Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Languages and models for hybrid automata: A coalgebraic perspective

Renato Neves *, Luís S. Barbosa

INESC TEC (HASLab) & Minho University, Portugal

ARTICLE INFO

Article history:
Received 15 April 2017
Received in revised form 28 August 2017
Accepted 30 September 2017
Available online xxxx

Keywords:
Coalgebra
Hybrid automata
Bisimulation
Regular expression

ABSTRACT

We study hybrid automata from a coalgebraic point of view. We show that such a perspective supports a generic theory of hybrid automata with a rich palette of definitions and results. This includes, among other things, notions of bisimulation and behaviour, state minimisation techniques, and regular expression languages.

1. Introduction

1.1. Motivation and context

Systems whose behaviour has both discrete and continuous aspects are traditionally qualified as ***hybrid*** [1–3]. They often arise as complex networks of computational units, sensors, and actuators, suitably coordinated so that a desired outcome can be reached. A classic example is provided by a cruise control system as it is essentially a digital device that interacts with actuators and sensors which control and measure a vehicle's velocity. The same pattern also occurs in thermostats, planes, electric grids, and surgical robots (see *e.g.* [3,2,4]).

The formal specification and analysis of hybrid systems typically resorts to the theory of ***hybrid automata*** [5], whose distinguishing feature is the ability of state variables to evolve continuously — thus making them able to express the behaviour of physical processes, like movement, time, temperature, or pressure. In addition, they carry syntactical machinery (guards, state invariants, and assignments) to facilitate the description of complex behaviour in a concise manner.

Being perhaps the most famous answer to the rapid emergence of hybrid systems [6–8], hybrid automata form an active research area that encompasses a broad range of topics, from decidability issues [5] to extensions that cater for ***input*** mechanisms [6,9] and ***uncertainty*** [10,11]. To create a new extension, however, frequently entails a return to the drawing board in order to redesign or adapt whatever definitions, notions or techniques are deemed relevant for it. The notion of ***bisimulation*** is a prime example of this, as it usually takes an apparently different form in each extension.

In a previous paper [12] we showed how to treat a number of variants of hybrid automata in a uniform manner using the theory of ***coalgebras*** [13]. In particular, we proved that the notions of bisimulation adopted by different types of hybrid automata are instances of a generic, coalgebraic definition; and we discussed briefly how to introduce new variants of hybrid

* Corresponding author.
E-mail addresses: nevrenato@di.uminho.pt (R. Neves), lsb@di.uminho.pt (L.S. Barbosa).

https://doi.org/10.1016/j.tcs.2017.09.038
0304-3975/

automata in a systematic way, with notions of bisimulation and behaviour coming for free and tailored to the context at hand. This was illustrated by reconstructing the theory of some classes of hybrid automata (for example, ***reactive Markov*** hybrid automata), and developing new variants (for example, of what was then called ***replicating*** hybrid automata). On the whole, that paper provided, we believe, a first step towards a coalgebraic, uniform theory of hybrid automata.

1.2. Contributions

In the current paper we give a complete and formal account of the research avenue announced in [12]. More concretely,

- we show that every functor $\mathsf{F} : \mathsf{Set} \to \mathsf{Set}$ induces a category of a specific type of hybrid automata (F shapes their 'discrete' transition type) and also a category that suitably captures their semantics. Both are categories of coalgebras and therefore several useful notions come for free.
- We prove the existence of a 'semantics' functor between these two categories, which, intuitively, associates every hybrid automaton to its corresponding model. This functor generalises the standard semantics for both ***classic hybrid automata*** [5] and ***probabilistic hybrid automata*** [11].

The idea of seeing hybrid automata as coalgebras emerged from our adoption of the ***black-box*** perspective as a strategy to handle hybrid systems (*cf.* [14]). In this view, the (discrete) state transitions of a hybrid system are internal, hidden from the environment whereas the continuous evolutions are external, making up the observable behaviour — think again about the operation of a cruise control system. One cannot directly observe the computations of the digital device; only their influence over the car's velocity which evolves over time. The black-box approach is a central concept in the theory of coalgebras and thus it naturally leads to the idea of regarding hybrid automata as coalgebras.

As discussed in [12], the coalgebraic perspective facilitates the analysis and design of hybrid automata once a suitable, coalgebraic semantics for them is set. For example, it provides a uniform, canonical notion of behaviour that faithfully reflects the black-box perspective and frames the behaviour into well known constructions (*e.g.* streams, binary trees) that mark a clear frontier between the discrete behaviour and the continuous one. Interestingly, the coalgebraic view also facilitates the understanding of hybrid automata and helps to systematise the concept along a plethora of, often elaborated, definitions in the literature. In its most basic variant, a hybrid automaton is reduced to a machine that from a state (internally) jumps to another and (externally) produces a continuous evolution. Moreover, the coalgebraic characterisation paves the way to a hierarchy of different types of hybrid automata organised with respect to their 'expressivity', a concept which is itself understood here coalgebraically.

In order to discuss some of these benefits in detail, we devote a large portion of the paper to a specific variant of hybrid automata classified as ***reactive*** — intuitively, it combines deterministic evolutions with an input dimension. We thoroughly study the coalgebraic theory of this variant, with special focus on the notions of behaviour and bisimulation. Furthermore, we use standard coalgebraic techniques to show that reactive hybrid automata admit a Kleene-like theorem. Along the process an illuminating message emerges: hybrid automata are hybrid also in the sense that they are neither purely syntactic nor purely semantic entities.

1.3. Roadmap

In Section 2 we recall briefly the theory of hybrid automata [5] and the theory of coalgebras [13]. Then, in the same section, we start our study by showing that classic hybrid automata, and some of their variants (*e.g.* reactive hybrid automata), can be straightforwardly interpreted as coalgebras.

In Section 3 we explore and discuss the coalgebraic theory of reactive hybrid automata. In particular, we introduce the aforementioned Kleene-like theorem, the semantics functor associated with reactive hybrid automata, and some of its properties. We also study the notions of behaviour and bisimulation that coalgebras provide in the context of this variant.

In Section 4 we take the generic perspective. In particular, we establish the formal correspondence between functors $\mathsf{F} : \mathsf{Set} \to \mathsf{Set}$ and variants of hybrid automata that document [12] alludes to. This leads to the (re)discovery of several variants of hybrid automata (*e.g.* probabilistic [11], ***weighted***, and ***replicating***). In the same section we revisit the generic notion of Φ-bisimulation and the hierarchy of hybrid automata introduced in [12], but now under the light of the functorial semantics that the current paper provides.

Finally, in Section 5 we conclude and discuss future work.

We assume that the reader has basic familiarity with category theory [15] and topology [16]. Throughout the paper we use an arrow with a tail $f : A \rightarrowtail B$ to stress that a map $f : A \to B$ is injective. Dually, we use a two-headed arrow $f : A \twoheadrightarrow B$ to represent a surjection. We use $\mathcal{P}$ to denote the powerspace construction and $\mathcal{D}$ to denote the distribution space construction whose distributions have finite support. Finally, for two sets A and B we denote by B^A the set of maps from A to B. If A and B are topological spaces then B^A denotes the set of continuous maps from A to B.

2. Preliminaries

2.1. Classic and probabilistic hybrid automata

Let us start by formally introducing the notion of predicate and the classic definition of hybrid automata [5].

Please cite this article in press as: R. Neves, L.S. Barbosa, Languages and models for hybrid automata: A coalgebraic perspective, Theoret. Comput. Sci. (2017), https://doi.org/10.1016/j.tcs.2017.09.038