Thoughts on Hybrid Systems

Chand T. John
Supervisor: Dr. Jayadev Misra
Department of Computer Sciences
University of Texas at Austin
Austin, TX

December 21, 2014

Abstract

There is a growing recognition of the presence of dynamical systems in many fields. In areas ranging from economics to physics and computer science to biology, many open fundamental problems are known to relate to the central problems in dynamical systems theory. Physicists generally work more with continuous dynamical systems, rather than discrete dynamical systems which computer scientists may study. The difference is that a continuous dynamical system is almost always defined as a system with an evolution function that satisfies a certain set of differential equations, while a discrete dynamical system is described by a state space involving discrete states and events which trigger discrete transitions between them. However, as computer science has come into contact with problems such as traffic management, automotive control, and modeling biological cell networks, which involve both continuous *and* discrete dynamical systems, a new formalism has emerged. This new formalism, the concept of a *hybrid system*, combines both the continuous and discrete formalisms into one entity that has fueled much research in recent years. Here we will formally introduce the notion of a hybrid system as an extension (or special case) of the concept of a dynamical system that is appropriate for many problems of interest in a wide variety of fields. We will then present summaries of some papers written on hybrid systems in recent years and present ideas for future research in the field.

1 Introduction

A *dynamical system*, roughly speaking, is a set of objects which are allowed to change over time, while obeying a set of rules. Generally, dynamical systems come in two flavors: discrete and continuous. A discrete dynamical system involves step-by-step state changes. A continuous dynamical system changes with time–for each $t \in \mathbb{R}^{\geq 0}$, the system has an associated state $x(t)$. Often, the description of how a continuous dynamical system behaves is given by a differential equation such as $\partial x/\partial t = tx + x^2$. In contrast, a discrete dynamical system's behavior is described using a transition relation or state space graph. For instance, a finite automaton [18] is a discrete dynamical system.

However, recently there has been an increased interest in studying systems that exhibit both discrete and continuous behaviors. In particular, a topic of interest in computer science is to understand, model and manipulate a set of discrete programs or agents that interact in a real-time environment. Thus it is necessary to monitor both discrete and continuous changes in a system's state. Hybrid systems provide a model for these types of problems.

In Section 2, we define hybrid systems and two of the central problems concerning them. In Section 3 we give summaries of some papers written on hybrid systems theory and its applications. In Section 4 we consider ideas for future research.

2 Hybrid Systems

Here we formally define the notion of a hybrid system. Essentially we can visualize a hybrid system as a graph whose edges represent discrete transitions and whose vertices represent continuous state changes

Each edge has guarded assignments–if the system traverses an edge, the state of the system must satisfy the guards, and if the guards are satisfied, the associated assignments are applied. Each vertex contains an invariant that must be true when the system is at that vertex, and a set of evolution laws for the continuous variables, usually in the form of differential equations.

2.1 Formal Definition

Now we will introduce the formal definition of a hybrid system as given in [3].

A *hybrid system* is a sextuple $H = (Loc, Var, Lab, Edg, Act, Inv)$ where:

1. Loc is a finite set of vertices called *locations*;
2. Var is a finite set of real *variables*. There is a function, called a *valuation*, $\nu : Var \to \mathbb{R}$ which assigns values to the variables: if $x \in Var$, then $\nu(x) \in \mathbb{R}$ is the value assigned to x. V denotes the set of possible valuations ν.

   We define a *state* of the hybrid system H to be a pair (l, ν) where $l \in Loc$ and $\nu \in V$. Denote the set of possible states by Σ.
3. Lab is a finite set of *synchronization labels* that contains the *stutter label* $\tau \in Lab$. The meaning of the stutter label is described below.
4. Edg is a finite set of edges called *transitions*. Each transition is a quadtuple $e = (l, a, \mu, l')$ where $l \in Loc$ is a *source location*, $l' \in Loc$ is a *target location*, $a \in Lab$ is a synchronization label, and $\mu \subseteq V^2$ is a *transition relation*. For every location $l \in Loc$, there is a set $Con_l \subseteq Var$ of *controlled variables* and a *stutter transition* (l, τ, Id_{Con}, l) where

   $$(\nu, \nu') \in Id_{Con} \Leftrightarrow \forall x \in Var, x \in Con_l \Rightarrow \nu(x) = \nu'(x).$$

   A transition e is *enabled* in a state (l, ν) if for some valuation $\nu' \in V$, $(\nu, \nu') \in \mu$. The state (l', ν') is then a *transition successor* of (l, ν).
5. Act is a labeling function that assigns a set of *activities* to each location $l \in Loc$. An *activity* is a function from $\mathbb{R}^{\geq 0}$ to V. Each activity must be *time-invariant*: for every $l \in Loc$, $f \in Act(l)$, and $t \in \mathbb{R}^{\geq 0}$, we require that $(f + t) \in Act(l)$, where $(f + t)(t') = f(t + t')$ for all $t' \in \mathbb{R}^{\geq 0}$.

   For every $l \in Loc$, $f \in Act(l)$, and $x \in Var$, we define $f^x : \mathbb{R}^{\geq 0} \to \mathbb{R}$ so that $f^x(t) = f(t)(x)$.
6. Inv is a labeling function that assigns an invariant $Inv(l) \subseteq V$ to each location $l \in Loc$.

The hybrid system H is *time-deterministic* if for every location $l \in Loc$ and every valuation $\nu \in V$, there is at most one activity $f \in Act(l)$ with $f(0) = \nu$.

2.2 Central Problems

Two of the central problems in the field of hybrid systems involve analysis and design.

1. *Analysis:* Given a hybrid system H and a specification S, does H satisfy S? For example, does H avoid a certain unsafe region of its state space?
2. *Design:* Given a specification S, can we design a hybrid system H that satisfies S?

Below we will look at several papers published on hybrid systems. Many of the theoretical papers are concerned with analysis of hybrid systems, while some of the applications are more concerned with design.

3 Summaries of Papers

Here we summarize several recent papers published on the topic of hybrid systems. First we look at papers focusing mainly on the theory of hybrid systems, and then we look at applications of the theory to the fields of multirobot coordination, barcode programming, and biological system modeling.

3.1 Theoretical Aspects

3.1.1 Analysis of Hybrid Systems

The paper [14] aims to determine whether a given hybrid system H satisfies a given property P. We would like to find the answer to this question in finite time. However, it turns out that this problem can be undecidable in some cases. This is because the state space of a hybrid system is uncountable: the continuous states are the source of this. If we were asking this same question about purely discrete systems (with a finite state space) instead of hybrid systems, then the question would always be decidable since the whole state space could be searched. The paper focuses on finding cases of this problem which *are* decidable.

The process of *abstraction* can help us check whether H satisfies P. Abstraction extracts a finite discrete system F from H by partitioning the state space of H in finite time. Then we can check whether F satisfies P. For many interesting classes of hybrid systems, it is shown that, proving that F satisfies P is equivalent to or sufficient for proving that H satisfies P. Thus, for those systems, there is a way to decide whether H satisfies P in finite time.

3.1.2 Reachability Analysis

The best studied analysis problem for hybrid systems is reachability. The term reachability refers to the following question: given a hybrid system H and a set of target states T, is there an execution of H that starts in some initial state and eventually reaches one of the states in T? In [12], a method is described for verifying properties of hybrid systems using reachability of an *abstract* system (derived by the process of abstraction explained in the summary of the previous paper). The paper combines ideas from two areas: predicate abstraction, and polyhedral approximation of reachable sets in linear continuous systems.

Predicate abstraction refers to a way of partitioning the uncountable space $\mathbb{R}^n$ using k Boolean predicates $p_1, \ldots, p_k$. Each p_i represents a polyhedron in $\mathbb{R}^n$. For instance, if $k = 3$, then the predicates $(p_1 = FALSE, p_2 = TRUE, p_3 = TRUE)$ denote the region of $\mathbb{R}^n$ which is outside of the first polyhedron and inside the second and third polyhedra.

The reachability problem for linear hybrid systems [9], which are hybrid systems whose continuous variables all vary linearly, is undecidable [7]. However, due to linearity, polyhedral approximations are possible. The paper [12] uses these approximations in its algorithm for verification of hybrid systems. The paper [9] explores possible generalizations of linear hybrid systems that still allow polyhedral analysis.

3.1.3 Membership Questions

The general problem attacked in [7] is as follows. Given a hybrid system H and some partial information I about the system, we wish to determine whether there exists an execution of H that is consistent with I. More specifically, the paper considers the following three problems:

1. *Timestamp generation:* Given a *path*, i.e. a sequence of edges $e_1, \ldots, e_r$ through H, is there an execution of H that follows this path? If so, we want to generate a sequence $t_1, \ldots, t_r$ of times, where t_i is a possible time at which e_i could be traversed in the execution.

2. *Untimed traces:* Given a *trace*, i.e. a sequence of events $l_1, \ldots, l_m$ in H, is there an execution of H that corresponds to this sequence?

3. *Timed traces:* Given a *timed trace*, i.e. a sequence of events $l_1, \ldots, l_m$ in H and their corresponding timestamps $t_1, \ldots, t_m$, is there an execution of H that corresponds to this sequence?

The paper considers the computational complexity of these problems for the following three types of automata:

1. Timed automata

2. Timed automata with linear constraints

3. Linear hybrid automata

For all three types of automata, the timestamp generation problem is shown to be decidable in polynomial time. If no ϵ-transitions are allowed, then for all three types of automata, the timed and untimed trace problems are NP-complete. For timed automata with ϵ-transitions, the timed and untime trace problems are PSPACE-complete. Finally, if ϵ-transitions are allowed, then the timed and untimed trace problems for timed automata with linear constraints and linear hybrid automata are undecidable.

3.1.4 Hierarchical Modeling and Refinement

One way to deal effectively with a complex object is to describe the object using hierarchies. A hybrid system can be thought of as a system in which digital programs interact with each other in an analog environment. Accordingly, we will describe a complex hybrid system using the notions of architectural and behavioral hierarchy. First we represent a hybrid system as an *architectural hierarchy*, where we treat the system as a group of individual agents interacting with each other. Then we represent each agent (digital program) by a *behavioral hierarchy* where the description of the agent's behavior is in a hierarchical form [5].

The computer language CHARON [13] [6] is useful for modeling hybrid systems hierarchically. The language allows the individual agents to interact through shared variables, while allowing each agent to have its own hidden local variables and control flow. The control flow in an agent is called a *mode*. The mode is always a hierarchical state machine made up of a tree of submodes. The state of a hybrid system changes both through guarded discrete transitions and continuous clock-based transitions.

The paper [5] shows that an execution of a mode can be traced by tracing the execution of its submodes, a result known as *compositionality*. Then it explains that, if an implementation design I is obtained from a specification design S by replacing a mode N in S with a new mode M, then to show that I refines S, it suffices to show that M refines N. A mode M *refines* a mode N if M and N have the same global variables and control points, and every trace of M is a trace of N.

3.2 Applications

3.2.1 Multirobot Coordination

The paper [4] explores the problem of coordinating a team of robots which explore a room with unknown geometry and obstacles, with the aim of locating these obstacles. This problem is militarily motivated, since using robots in hazardous areas to detect obstacles is safer than deploying humans to conduct the same task. Linear hybrid systems are used to model the robots as individual agents which must be coordinated, and the difficulties of this task are reported in the paper.

3.2.2 Programs in Barcodes

The central question addressed in the paper [16] is: can we make an interface that allows a user to customize an embedded system for his own needs? Currently, palm pilots and cell phones have interfaces that allow some degree of customizability, but there is much room for improvement. Embedded systems in other machines such as cars are only programmable (customizable) by their manufacturers.

There are two main concerns for getting embedded systems onto a device:

1. *Deliverability:* How easy it is to get the embedded system onto the device
2. *Predictability:* How well we know how the API will be used/misused by the user

The specific case studied in [16] is the problem of programming microwave ovens with two-dimensional barcodes. The design proposed in [16] is feasible and well-designed with respect to the concerns of deliverability and predictability.

3.2.3 Modeling Biological Systems

In [2], biological cell networks are modeled as hybrid systems. Results of two simulations described in the paper agree with experiment. These results suggest that hybrid systems can effectively model complex biological networks. Also, understanding the structure of biological networks can provide ideas on designing robust embedded systems. A potential new field could emerge from the studies of biological networks and

hybrid systems: *stochastic hybrid systems* could be useful in modeling biological processes such as biochemical reactions. Such reactions are represented by their rate equations, which can be written down as differential equations.

The paper [1] provides a description of BioSketchPad (BSP), a tool for modeling and designing biomolecular and cellular networks.

4 Future Research Ideas

We will consider two ways of looking at analysis problems for hybrid systems. One will be from a game-theoretic perspective, where an execution of a hybrid system will be modeled as a game between two players. This approach mainly looks at the *discrete* states and transitions. The other perspective we consider will be to model complex subsets of $\mathbb{R}^n$ using self-affine fractals. Such fractals admit relatively simple representations, thus aiding analysis of more complex state spaces. This approach focuses mainly on the *continuous* states of the system.

4.1 Game-Theoretic Approach

We shall introduce our game-theoretic perspective on hybrid systems through the analysis problem of determining whether a given hybrid system H can avoid all unsafe states. Denote the set of all locations (discrete states) by Loc, and denote the safe states by $Safe \subseteq Loc$. Denote the unsafe states by $Unsafe = Loc - Safe$. Now let us define a *game* between two players A and B, who take turns picking enabled transitions to follow. The game ends if H has landed in a location where there are no enabled transitions. If this event does not occur, the game can go on forever. In either case, we can express the safety question in terms of a game. Suppose that from any location, the player who must choose the next discrete transition to follow has the option the time at which he chooses to follow the transition. Suppose A's goal is to try to keep H inside $Safe$, the set of safe discrete states. Suppose B's goal is to try to move H into $Unsafe$. Then the safety question reduces to the following question: does A have a winning strategy in this game, regardless of whether A or B goes first? Now we can apply game theoretic methods to analyze the complexity of the game and determine what conditions specifically favor B's victory. We may also be able to compute some probabilities of various events occurring in the game.

4.2 Self-Affine Fractal Approach

An introduction to the generation of self-affine fractals using *iterated function systems* is given in [15]. As suggested in [17], the self-affine nature of polynomial Bézier curves can be used to represent self-affine fractals using only Bézier curves. Without going into the details of the theory behind this, let us just state that there are ways to model complex fractals using simple curves. Then, perhaps we could devise efficient alternatives to linear hybrid systems where we model the continuous state space using complex fractal regions that can be represented using only simple curves. This way we would be able to represent the state space easily, as is the case with polyhedra, but we would be able to deal with more complex sets. However, more research into the properties of self-affine sets must be done before we can connect fractal shapes in a general way to differential equations. Perhaps some other type of continuous transition should be used, such as a dynamical system on a fractal, instead of differential equations, in labeling the continuous transitions inside locations. How efficient fractal solutions will be is a bit unclear, since it depends largely on how we decide to label the transitions and represent fractals.

5 Conclusion

References

[1] R. Alur, C. Belta, F. Ivancic, V. Kumar, H. Rubin, J. Schug, O. Sokolsky, J. Webb, "Visual Programming for Modeling and Simulation of Biomolecular Regulatory Networks", *International Conference on High Performance Computing*, 2002.

[2] R. Alur, C. Belta, F. Ivancic, V. Kumar, M. Mintz, G. Pappas, H. Rubin, J. Schug, "Hybrid Modeling and Simulation of Biological Systems", *Fourth International Workshop on Hybrid Systems: Computation and Control*, LNCS 2034, pp. 19-32, 2001.

[3] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, S. Yovine, "The Algorithmic Analysis of Hybrid Systems". Shorter version of paper in *Theoretical Computer Science* **138**, 1995, pp. 3-34.

[4] R. Alur, J. Esposito, M. Kim, V. Kumar, I. Lee, "Formal Modeling and Analysis of Hybrid Systems: A Case Study in Multirobot Coordination", *FM '99: Proceedings of the World Congress on Formal Methods*, LNCS 1708, pp. 212-232, Springer, 1999.

[5] R. Alur, R. Grosu, I. Lee, O. Sokolsky, "Compositional Refinement of Hierarchical Hybrid Systems", *Fourth International Workshop on Hybrid Systems: Computation and Control*, LNCS 2034, pp. 33-48, 2001.

[6] R. Alur, R. Grosu, Y. Hur, V. Kumar, I. Lee, "Modular Specifications of Hybrid Systems in CHARON", *Proceedings of Third International Workshop on Hybrid Systems: Computation and Control*, LNCS 1790, pp. 6-19, 2000.

[7] R. Alur, R. Kurshan, M. Viswanathan, "Membership Problems for Timed and Hybrid Automata", *19th IEEE Real-Time Systems Symposium*, 1998.

[8] R. Alur, S. Kannan, M. Yannakakis, "Communicating Hierarchical State Machines", *Proceedings of the 26th International Colloquium on Automata, Languages, and Programming*, LNCS 1644, Springer, pp. 169-178, 1999.

[9] R. Alur, S. Kannan, S. La Torre, "Polyhedral Flows in Hybrid Automata", *Proceedings of the Second International Conference on Hybrid Systems: Computation and Control*, LNCS 1599, pp. 5-18, Springer, 1999.

[10] R. Alur, S. La Torre, G. Pappas, "Optimal Paths in Weighted Timed Automata", *Fourth International Workshop on Hybrid Systems: Computation and Control*, LNCS 2034, pp. 49-62, 2001.

[11] R. Alur, T. A. Henzinger, O. Kupferman, "Alternating-Time Temporal Logic", *38th IEEE Symposium on Foundations of Computer Science*, pp. 100-109, 1997.

[12] R. Alur, T. Dang, F. Ivancic, "Reachability Analysis of Hybrid Systems via Predicate Abstraction", *Fifth International Workshop on Hybrid Systems: Computation and Control*, 2002.

[13] R. Alur, T. Dang, J. Esposito, Y. Hur, F. Ivancic, V. Kumar, I. Lee, P. Mishra, G. Pappas, O. Sokolsky, "Hierarchical Modeling and Analysis of Embedded Systems", *First Workshop on Embedded Software*, 2001.

[14] R. Alur, T. Henzinger, G. Lafferriere, G. Pappas, "Discrete Abstractions of Hybrid Systems", Proceedings of the IEEE, 2000.

[15] M. Barnsley, *Fractals Everywhere*, Academic Press, 1988.

[16] A. Goodloe, M. McDougall, C. Gunter, R. Alur, "Predictable Programs in Barcodes", *International Conference on Compilers, Architecture and Synthesis of Embedded Systems* (CASES), 2002.

[17] C. John, "All Bézier Curves Are Attractors of Iterated Function Systems", *New York Journal of Mathematics*, to appear.

[18] M. Sipser, *Introduction to the Theory of Computation*, PWS Publishing, 1997.