Hybrid Systems and Quantum Automata: Preliminary Announcement

R. L. Grossman*
University of Illinois at Chicago

M. Sweedler†
Cornell University

May, 1995

This is a draft of a paper which later appeared in the proceedings Hybrid Systems II, P. Antsaklis, W. Kohn, A. Nerode, S. Sastry, editors, Springer Lecture Notes in Computer Science, Volume 999, pages 191-201, 1995.

Abstract

Let H denote an algebra of input symbols or events. If X is the state space for a system, then one can form the space R of observations of X. Under suitable conditions, both X and R are H-modules. Loosely speaking, the formal systems studied in this paper consists of a bialgebra H describing the input symbols and two H-modules describing the states and observations of the system. Finite automata and input-output systems are concerned with commutative R, while quantum systems, such as quantum automata, are concerned with non-commutative R arising from Hermitian operators on the state space. Of special interest are those systems consisting of interacting networks of classical systems and automata. These types of systems have become known as hybrid systems and are examples of formal systems with commutative R. In this paper, we present a number of examples of hybrid systems and quantum automata and point out some relationships between them. This is a preliminary announcement: a detailed exposition, including proofs, will appear elsewhere.

*This research was supported in part by NASA grant NAG2-513, DOE grant DE-FG02-92ER25133, and NSF grants IRI 9224605 and CDA 9303433. Part of the work was done while visiting the Design Research Institute and Computer Science Department at Cornell University.

†Sponsored in part by Army Research Office contract DAAL03-91-C-002.

1 Introduction

Loosely speaking, by a formal system we mean a system which accepts discrete input symbols, updates its state by flowing in a state space, and which is observed through outputs. A hybrid system as viewed in [7] is a formal system in which the observations have the structure of an algebra. A quantum automaton as viewed in [9] is a formal system in which the observations are Hermitian operators on the state space.

Automata, like other systems, have external and internal descriptions. The external description is given by the map from inputs to outputs, while the internal description views the system as a flow on suitable state space. In the latter description, the inputs determine the flow, while the outputs are a function of the flow. A realization of a system takes an external input-output description and produces an internal description involving a state space.

The purpose of this paper is to give some interesting examples of formal systems and their realizations and to point out some relationships between quantum automata and hybrid systems.

This work grew out of [6] and [7] whose goal was to use algebraic methods to study physical and engineering systems. Both systems modeling physical processes, such as a mechanical device, and engineering processes, such as the evolution of a computer program, can be understood using the basic concepts of inputs, outputs and states. Since we are concerned with the algebraic structure of such systems rather than the analytic structure, we speak of *formal systems.* For example, with the latter, attention would be paid to the convergence of series; with the former, it is sufficient whether the series are defined.

Formal Systems. Let k denote a field of characteristic zero. Let Ω denote the space of input symbols or input events, which we call the input alphabet. Let Ω^* denote the monoid of words formed from the alphabet Ω. The product in Ω^* is concatenation. Let $H = k\Omega^*$ be the algebra over k formed by taking finite formal sums of words. The product in H arises from the product in Ω^*. H is a bialgebra, which gives the dual an algebra structure.

If X is the state space for a system, then one can form the space R of observations of X. Under suitable conditions, both X and R are H-modules. Loosely speaking, a formal system consists of a bialgebra H describing the input symbols and two H-modules describing the states and observations of the system. Classical systems, such as those studied in [6] and [7], are concerned with commutative R, while quantum systems, such as quantum automata, are concerned with non-commutative R arising from Hermitian operators on the state space.

Of special interest are those systems consisting of interacting networks of classical systems and automata. These types of systems have become known as hybrid systems [5]. It turns out that hybrid systems can also be viewed as formal systems [7] and that the observation space of a hybrid system has a natural noncommutative structure, so that hybrid systems are naturally associated with quantum automata [8]. This suggests that a better understanding of quantum automaton would lead to a better understanding of hybrid systems, which is one of the motivations for this present work.

Related work. In this paper, we view a hybrid system as an interacting collection of nonlinear input-output systems, each corresponding to a different mode of the hybrid system: the role of the automaton is to switch modes. There are several other view points and many questions one can pose about hybrid systems. See [5] for a collection of papers discussing some of these.

The basic idea of quantum computing was introduced by Feynmam [4]. Important foundations were provided by the papers of Benioff [1] and Deutsch [3]. Recently, quantum computation has invaded complexity theory [2] and algorithm design [11].

Organization of this paper. In Section 2, we introduce our approach to quantum automata and hybrid systems with two simple examples. In Section 3, we provide some background material. Formal systems are defined in Section 4. Quantum automata recognizing arbitrary languages are described in Section 5 and those recognizing context free languages are defined in Section 6.

This is a preliminary announcement: a detailed version with proofs will appear later.

2 Two Examples

In this section, we give two important motivating examples. First we define define a quantum automaton which can recognize expressions which contain balanced parentheses. In the second example, we will define a simple hybrid system with two modes, each specifying a different nonlinear system in the plane. In both cases, we will define a space of input symbols H, a state space and an observation space.

It turns out that the following differences between these two examples are fundamental:

- For hybrid systems, the space of observations is an algebra R as well as an H-module, and these two structures are compatible.
- For quantum automata, the space of observations is still an H-module, but not necessarily an algebra.

For both examples, let k denote a field of characteristic zero.

Example 1. *A quantum automata recognizing parentheses expressions.* In this example, which is adapted from [9], the input alphabet Ω consists of two symbols l and r, denoting left and right parentheses. Let Ω^* denote the monoid consisting of words formed from the alphabet $\{l, r\}$. Let $H = k\Omega^*$ denote the k-algebra whose basis consists of words $w \in \Omega^*$ and whose multiplication is induced from the multiplication in Ω^*. In this way, we have defined the algebra of H of input symbols or input events.

Let X denote the vector space whose basis consist of the elements x_0, x_1, x_2, ..., and let x_i^* denote the dual basis of X^*. Assume that $H = k\Omega^*$ acts on basis elements of X as follows:

$$\begin{aligned} r \cdot x_i &= x_{i+1}, \qquad i \geq 0 \\ l \cdot x_i &= x_{i-1}, \qquad i \geq 1 \\ l \cdot x_0 &= 0 \end{aligned}$$

and is extended to act linearly on X. This defines an action of H on X which codes the dynamics of the quantum automaton.

Define a quantum observation $S : X \longrightarrow X^*$ via

$$\begin{aligned} Sx_i &= x_0^*, \qquad i \neq 0 \\ Sx_0 &= \sum_{i \geq 0} x_i^* \end{aligned}$$

and extend S to X so that $[Sx](y) = \overline{[Sy](x)}$, for all $x, y \in X$ so that S is Hermitian, as defined below.

It is also easy to check that the map

$$w \in \Omega^* \mapsto [S(w \cdot x_0)](w \cdot x_0)$$

is one precisely when the word w is a balanced expression in left and right parentheses and zero otherwise.

Let $L \subset \Omega^*$ denote the language consisting of well balanced parentheses expressions. Let $p \in H^*$ denote the characteristic series of the language L so that $p(w)$ is one precisely when $w \in L$. We can write the equation above

$$p(w) = [S(w \cdot x_0)](w \cdot x_0).$$

This can be interpreted as a realization. The left hand side consists of data involving the inputs and outputs, while the right hand side consists of an action of input symbols w on states coding the dynamics and a quantum observation S.

To summarize, we have constructed a quantum automaton which recognizes well balanced parentheses expressions. Recall that a finite automaton is not capable of recognizing such expressions.

Example 2. *A two mode hybrid system, switching between planar control systems.* In this example, which is adapted from [7]. we are given two nonlinear control systems on k^2

$$\dot{x}^{<\beta>}(t) = (u_1(t)E_1^{<\beta>} + u_2(t)E_2^{<\beta>})x(t), \qquad \beta = 1, 2$$

and an automaton which switches between them. Here $E_k^{<\beta>}$ for $\beta = 1, 2$ and $k = 1, 2$ are vector fields in the plane; that is derivations of some ring R_j of functions on the plane k^2, and $t \mapsto u_k(t)$ are controls. For example, one can take the ring of polynomials $R_j = k[x_1, x_2]$ or formal power series $R_j = k[[x_1, x_2]]$, for $j = 1$, 2.

The goal is to describe the structure of the entire ensemble consisting of the two nonlinear systems and the automaton switching between them. We begin by describing the space of observations of the systems. Since the space of observations of the nonlinear system for mode j is simply R_j, it is natural to take the space R of observations for the entire hybrid system to be $R = R_1 \oplus R_2$, [6].

The space of input events Ω for the automaton consists of two symbols β_1, which we interpret as "move to mode 1" and β_2, which we interpret as "move to mode 2."

In this example, the algebra of input events for each of the nonlinear input-output systems is the free associative algebra $k{<}\xi_1, \xi_2, \beta_1, \beta_2{>}$ in the indeterminates ξ_1, ξ_2, β_1, and β_2.

We specify the action of ξ_1 and ξ_2 on R by specifying its actions on R_i, for $i = 1$, 2: on R_1

$$\begin{array}{lll} \xi_1 & \text{acts as} & E_1^{<1>} \\ \xi_2 & \text{acts as} & E_2^{<1>} \end{array}$$

on R_2

$$\begin{array}{lll} \xi_1 & \text{acts as} & E_1^{<2>} \\ \xi_2 & \text{acts as} & E_2^{<2>} \end{array}$$

We next specify the action of β_1 and β_2 on R:

$$\begin{array}{rcl} \beta_1(f \oplus g) & = & f \oplus f \\ \beta_2(f \oplus g) & = & g \oplus g. \end{array}$$

Intuitively, β_1 maps all states into State 1, and β_2 maps all states into State 2. The action of β_1 on R is the transpose of this map. For the element

$$(u_1\xi_1 + u_2\xi_2)\beta_2(v_1\xi_1 + v_2\xi_2) \in H,$$

and state i, this is to be interpreted as flowing along $v_1 E_1^{<i>} + v_2 E_2^{<i>}$, making a transition to State 2, and then flowing along $u_1 E_1^{<2>} + u_2 E_2^{<2>}$.

To summarize, the space of input events for this two mode hybrid system is the algebra

$$H = k{<}\xi_1, \xi_2, \beta_1, \beta_2{>},$$

while the algebra of observations is

$$R = R_1 \oplus R_2.$$

We have showed how R has an H-module structure. It turns out that the algebra structure of R is compatible with the H-module structure so that R has the structure of what is called an H-module algebra. Note that R is commutative. The space of observation functions R captures the state space of the hybrid system, while the action of H on R captures the dynamics. Any element $f \in R$ can be thought of as an observation of the hybrid system. For more details, including a description of the realization of this system, see [7].

3 Preliminary Material

To describe more complicated examples requires some preparation, which is the subject of this section. This section is adapted from [9], which provides additional background material. Let k denote a field with conjugation. Let X denote a vector space over k and X^* its k-linear dual.

Hermitian forms and operators. A linear operator

$$S : X \longrightarrow X^*$$

naturally defines an inner product on X

$$<x, y>_S = [S(x)](y), , \qquad x, y \in X.$$

Conversely, for any inner product $<<x, y>>$, we can define a linear operator $T : X \longrightarrow X^*$ via the formula

$$[T(x)](y) = <<x, y>>.$$

The operator S is called Hermitian in case

$$[S(x)](y) = \overline{[S(y)](x)}, \qquad x, y \in X.$$

This is equivalent to $<\cdot, \cdot>_S$ being a Hermitian inner product.

Bialgebras. In this section, which is adapted from [7], we briefly cover some facts about bialgebras and H-modules, which we require in order to a give a careful definition of formal systems. This information is repeated here to make this paper self-contained. For more details, see [12].

An *algebra* A over the field k is a k-vector space A equipped with a multiplication $A \otimes A \longrightarrow A$ mapping $a \otimes b \mapsto ab$ and a unit $k \longrightarrow A$ mapping $1 \in k \mapsto 1 \in A$. The algebra is called *augmented* if there is an algebra homomorphism $A \longrightarrow k$.

A *coalgebra* C over the field k is a k-vector space C equipped with a comultiplication $C \longrightarrow C \otimes C$ and a counit $C \longrightarrow k$. A *bialgebra* H over the field k is a k-vector space H which has both an algebra and a coalgebra structure such that the comultiplication and the counit maps are algebra homomorphisms, or equivalently, such that the multiplication and unit maps are coalgebra morphisms.

Coproduct Notation. If H is a coalgebra, it is convenient to write the comultiplication $H \longrightarrow H \otimes H$ as the map $h \mapsto \sum_{(h)} h_{(1)} \otimes h_{(2)}$. This notation will be used throughout the remainder of the paper.

Here are two examples of bialgebras which occur in hybrid systems. Let G be a semigroup with unit. Then the semi-group algebra kG consisting of all formal finite linear combinations of elements of G with comultiplication defined by $g \mapsto g \otimes g$ and counit defined by $g \mapsto 1$ for $g \in G$, is a bialgebra. In this paper, this case arises when we have an input alphabet Ω and form the semigroup Ω^* of all words by concatenating input symbols from Ω. As another example, let L be a Lie algebra. Then the universal enveloping algebra $U(L)$ with comultiplication defined by $x \mapsto 1 \otimes x + x \otimes 1$ and counit defined by $x \mapsto 0$ for $x \in L$, is a bialgebra. In this paper, this example arises by viewing the free associative algebra $k{<}\xi_1, \xi_2, \ldots, \xi_n{>}$ as the universal enveloping algebra on the free Lie algebra generated by the symbols ξ_j and the Lie bracket defined by the commutator $[\xi_i, \xi_j] = \xi_i \xi_j - \xi_j \xi_i$.

Let H be a bialgebra. A vector space X is called an *H-module* in case there is an action of H on X denoted $h \cdot x$, for $h \in H$ and $\in X$, which is k-linear and which satisfies $(h_2 h_1) \cdot x = h_2 \cdot (h_1 \cdot x)$, where $h_2, h_1 \in H$ and $x \in X$. In this paper, the action of a H-module codes the dynamics of a formal system.

An algebra R with augmentation $\epsilon : R \longrightarrow k$ is called a left *H-module algebra* in case R is a left H-module and

$$h \cdot (ab) = \sum_{(h)} (h_{(1)} \cdot a)(h_{(2)} \cdot b),$$

and

$$h \cdot 1 = \epsilon(h) \cdot 1$$

for all a, $b \in R$, $h \in H$.

4 Formal Systems

After this preparatory material, we can now give definitions of formal systems, hybrid systems, and quantum automata.

A *formal system* over a field k with involution $\alpha \mapsto \bar{\alpha}$ consists of

1. a bialgebra H containing elements which we interpret as input symbols or events and the words formed from them;
2. an H-module $\mathcal{O}$, which we interpret as the space of observables.

A *hybrid system* is the special case in which the observables also form an algebra, while a *quantum automaton* is the special case in which the observables can be identified with Hermetian operators on an inner product space.

Here are two very basic examples. Let X denote a state space and assume there is an action of H on X, coding the dynamics. A hybrid system arises by taking classical state space observations

$$\mathcal{O} = \{f : X \longrightarrow k\}.$$

A quantum automaton arises by taking Hermetian observations

$$\mathcal{O} = \{\text{Hermetian } S : X \longrightarrow X^*\}.$$

In both cases, the action of H on X induces an action of H on $\mathcal{O}$.

5 Quantum automata and languages

Algebra of input events. Consider a language defined over an input alphabet Ω. As usual, let Ω^* denote the monoid of words formed by concatenating input symbols and let $H = k\Omega^*$ denote the corresponding k-algebra formed from the vector space whose basis is the set of words $w \in \Omega^*$. In this way, we define a bialgebra of input events H.

State space. Let $X = k\hat{\Omega}^*$ denote infinite formal linear combinations over k of elements of Ω^*. This is the algebra of formal noncommutative power series in the alphabet Ω over the field k and turns out to be the state space of the quantum automaton. There is a conjugation, denoted $x \mapsto \bar{x}$, defined on X which for each power series, conjugates the coefficients and replaces words by its palindrome.

Suffix action. We now define an action of H on X, called the *suffix* action, which codes the dynamics For each word $v \in \Omega^*$, the suffix action, denoted $v \rightharpoondown$ is a linear map from H to H. For w a word in Ω^*, thought of a lying in X,

$$\begin{array}{rcll} v \rightharpoondown w & = & u, & uv = w \\ & & 0, & \text{otherwise} \end{array}$$

Here u is a word in Ω^*. The action $v \rightharpoondown$ extends to all of X by linearity. Also, by linearity the action $h \rightharpoondown$ extends to all $h \in H$.

One can check that the suffix action makes X an H-module in that $(uv) \rightharpoondown w = u \rightharpoondown (v \rightharpoondown w)$, for .

Hermetian observable. We now define a Hermitian form on the state space X: Let $\lambda \in \Omega^*$ denote the empty word. Define

$$<x, y> = \chi_\lambda(x\bar{y}), \qquad x, y \in X$$

where $\chi_\lambda : k\Omega^* \longrightarrow k$ is defined by

$$\begin{array}{rcll} \chi_\lambda(w) & = & 1, & w = \lambda \\ & & 0, & \text{otherwise.} \end{array}$$

Here $w \in \Omega^*$ and the map is extended to all of X by linearity.

Of course, we can view this as a Hermitian operator

$$S : X \longrightarrow X^*$$

via $[S(x)](y) = <x, y>$.

Realizations of an arbitrary language. Given a language $L \subset \Omega^*$, let $p \in H^*$ denote the characteristic series of the language, where

$$\begin{array}{rcll} p(w) & = & 1 & w \in L \\ & & 0 & \text{otherwise} \end{array}$$

We also define the element

$$x_L = \sum_{l \in L} l \in X.$$

Note that this makes sense even if the language is infinite since X contains infinite sums.

For consistency with the notation above, think of $x_0 = x_L$, which we view as an initial condition. One verifies directly [9] that

$$p(h) = \sum_h <h_{(1)} \rightharpoonup x_0, h_{(2)} \rightharpoonup x_0>.$$

Note that this equation uses the coalgebra structure of H. In the case that $h \in k\Omega^*$ is a word in Ω^*, this simplifies to

$$\begin{array}{rcll} p(h) = <h \cdot x_0, h \cdot x_0> & = & 1 & h \in L \\ & & 0 & \text{otherwise.} \end{array}$$

Summary. To summarize, given a language with input alphabet Ω, we defined a bialgebra of input symbols $H = k\Omega^*$, a state space $X = k\hat{\Omega}^*$, and an action (the suffix action) of H on X. We also defined a Hermitian observable $S : X \longrightarrow X^*$ and wrote down a quantum recognizer for the language. In other words, given an arbitrary language, we constructed a quantum automaton which recognizes it.

6 Quantum automata and context free languages

In this section, we consider quantum automata which recognize context free languages. Classically, these are recognized by push down automaton. This is related to the fact that a realization of a context free language satisfies a finiteness condition, which we describe in this section. This is analogous to the fact that a natural equivalence relation on words in Ω^* has finite index for regular automata [10]. This is also analogous to the notion of finite Lie rank defined in [6] for formal series associated to nonlinear input-output systems. See [7].

We proceed as above: given an input alphabet Ω, we define the bialgebra $H = k\Omega^*$ and the state space $X = k\hat{\Omega}^*$.

Finiteness Condition. For an integer $n > 0$, and define the ring of noncommuting polynomials in the indeterminates Z_1, ..., Z_n with coefficients from H

$$R_n = H\{Z_1, \ldots, Z_n\}.$$

It is important to note that the Z_j do not commute with the coefficients in H. We can also write this as

$$R_n = k(\Omega^* \cup \{Z_1, \ldots, Z_n\})^*.$$

Let A denote a ring containing a homomorphic image of H. We need to define what it means for an element $a \in A$ to be algebraic over H.

> *Equations.* A *PDA equation* in R_n is an expression of the form
>
> $$Z_j - \alpha_j(Z_1, Z_2, \ldots, Z_n) = 0,$$
>
> where $\alpha_j \in R_n$ and α_j does not contain the empty word λ nor the single term Z_i, for any i. Of course α_j can contain a term of the form wZ_i, where $w \in \Omega^*$ and $w \neq \lambda$.
>
> *Algebraic elements.* An element $a \in A$ is *algebraic* over H in the sense of formal languages in case there exists $n > 0$ and elements a_1 ..., a_n and equations $Z_1 - \alpha_1$, ..., $Z_n - \alpha_n \in R_n$ such that
>
> 1. $a_i = \alpha_i(a_1, \ldots, a_n)$
> 2. a lies in the subalgebra of A generated by $a_1, \ldots, a_n$ and (the homomorphic image of) H.
>
> *Algebraic elements of X.* Recall that X is the algebra of formal power series in the noncommuting variables in the alphabet Ω. Note that there is a subalgebra, which is a copy of $H \in X$ consisting of those power series which are only finite sums, and hence noncommuting polynomials. Having of H in X gives an action of H on X by left multiplication, but it is important to note that this does not give the suffix action. However, having H as a subalgebra of X, we can speak of elements of X as being *algebraic* over H.

In this approach to recognizing languages using quantum recognizers, there is a finiteness condition characterizing context free languages which is based upon the notion of algebraic elements. We have

Theorem 1 *Fix an alphabet Ω, the monoid of words Ω^* and the corresponding bialgebra of input events $H = k\Omega^*$. Define the state space $X = k\hat{\Omega}^*$. With the notation above, a language $L \subset \Omega^*$ with characteristic series $p \in H^*$ is context free iff there is a $x_0 \in X$ giving the quantum automata recognizer*

$$p(h) = \sum_h < h_{(1)} \rightharpoondown x_0, h_{(2)} \rightharpoondown x_0 >,$$

where the element $x_0 \in X$ is algebraic over H.

See [9] for a proof.

7 Conclusion

In this paper, we have defined formal systems and considered two cases: hybrid systems and quantum automata. We have also given several examples of formal systems and their realizations.

We view a hybrid system as an interacting collection of nonlinear input-output systems, each corresponding to a different mode, and an automaton which switches between them. As the example of the hybrid system in Section 2 illustrates, a hybrid system may be given by defining an algebra of input events H containing variables corresponding to both the continuous and discrete components of the hybrid system. An important advantage of our approach is that by viewing the space of observations $\mathcal{O}$ as fundamental, the variables corresponding to the discrete and continuous components of the hybrid system may be treated symmetrically. Each induces an action on the observation space $\mathcal{O}$ coding the dynamics. Variables corresponding to continuous components act as derivations; those corresponding to discrete components act as homomorphisms. The reason for viewing H as bialgebra and $\mathcal{O}$ as an H-module is to to treat these two actions symmetrically.

Finite automata recognize regular languages and correspond to formal systems in which the H-module $\mathcal{O}$ of observations is a commutative algebra. Push down automata recognize context free languages and correspond to formal systems in which the H-module $\mathcal{O}$ of observations consists of Hermetian operators. We identified the finiteness condition associated with quantum automata which recognize context free languages.

This illustrates the power of the fundamental idea of understanding dynamics by focusing on the action induced on observations rather than on the underlying state space itself.

References

[1] P. Benioff, "Quantum mechanical Hamiltonian models of Turing machines," *Journal of Statistical Physics,* Volume 29, pp. 515–546, 1982.

[2] A. Berthiaume and G. Brassard, "The quantum challenge to structural complexity theory," *Proceedings of the Seventh IEEE Conference on Structure in Complexity Theory,* IEEE, pp. 132–137, 1992.

[3] D. Deutsch, "Quantum theory, the Church-Turing principle and the universal quantum computer," *Proceedings of the Royal Society,* Volume A425, pp. 73–90, 1985.

[4] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics,* Voluem 21, pp. 467–488, 1982.

[5] R. L. Grossman, A. Nerode, A. Ravn, and H. Rischel, editors, "Hybrid Systems", Springer Lecture Notes in Computer Science, Springer-Verlag, Bonn, 1993.

[6] R. Grossman and R. G. Larson, "The realization of input-output maps using bialgebras," *Forum Mathematicum*, Volume 4, pp. 109–121, 1992.

[7] R. L. Grossman and R. G. Larson, "An algebraic approach to hybrid systems," *Journal of Theoretical Computer Science*, Volume 138, pp. 101–112, 1995.

[8] R. L. Grossman, A. Nerode, and M. Sweedler, "A hilbert space approach to hybrid systems using formal dynamical systems," submitted for publication.

[9] R. L. Grossman and M. Sweedler, "Quantum automata and their realizations," submitted for publication.

[10] J. E. Hopcroft and J. D. Ullman, *Formal Languages and their Relation to Automata,* Addison-Wesley, Reading, 1969.

[11] P. W. Shor, "Algorithms for the quantum computation: Discrete log and factoring," submitted for publication.

[12] M. E. Sweedler, *Hopf algebras,* W. A. Benjamin, New York, 1969.