representation $[D_R \to D_B]$ of the space of continuous streams $\mathrm{C}(R \to B)$. We first consider the case when R is discrete, i.e., $R = \mathbb{N}$.

Lemma 6.3. $[\mathbb{N}_\perp \to D_B]$ *is a dense representation of* $\mathrm{C}(\mathbb{N} \to B)$.

Proof. Suppose $\bigsqcup_{i=1}^k \langle n_i; F_i\rangle \in [\mathbb{N}_\perp \to D_B]_c$. Without loss of generality, recalling the definition of a closed neighbourhood system, we can assume the n_i are distinct. Choose $x_0 \in B$ and $x_i \in F_i$. Then define $f\colon \mathbb{N}_\perp \to D_B$ by

$$f(n) = \begin{cases} I_{x_i} \sqcup [F_i], & \text{if } n = n_i; \\ I_{x_0}, & \text{if } n \neq n_i, \text{for } i = 1, \dots, k; \end{cases}$$

and $f(\perp) = \perp$. Clearly $\bigsqcup_{i=1}^k \langle n_i; F_i\rangle \sqsubseteq f$ and $f \in [\mathbb{N}_\perp \to D_B]^R$.

In case $\langle \perp; F\rangle$ appears in the compact element we alter the definition of f by choosing $x_i \in F \cap F_i$, $x_0 \in F$, and setting $f(\perp) = [F]$. □

An effective domain representation (D, D^R, ν, α) is *effectively dense* if there is a total recursive function d which given an α-index of $a \in D_c$ computes an $\bar{\alpha}$-index of some $x \in D^R$ such that $a \sqsubseteq x$. That is, for each $n \in \omega$,

$$\alpha(n) \sqsubseteq \bar{\alpha}d(n) \in D^R.$$

It follows from the proof of Lemma 6.3 that if D_B is an effective domain representation of B then the domain representation

$$([\mathbb{N}_\perp \to D_B], [\mathbb{N}_\perp \to D_B]^R, \nu)$$

of $\mathrm{C}(\mathbb{N} \to B)$ is effectively dense.

Corollary 6.4. *Let R be discrete time, let T be discrete or continuous time, and let A and B be metric spaces. Then each continuous stream transformer*

$$F\colon \mathrm{C}(R \to B) \to \mathrm{C}(T \to A)$$

lifts to a continuous

$$\bar{F}\colon [D_R \to D_B] \to [D_T \to D_A].$$

The case when R is continuous time, i.e. R is modelled by $\mathbb{R}$, is more delicate. The density of the domain representation $[\mathcal{R} \to D_B]$ depends on topological properties of the metric space B and the standard domain representation D_B of B. Note that if B is a discrete metric space, say $\mathbb{N}$ with standard domain representation $(\mathbb{N}_\perp, \mathbb{N}, \mathrm{id})$, then $[\mathcal{R} \to \mathbb{N}_\perp]$ is not a dense representation. For example,

$$\langle [0,1]; 1\rangle \sqcup \langle [2,3]; 2\rangle$$

has no representing element above it.

Below we let $\mathcal{R}$ be the standard interval domain representation of $\mathbb{R}$ and we let (E, E^R, μ) be a standard domain representation of a metric space X. For each $[a,b] \in \mathcal{R}_c$ and $F \in E_c$ we use the notation

$$\langle\!\langle [a,b]; F\rangle\!\rangle = \{f\colon \mathbb{R} \to X \mid f \text{ is continuous and } f([a,b]) \subseteq F\}.$$

Lemma 6.5. *Let $f \in \bigcap_{i=1}^{n} \langle\!\langle [a_i,b_i]; F_i\rangle\!\rangle$. Then $\{\langle [a_i,b_i]; F_i\rangle : i = 1,\ldots,n\}$ is consistent in $[\mathcal{R} \to E]$ and there is $\bar{f} \in [\mathcal{R} \to E]^R$ representing f such that*

$$\bigsqcup_{i=1}^{n} \langle [a_i,b_i]; F_i\rangle \sqsubseteq \bar{f}.$$

Proof. If $x \in \bigcap_{i\in I}[a_i,b_i]$ for $I \subseteq \{1,\ldots,n\}$ then $f(x) \in \bigcap_{i\in I} F_i$ proving the consistency statement.

Let $\tilde{f} \in [\mathcal{R} \to E]$ represent f. The function space representation is upwards closed so it suffices to show that $\tilde{f}$ and $\bigsqcup_{i=1}^{n} \langle [a_i,b_i]; F_i\rangle$ are consistent. For the latter it suffices to consider consistency with each compact approximation of $\tilde{f}$. So suppose $\bigsqcup_{i=1}^{m} \langle [c_i,d_i]; G_i\rangle \sqsubseteq \tilde{f}$. Let $I \subseteq \{1,\ldots,n\}$ and $J \subseteq \{1,\ldots,m\}$, and suppose

$$\Big(\bigcap_{i\in I}[a_i,b_i]\Big) \cap \Big(\bigcap_{j\in J}[c_j,d_j]\Big) \neq \emptyset$$

with x as a witness. Then $f(x) \in \bigcap_{i\in I} F_i$ by hypothesis. Consider the maximal ideal I^x representing x. Thus $\tilde{f}(I^x)$ represents $f(x)$. But for each $j \in J$, $x \in [c_j,d_j]$ and $\langle [c_j,d_j]; G_j\rangle \sqsubseteq \tilde{f}$ so $G_j \in \tilde{f}(I^x)$. But then $f(x) = \mu(\tilde{f}(I^x))$, that is, $f(x) \in G_j$. We have shown that $(\bigcap_{i\in I} F_i) \cap (\bigcap_{j\in J} G_j) \neq \emptyset$. □

Definition 6.6. A space X is *arcwise connected* if for every pair of distinct points $x_1, x_2 \in X$ there is a continuous embedding $h\colon [0,1] \to X$ such that $h(0) = x_1$ and $h(1) = x_2$.

Lemma 6.7. *Let X be a metric space with a standard representation (E, E^R, μ). Suppose X is arcwise connected and each $F \in E_c$ is arcwise connected. Then the function space representation $([\mathcal{R} \to E], [\mathcal{R} \to E]^R, \gamma)$ is dense.*

Note that the representation $\mathcal{R}$ of $\mathbb{R}$ satisfies the hypotheses. Similarly there is a standard representation of $\mathbb{R}^n$ satisfying the hypotheses.

Proof. It suffices to find $f \in \bigcap_{i=1}^n \langle\!\langle [a_i, b_i]; F_i \rangle\!\rangle$ for each $\bigsqcup_{i=1}^n \langle [a_i, b_i]; F_i \rangle \in E_c$, by Lemma 6.5. Fix such a compact element. Partition $\{1, \ldots, n\}$ into $I_1, \ldots, I_k$ such that $\bigcup_{i \in I_j} [a_i, b_i]$ is connected but

$$(\bigcup_{i \in I_j} [a_i, b_i]) \cap (\bigcup_{i \in I_l} [a_i, b_i]) = \emptyset$$

for $j \neq l$. Then $\bigcup_{i \in I_j} [a_i, b_i] = [\bar{a}_j, \bar{b}_j]$ and we may assume that $\bar{b}_j < \bar{a}_{j+1}$ for $j = 1, \ldots, k-1$.

Assume we have constructed $f_j \in \bigcap_{i \in I_j} \langle\!\langle [a_i, b_i]; F_i \rangle\!\rangle$ for $j = 1, \ldots, k$. Since the space X is arcwise connected there are continuous functions g_j for $j = 1, \ldots, k-1$ such that $g_j(\bar{b}_j) = f_j(\bar{b}_j)$ and $g_j(\bar{a}_{j+1}) = f_{j+1}(\bar{a}_{j+1})$. These functions can now be glued together to define

$$f(x) = \begin{cases} f_1(\bar{a}_1), \text{ if } x \leq \bar{a}_1; \\ f_j(x), \ \ \text{if } \bar{a}_j \leq x \leq \bar{b}_j, j = 1, \ldots, k; \\ g_j(x), \ \ \text{if } \bar{b}_j \leq x \leq \bar{a}_{j+1}, j = 1, \ldots, k-1; \\ f_k(\bar{b}_k), \text{ if } \bar{b}_k \leq x. \end{cases}$$

Clearly f is continuous and $f \in \bigcap_{i=1}^n \langle\!\langle [a_i, b_i]; F_i \rangle\!\rangle$.

Thus it suffices to consider the case when $\bigcup_{i=1}^n [a_i, b_i]$ is connected. Let $c_1 < c_2 < \cdots < c_k$ be a strictly increasing listing of the set $\{a_1, \ldots, a_n\} \cup \{b_1, \ldots, b_n\}$. For each j, choose $d_j \in \bigcap_{c_j \in [a_i, b_i]} F_i$ and let $I_j = \{i : (c_j, c_{j+1}) \subseteq [a_i, b_i]\}$. Then $I_j \neq \emptyset$ and $d_j, d_{j+1} \in \bigcap_{i \in I_j} F_i$. By assumption $\bigcap_{i \in I_j} F_i$ is arcwise connected, since E_c is closed under intersection, so there is a continuous function g_j such that $g_j(c_j) = d_j$, $g_j(c_{j+1}) = d_{j+1}$, and $g_j([c_j, c_{j+1}]) \subseteq \bigcap_{i \in I_j} F_i$. Now we define $f : \mathbb{R} \to X$ by

$$f(x) = \begin{cases} d_1, & \text{if } x \leq c_1; \\ g_j(x), & \text{if } c_j \leq x \leq c_{j+1}, j = 1, \ldots, k-1; \\ d_k, & \text{if } c_k \leq x. \end{cases}$$

Then f is continuous and $f \in \bigcap_{i=1}^n \langle\!\langle [a_i, b_i]; F_i \rangle\!\rangle$. □

It is worth noting that for $X = \mathbb{R}$ and $E = \mathcal{R}$, the representation

$$([\mathcal{R} \to \mathcal{R}], [\mathcal{R} \to \mathcal{R}]^R, \nu)$$

of $\mathrm{C}(\mathbb{R} \to \mathbb{R})$ is effectively dense.

Theorem 6.8. *Let R be continuous time, let T be discrete or continuous time, and let A and B be metric spaces with standard domain representations D_A and D_B. Assume B is arcwise connected and each $F \in (D_B)_c$ is arcwise connected. Then a stream transformer*

$$F: \mathrm{C}(R \to B) \to \mathrm{C}(T \to A)$$

is continuous if, and only if, F has a continuous lifting

$$\bar{F}: [D_R \to D_B] \to [D_T \to D_A].$$

Proof. By Proposition 4.4, Lemma 6.7 and Theorem 6.2. □

In particular we have

Corollary 6.9. *A functional $F: \mathrm{C}(\mathbb{R} \to \mathbb{R}) \to \mathrm{C}(\mathbb{R} \to \mathbb{R})$ is continuous with respect to the compact-open topology if, and only if, F has a continuous lifting $\bar{F}: [\mathcal{R} \to \mathcal{R}] \to [\mathcal{R} \to \mathcal{R}]$.*

D. Normann [35] has recently extended our results about density and lifting to the whole finite type structure over $\mathcal{R}$.

To summarise, we have shown that a continuous stream transformer

$$F: \mathrm{C}(R \to B) \to \mathrm{C}(T \to A)$$

has a continuous lifting

$$\bar{F}: [D_R \to D_B] \to [D_T \to D_A].$$

if R is discrete time or if R is continuous time and $B = \mathbb{R}$ or $\mathbb{R}^n$.

In the remaining case when R is continuous time and B is a discrete space then $\mathrm{C}(R \to B)$ is homeomorphic to B.

6.2 Simple transformations of non-continuous streams

The characterisation theorems in Section 6.1 provide domain representation methods for many examples of transformations acting on *continuous* streams $\mathrm{C}(R \to B)$. However, transformations of signals with discrete data, such as signals from $\mathbb{R}_+$ to $\mathbb{N}$, which are necessarily discontinuous when non-constant, are not covered though they are important in system modelling.

In Section 5 we saw how to provide domain representations of discontinuous streams (Theorem 5.2) and, in particular, the class of non-zeno signals. Now we will look at their transformations: we will show how to provide domain representations for stream transformations that are single or multiple access on non-zeno signals, see Section 2.2.

First we need to be precise about what we mean by a stream transformer on non-zeno signals having a domain representation.

Below we let $\mathcal{R}_+$ denote the standard closed interval domain representation of $\mathbb{R}_+$ and we let A and B denote discrete spaces with, for simplicity, flat domain representations. The stream space of non-zeno signals over A is denoted by

$$\mathrm{NZ}(\mathbb{R}_+ \to A).$$

For non-zeno signals it is reasonable to modify the notion of approximate representation as follows.

Definition 6.10. The function $\bar{\varphi} \in [\mathcal{R}_+ \to A_\perp]$ is a *non-zeno representation* of $\varphi \in \mathrm{NZ}(\mathbb{R}_+ \to A)$ if

(i) for each $t > 0$, the set $\{t' \in [0, t]: \bar{\varphi}(I_{t'}) = \perp\}$ is finite and does not contain 0, and
(ii) $\bar{\varphi}$ represents φ exactly for all t such that $\bar{\varphi}(I_t) \neq \perp$.

The set $\{t \in \mathbb{R}_+ : \bar{\varphi}(I_t) = \perp\}$ is called the *exceptional set* for the representation $\bar{\varphi}$ of φ.

Definition 6.11. Let $F: \mathrm{NZ}(\mathbb{R}_+ \to B) \to \mathrm{NZ}(\mathbb{R}_+ \to A)$ be a stream transformer on non-zeno signals. Then

$$\bar{F}: [\mathcal{R}_+ \to B_\perp] \to [\mathcal{R}_+ \to A_\perp]$$

is an *approximate non-zeno representation* or *lifting of* F if $\bar{F}$ is continuous and whenever $\bar{\varphi} \in [\mathcal{R}_+ \to B_\perp]$ is a non-zeno representation of $\varphi \in \mathrm{NZ}(\mathbb{R}_+ \to B)$ then $\bar{F}(\bar{\varphi})$ is a non-zeno representation of $F(\varphi)$.

Recall from Section 2.7 that single or multiple access signal operators with strict retimings take non-zeno signals to non-zeno signals.

Theorem 6.12. *Let $F: \mathrm{NZ}(\mathbb{R}_+ \to B) \to \mathrm{NZ}(\mathbb{R}_+ \to A)$ be a single access stream transformer with respect to $\pi: B \to A$ and a strict retiming $\tau: \mathbb{R}_+ \to \mathbb{R}_+$, where A and B are discrete spaces. Then F has an approximate non-zeno representation*

$$\bar{F}: [\mathcal{R}_+ \to B_\perp] \to [\mathcal{R}_+ \to A_\perp]$$

Proof. Define $\bar{F}: [\mathcal{R}_+ \to B_\perp]_c \to [\mathcal{R}_+ \to A_\perp]$ by

$$\begin{aligned} \bar{F}\Big(\bigsqcup_{i=1}^{k} \langle [a_i, b_i]; x_i \rangle\Big) = \bigsqcup \Big\{ \langle [c, d]; \pi(x) \rangle : (\exists i)(x = x_i \ \& \\ [0 < a_i \ \& \ [c, d] \subseteq (\tau^{-1}(a_i), \tau^{-1}(b_i)) \text{ or} \\ a_i = 0 < b_i \ \& \ [c, d] \subseteq [0, \tau^{-1}(b_i))]) \Big\}. \end{aligned}$$

It is routine to verify that $\bar{F}$ is well-defined and monotone and hence $\bar{F}$ extends continuously to

$$\bar{F}: [\mathcal{R}_+ \to B_\perp] \to [\mathcal{R}_+ \to A_\perp].$$

Let $\bar{\varphi} \in [\mathcal{R}_+ \to B_\perp]$ be a non-zeno representation of $\varphi \in \mathrm{NZ}(\mathbb{R}_+ \to B)$. Let $t_1 < t_2 < \ldots$ be the exceptional set for the representation $\bar{\varphi}$ of φ. We claim that $\bar{F}(\bar{\varphi})$ is a non-zeno representation of $F(\varphi)$ with exceptional set $\tau^{-1}(t_1) < \tau^{-1}(t_2) < \ldots$.

Suppose $\varphi(0) = x$ and choose δ such that $0 < \delta < t_1$. Then $\langle [0, \delta]; x \rangle \sqsubseteq \bar{\varphi}$ so there is d such that

$$\langle [0, d]; \pi(x) \rangle \sqsubseteq \bar{F}(\langle [0, \delta]; x \rangle) \sqsubseteq \bar{F}(\bar{\varphi}).$$

Thus

$$\pi(x) = \langle [0, d]; \pi(x) \rangle (I_0) \sqsubseteq \bar{F}(\bar{\varphi})(I_0),$$

that is, $\bar{F}(\bar{\varphi})(I_0) = F(\varphi)(0)$.

Similarly, $\bar{F}(\bar{\varphi})(I_t) = F(\varphi)(t)$ when $t \neq \tau^{-1}(t_i)$ for any i.

Now suppose $t = \tau^{-1}(t_i)$ for some i. Then $\bar{\varphi}(I_{t_i}) = \perp$. It then follows from the definition of $\bar{F}$ that $\bar{F}(\bar{\varphi})(I_t) = \perp$. □

Suppose (A, α) and (B, β) are computable structures. Then $(A_\perp, \tilde{\alpha})$ and $(B_\perp, \tilde{\beta})$ are effective domains with numberings obtained from α and β in a canonical way. It follows that $[\mathcal{R}_+ \to A_\perp]$ and $[\mathcal{R}_+ \to B_\perp]$ are effective domains with numberings obtained from α, β and a standard numbering of $\mathcal{R}_+$. Analysing the proof of Theorem 6.12, in particular the definition of $\bar{F}$, we obtain the following theorem (with numberings suppressed).

Theorem 6.13. *Let A and B be computable structures and let $F: \mathrm{NZ}(\mathbb{R}_+ \to B) \to \mathrm{NZ}(\mathbb{R}_+ \to A)$ be a single access stream transformer with respect to $\pi: B \to A$ and a strict retiming $\tau: \mathbb{R}_+ \to \mathbb{R}_+$. Assume that π is computable and τ is effective. Then F has an effective approximate non-zeno representation*

$$\bar{F}: [\mathcal{R}_+ \to B_\perp] \to [\mathcal{R}_+ \to A_\perp]$$

Remark 6.14. Theorems 6.12 and 6.13 easily extend to multiple access stream transformers on non-zeno signals with essentially the same proof.

7 Concluding remarks

We have given an introduction to streams and stream transformations with an emphasis on transformations of both discrete and continuous time streams and their connections. We have posed the problem of creating a unified semantic framework for analysing the computability of the 16 different kinds of stream transformations

$$F: (R \to B) \to (T \to A)$$

that depend upon whether time T and R, and data A and B, are discrete or continuous.

In this paper we have given a solution to the problem. It is based on the theory of algebraic domain representations for topological spaces and algebras.

Specifically, we have demonstrated that the domain methods can be successfully applied to all cases of transformations of continuous streams.

In addition, we have explored the problem of representing discontinuous streams, such as commonly arise in models based on signals $T \to A$, where time T is continuous and data A is discrete. The computability of transformations of discontinuous streams is an interesting subject about which little is known, at present.

Other general approaches to computability in topological spaces may provide alternate solutions to the problem of creating a general theory of stream computability. For example, effective metric space theory [34] or Weihrauch's TTE [57] may be applied, although these approaches are known to be equivalent with algebraic domain representations [48].

References

1. J. A. ANDERSON AND E. ROSENFELD (eds.), *Neurocomputing*, MIT Press, 1988.
2. J. BLANCK, Domain representability of metric spaces, *Annals of Pure and Applied Logic* 83 (1997), 225–247.
3. J. BLANCK, Domain representations of topological spaces, *U.U.D.M. Report* 1997:26, 1997. *Theoretical Computer Science*, to appear.
4. L. BLUM, M. SHUB AND S. SMALE, On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions, and universal machines, *Bulletin of the American Mathematical Society* 21 (1989), 1-46.
5. V. BRATTKA, Recursive characterisation of computable real valued functions and relations, *Theoretical Computer Science* 162 (1996), 45-77.
6. M. BROY, F. DEDERICHS, C. DENDORFER, M. FUCHS, T. F. GRITZNER AND R. WEBER, The design of distributed systems — An introduction to FOCUS, *Technical Report TUM* 19202-2, Institut f¸r Informatik, Technical University Munich, 1993.
7. G. S. CEITIN, Algorithmic operators in constructive complete separable metric spaces, *Doklady Akademii Nauk SSSR* 128, 49–52, 1959.
8. J. P. CRUTCHFIELD AND K. KANEKO, Phenomenology of spatio-temporal chaos, in H Bai-lin (ed.), *Directions in Chaos*, World Scientific, 1987.
9. N. J. CUTLAND, *Computability: An Introduction to Recursive Function Theory*, Cambridge University Press, 1980.
10. S. M. EKER AND J. V. TUCKER, Specification and verification of synchronous concurrent algorithms: a case study of the Pixel Planes architecture, in P. M. Dew et al. (eds.), *Parallel processing for computer vision and display*, Addison-Wesley, 1989, 16-49.
11. S. M. EKER, V. STAVRIDOU AND J. V. TUCKER, Verification of synchronous concurrent algorithms using OBJ3. A case study of the Pixel Planes architecture, in G Jones and M Sheeran (eds.), *Designing Correct Circuits*, Springer, 1991, 231-252.
12. YU. L. ERSHOV,Theorie der Numerierungen III, *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 23 (1977), 289-371.
13. F. FOGELMAN SOULIE, Y. ROBERT AND M. TCHUENTE (eds.), *Automata networks in computer science*, Manchester University Press, 1986.
14. P. DI GIANANTONIO, Real number computability an domain theory, *Information and Computation* 127 (1996) 11–25.

15. E. R. Griffor (ed.), *Handbook of Computability Theory*, Elsevier, to appear.
16. R. L. Grossman, A. Nerode, A. P. Ravn and H. Rischel (eds.), *Hybrid Systems*, Lecture Notes in Computer Science vol. 736, Springer-Verlag, 1993.
17. A. Grzegorczyk, Computable functionals, *Fundamenta Mathematicae* 42, 168–202, 1955.
18. N. A. Harman and J. V. Tucker, Clocks, retimings, and the formal specification of a UART, in G Milne (ed.), *The fusion of hardware design and verification (Proceedings of IFIP Working Group 10.2 Working Conference)*, North-Holland, 375-396.
19. N. A. Harman and J. V. Tucker, The formal specification of a digital correlator I : User specification process, in K McEvoy and J V Tucker (eds.), *Theoretical foundations of VLSI design*, Cambridge University Press, 1990, 161-262.
20. N. A. Harman and J. V. Tucker, Consistent refinements of specifications for digital systems, in: P Prinetto and P Camurati (eds.), *Proceedings of ESPRIT BRA CHARME Advanced Research Workshop on Correct hardware design methodologies*, Elsevier, Amsterdam, 1991, 281-304.
21. N. A. Harman and J. V. Tucker, Algebraic models computers and the correctness of micro processors, in G J Milne and L Pierre (eds.), *Correct hardware design and verification methods (Proceedings of IFIP Working Group 10.2, May 1993)*, Springer Lecture Notes in Computer Science 683, Berlin, 1993, 92-108.
22. N. A. Harman and J. V. Tucker, Algebraic models of microprocessors: architecture and organisation, *Acta Informatica* 33 (1996) 421-456.
23. N. A. Harman and J. V. Tucker, Algebraic models of microprocessors: the verification of a simple computer, in V Stavridou (ed.), *Mathematics for dependable systems II, Proceedings of the Second IMA Conference*, Oxford University Press, 1997, 135-169
24. K. M. Hobley, B. C. Thompson and J. V. Tucker, Specification and verification of synchronous concurrent algorithms: a case study of a convolution algorithm, in G Milne (ed.), *The fusion of hardware design and verification (Proceedings of IFIP Working Group 10.2 Working Conference)*, North-Holland, 347-374.
25. K. Hobley and J. V. Tucker, Clocks, retimings and the transformation of synchronous concurrent algorithms. in G Megson (ed.), *Transformational approaches to systolic design*, Chapman Hall, 1994, 99-132.
26. A. V. Holden, J. V. Tucker, H. Zhang and M. Poole, Coupled map lattices as computational systems, *American Institute of Physics - Chaos* 2 (1992) 367-376.
27. G. Kahn, The semantics of a simple language for parallel processing, in *Proceedings IFIP Congress* 74, IFIP, 1974, 471-475
28. K. Kaneko (ed.), *Coupled Map Lattices - Theory and Applications*, Wiley, 1993.
29. D. Lacombe, Extension de la notion de fonction rÈcursive aux fonctions d'une ou plusieurs variables rÈelles I, II, III, *Comptes Rendus de l'AcadÈmie des Sciences, SÈrie A* 240, 2478–2480, and 241, 13–14, 151–153, 1955.
30. A. I. Mal'cev, *The Metamathematics of Algebraic Systems. Collected Papers: 1936-1967*, North-Holland, Amsterdam, 1971, 148-212.
31. K. McEvoy and J. V. Tucker, Theoretical foundations of hardware design, in K McEvoy and J. V. Tucker (eds.), *Theoretical foundations of VLSI design*, Cambridge University Press, 1990, 1-62.
32. G. Megson, *An introduction to systolic algorithm design*, Oxford University Press, 1992.
33. K. Meinke and J. V. Tucker, Scope and limits of synchronous concurrent computation, in F. H. Vogt (ed.), *Concurrency* '88, Springer Lecture Notes in Computer Science 335, Springer-Verlag, 163-180.

34. Y. N. MOSCHOVAKIS, Recursive metric spaces, *Fundamenta Mathematicae* 55 (1964), 215-238.
35. D. NORMANN, The continuous functionals of finite types over the reals, manuscript, 1998.
36. M. J. POOLE, J. V- TUCKER AND A. V. HOLDEN, Hierarchies of spatially extended systems and synchronous concurrent algorithms, this volume.
37. M. B. POUR-EL AND J. I. RICHARDS, *Computability in Analysis and Physics*, Perspectives in Mathematical Logic, Springer-Verlag, Berlin, 1989.
38. M. O. RABIN, Computable algebra, general theory and theory of computable fields, *Transactions of the American Mathematical Society* 95 (1960), 341-360.
39. A. RABINOVICH, Automata over continuous time, manuscript,1997.
40. H. ROGERS, JR., *Theory of Recursive Functions and Effective Computability*, McGraw-Hill, New York, 1967.
41. R. STEPHENS, A survey of stream processing, *Acta Informatica* 34 (1997), 491-541.
42. V. STOLTENBERG-HANSEN, I. LINDSTRÖM AND E. R. GRIFFOR, *Mathematical Theory of Domains*, Cambridge University Press, 1994.
43. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Complete local rings as domains, CTCS Report 1.85, University of Leeds, 1985.
44. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Complete local rings as domains, *Journal of Symbolic Logic* 53 (1988), 603-624.
45. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Algebraic and fixed point equations over inverse limits of algebras, *Theoretical Computer Science* 87 (1991), 1-24.
46. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Infinite systems of equations over inverse limits and infinite synchronous concurrent algorithms, in J. W. de Bakker and W.-P. de Roever and G. Rozenberg (eds.), *Semantics - Foundations and Applications.* Lecture Notes in Computer Science vol 666, 1993, 531-562.
47. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Effective algebra, in S. Abramsky et al. (eds.), *Handbook of Logic in Computer Science*, vol. IV, Oxford University Press, 1995, 357–526.
48. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Concrete models of computation for topological algebras, *Theoretical Computer Science* , to appear.
49. V. STOLTENBERG-HANSEN AND J. V. TUCKER, Computable rings and fields, in E. R. Griffor (ed.), *Handbook of Computability Theory*, Elsevier, to appear.
50. B. C. THOMPSON AND J. V. TUCKER, Equational specification of synchronous concurrent algorithms and architectures, *Computer Division Research Report CSR* 9-91, University College of Swansea, 1991. Second Edition 1994.
51. B. TRAKHTENBROT, Origins and metamorphoses of the Trinity: Logics, Nets, Automata, in *Proceedings of Logic in Computer Science*, IEEE Computer Society Press, 1995, 506-507.
52. J. V. TUCKER AND J. I. ZUCKER, *Program Correctness over Abstract Data Types with Error-State Semantics*, North-Holland, Amsterdam, 1988.
53. J. V. TUCKER AND J. I. ZUCKER, Theory of computation over stream algebras, and its applications, in I. M. Havel and V. Koubek (eds.), *Mathematical Foundations of Computer Science* 1992, 17th International Symposium, Prague, Springer Lecture Notes in Computer Science 629, Berlin, 62-80.
54. J. V. TUCKER AND J. I. ZUCKER, Computable functions on stream algebras, in H. Schwichtenberg (ed.), *Proof and Computation*, Proceedings of NATO Advanced Study Institute, International Summer School 1993 at Marktoberdorf, Springer, 1994, 341-382.

55. J. V. Tucker and J. I. Zucker, Computable functions and semicomputable sets on many sorted algebras, in S. Abramsky et al. (eds.), *Handbook of Logic in Computer Science*, vol. V, Oxford University Press, to appear.
56. J. V. Tucker and J. I. Zucker, Computation by "While" programs on topological partial algebras, *University of Wales Swansea Computer Science Report* 10.97, 1997. *Theoretical Computer Science*, to appear.
57. K. Weihrauch, *Computability*, EATCS Monographs on Theoretical Computer Science 9, Springer-Verlag, Berlin, 1987.
58. G. J. Whitrow, *The natural philosophy of time*, Second Edition, 1980.
59. S. Wolfram (ed.), *Theory and applications of cellular automata*, World Scientific, Singapore, 1986.