Translating Time-course Gene Expression Profiles into Semi-Algebraic Hybrid Automata via Dimensionality Reduction *

A. Casagrande[1,2], K. Casey[4], R. Falchi[3], C. Piazza[1], B. Ruperti[3], G. Vizzotto[3], and B. Mishra[4,5]

[1] Dept. of Math. and Computer Science, University of Udine, Udine, Italy
[2] Institute of Applied Genomics, Udine, Italy
[3] Dept. of Crop Science and Agricultural Engineering
[4] Courant Institute of Mathematical Science, NYU, New York, U.S.A.
[5] NYU School of Medicine, 550 First Avenue, New York, 10016 U.S.A.
{casa,piazza}@dimi.uniud.it, {rachele.falchi,ruperti,vizzotto}@uniud.it, {mishra,kjc261}@nyu.edu

Abstract. Biotechnological innovations that allow one to sample gene expression (e.g. gene-expression arrays or real-time Polymerase Chain Reaction (PCR)), have made it possible to measure the gene expression levels of a biological system with varying degree of accuracy, cost and speed. By repeating the measurement steps at different sampling rates during the system evolution, one can both infer relations among the genes (e.g., with clustering techniques), as well as define a dynamic model of the underlying biological system. When a very large number of genes and measurements are involved, they raise several difficult algorithmic questions, as accurate model-building, checking and inference tasks are simply beyond the skills of any human expert for all but a few trivial examples. Hence, the automation of the required analysis task is currently viewed as a critical milestone, necessary for the development of systems biology. Semi-algebraic hybrid automata have already been proposed as a modeling formalism for biological systems (see, e.g., [17, 6]), and have demonstrated their abilities to handle complex biochemical pathways. Unfortunately, a suitable algorithmic program aimed at the construction of hybrid automata from biological measurement data still remains remote. This paper addresses this challenge with an automatic procedure to build semi-algebraic hybrid automata from gene-expression profile sequences. In order to reduce the size of the resulting automata and to minimize their analysis computational complexity, our approach exploits various dimensionality reduction techniques, i.e., correlation coefficients and rate distortion clustering, applied simultaneously to genes and to measurements. The paper concludes with several interesting experimental results, all aimed at the study of peach fruit.

* This work is developed within the framework of the HYCON Network of Excellence, contract number FP6-IST-511368 and partially supported by the projects PRIN 2005 2005015491 and PRIN 2004 2004079422_004 (Role of sugar signalling in peach fruit quality development) and by the regional project BioCheck. B.M. has been supported by funding from two NSF ITR grants and one NSF EMT grant.

1 Introduction

It is often said that progress in science is characterized by successive steps of measurement, arithmetization, algorithmization, and algebraization—each step representing in a succinct manner the intuitions collected in the earlier step. In biology, various breakthrough in biotechnology, e.g., sequencing, DNA synthesis, DNA amplification with PCR, high-throughput measurement of DNA/RNA abundance through real time PCR [5, 10, 14], SAGE or microarrays [18, 12], etc., have made it possible to obtain a numerical picture of the transcriptomic state of a cell at a certain instant and under certain conditions. Equipped with such a collection of numerical pictures of these states, one may organize them into a state-diagram for further statistical and algorithmic study of the dynamics implied by the state-transitions (see, e.g., [3, 9, 16, 4]). Computational systems biology has come to represent the many varied efforts within this framework, and yet, it shies away from the final step of the algebraization of biology. It may even not be clear what such a final step would entail.

Here, we propose a framework for the algebraization of biology, by examining the question of translating time-course data of numerical biological measurements into the well-studied structures of semi-algebraic hybrid automata [17, 7, 6]. We concede that this is a first step in this direction, and would require much additional collaboration with biologists, algebraists and computer scientists to establish its final theoretical foundation. In particular, we believe that this new field will need to borrow many ideas originally developed in the context of rate-distortion theory in communication engineering, where the notion of lossy-compression was rigorously studied by Shannon and Kolmogorov [19].

This paper highlights many such connections and provides several heuristic algorithms that can be used for practical data analysis. It concludes with a discussion of the possible future paths of the emerging area of "Algebraic Biology."

1.1 Semi-Algebraic Hybrid Automata

The notion of *Hybrid Automata* was first introduced [1] as a model and specification language for systems with both continuous and discrete dynamics, i.e., for systems consisting of a discrete program within a continuously changing environment. The simplest class of such models studied in computer science was the class of timed-automata to model asynchronous systems with many local clocks evolving at different but constant rates, while the system made discrete state transitions according to the local time. Subsequently, the field has seen many interesting and nontrivial generalizations (see, e.g., [2, 15, 7]). Here, we focus on one that is motivated by our interest in modeling biochemical processes.

First we introduce some notations and conventions. Capital letters Z_m, Z'_m, where $m \in \mathbb{N}$, denote variables ranging over $\mathbb{R}$. Analogously, Z denotes the vector of variables $\langle Z_1, \ldots, Z_k \rangle$ and Z' denotes the vector $\langle Z'_1, \ldots, Z'_k \rangle$; and Z^n denotes the vector $\langle Z^n_1, \ldots, Z^n_k \rangle$. The temporal variables T and T' model time and range over $\mathbb{R}^+$. We use the small letters $p, q, r, s, \ldots$ to denote k-dimensional vectors of real numbers. Occasionally, we will use the notation $\varphi[X_1, \ldots, X_m]$ to

stress the fact that the set of free variables of the first-order formula φ, denoted by *Free*(φ), is included in the set of variables $\{X_1, \ldots, X_m\}$. By extension, if $\{X^1, \ldots, X^n\}$ is a set of variable vectors, $\varphi[X^1, \ldots, X^n]$ indicates that the free variables of φ are included in the set of components of $X^1, \ldots, X^n$. Moreover, given a formula $\varphi[X^1, \ldots, X^i, \ldots, X^n]$ and a vector p of the same dimension as the variable vector X^i, the formula obtained by component-wise substitution of X^i with p is denoted by $\varphi[X^1, \ldots, X^{i-1}, p, X^{i+1}, \ldots, X^n]$. If in φ the free variables are just the components of X^i, we can compute the truth value of $\varphi[p]$.

We are now ready to formally introduce semi-algebraic hybrid automata already presented in [17] and further studied in [7, 6]. For each node of a graph, we have an invariant condition and a dynamic law. This dynamic law may depend on the initial conditions, i.e., on the values of the continuous variables at the beginning of the evolution in the state. The jumps from one discrete state to another are regulated by the activation and reset conditions. All these conditions are defined through first-order formulæ over the reals, i.e., over the first-order language of $(\mathbb{R}, 0, 1, +, \times, =, >)$.

Definition 1 ((Semi-Algebraic) Hybrid Automata - Syntax). *A* hybrid automaton $H = (Z, Z', \mathcal{V}, \mathcal{E}, Inv, \mathcal{F}, Act, Reset)$ *of dimension k consists of the following components:*

1. $Z = \langle Z_1, \ldots, Z_k \rangle$ *and* $Z' = \langle Z'_1, \ldots, Z'_k \rangle$ *are two vectors of variables ranging over the reals* $\mathbb{R}$*;*
2. $\langle \mathcal{V}, \mathcal{E} \rangle$ *is a directed graph; the objects,* $v \in \mathcal{V}$*, are called* locations*;*
3. *Each vertex* $v \in \mathcal{V}$ *is labeled by the formulæ* $Inv(v)[Z]$ *and* $Dyn(v)[Z, Z', T]$*;*
4. *Each edge* $e \in \mathcal{E}$ *is labeled by the formulæ* $Act(e)[Z]$ *and* $Reset(e)[Z, Z']$.

We say that H is semi-algebraic *if the constraints Inv, Dyn, Act, and Reset are first-order formulæ over the reals (i.e., over* $(\mathbb{R}, 0, 1, +, \times, =, >)$*).*

The semantics of hybrid automata is given in terms of continuous and discrete transitions.

Definition 2 (Hybrid Automata - Semantics). *A* state ℓ *of H is a pair* $\langle v, r \rangle$*, where* $v \in \mathcal{V}$ *is a location and* $r = \langle r_1, \ldots, r_k \rangle \in \mathbb{R}^k$ *is an assignment of values for the variables of Z. A state* $\langle v, r \rangle$ *is said to be* admissible *if* $Inv(v)[r]$ *is true.*
The continuous reachability transition relations $\xrightarrow{t}_C$*, where* $t > 0$ *is the transition elapsed time, between admissible states is defined as follows:*

$$\langle v, r \rangle \xrightarrow{t}_C \langle v, s \rangle \iff \begin{array}{l} \text{The equation } s = f_v(r, t) \text{ holds, and for each} \\ t' \in [0, t] \text{ the formula } Inv(v)[f_v(r, t')] \text{ is true.} \end{array}$$

The discrete reachability transition relation $\rightarrow_D$ *between admissible states is defined as follows:*

$$\langle v, r \rangle \rightarrow_D \langle u, s \rangle \iff \begin{array}{l} \text{The relation } \langle v, u \rangle \in \mathcal{E} \text{ holds, and the formulæ} \\ Act(\langle v, u \rangle)[r] \text{ and } Reset(\langle v, u \rangle)[r, s] \text{ are true.} \end{array}$$

Building upon continuous and discrete transitions, we can introduce the notions of *trace* and *reachability*. A trace is a sequence of continuous and discrete

transitions. A point s is reachable from a point r, if there is a trace from r and to s. We use $\ell \to \ell'$ to denote that either $\ell \xrightarrow{t}_C \ell'$, for some t, or $\ell \to_D \ell'$.

Definition 3 (Hybrid Automata - Reachability). *Let I be either $\mathbb{N}$ or an initial interval of $\mathbb{N}$. A* trace *of H is a sequence $\ell_0, \ell_1, \ldots, \ell_i$, with $i \in I$, of admissible states such that $\ell_{i-1} \to \ell_i$ holds for each $i \in I$ with $i > 0$ and continuous and discrete transitions are alternating. Such a trace is also denoted by $(\ell_i)_{i \in I}$. A point $r \in \mathbb{R}^k$* reaches *a point $s \in \mathbb{R}^k$ (in time t), if there exists a trace $\ell_0, \ldots, \ell_n$ of H such that $\ell_0 = \langle v, r \rangle$ and $\ell_n = \langle u, s \rangle$, for some $v, u \in \mathcal{V}$ (and t is the sum of the elapsed continuous transition times).*

In [17] we defined first-order formulæ over the reals which allow one to study the reachability problem over semi-algebraic hybrid automata. The problem is undecidable in the general case, since it is necessary to consider an infinite number of formulæ. However, in [7, 6] we introduced two classes of semi-algebraic automata over which we demonstrated the decidability of the reachability problem showing that it is sufficient to consider a finite number of formulæ. Moreover, we showed that this decidability result for the reachability problem is also the basis for the decidability of model checking with other more complex temporal logic formulæ, which can be used to analyze biochemical pathways.

However, this earlier work was based on the assumption that the hybrid automaton model was available and accurately captured the dynamics of the underlying biochemical system. Its means of construction, however, were left unspecified. Construction of such models from experimental time-course data is the subject of this paper. Specifically, this paper deals with a suitable approach for identifying a semi-algebraic hybrid automaton representation of a biochemical dynamic system, where the biochemical system is initially represented as a matrix of gene expression data sampled at many discrete time instants.

2 From Time-courses to Semi-Algebraic Automata

We would like to capture the activity of a biological system using the formalism of hybrid automata. Specifically, we aim to represent the concerted activity of an organism's gene expression and regulation using the discrete and continuous dynamics of semi-algebraic hybrid automata as defined above.

One of the main problems that arises when hybrid automata are used to represent biological systems is that each component (e.g. gene) is modeled with a continuous variable. As a consequence the resulting automaton has a high computational complexity. In particular, when first-order formulæ are used to study reachability, the number of variables occurring in the formulæ are multiples of the number of continuous variables of the automaton (see, e.g., [7]). Thus, we would like to reduce the complexity of the system under study by grouping genes that have similar dynamics together and then considering this compressed representation when building our automata. Of course, there will be some loss of information when one clusters the data in such a way, as such, it is important to handle any data compression in a responsible manner.

When we define an automaton to represent a biological system, another difficulty consists in the identification of the locations. If the system is represented as a system of differential equations, then we can immediately define a trivial automaton having just one location whose dynamical law is an algebraic approximation of the solutions to the differential equations. However, this dynamical law could be very complex, and it may be convenient to split the single location into multiple locations in order to get simpler dynamical laws. In our case, the system is represented as a set of time series data that captures the temporal evolution of the genes' expression. One would like to find points in time at which elements in the data substantially change their behavior and then consider locations that correspond to the intervals of time between these critical time points. That is, we would like a temporal partition of the time series such that the data is broken into a number of disjoint temporal windows, each of which represents some coordinated biological activity, and the boundaries of which correspond to significant reorganization of gene expression. One could then identify locations with individual temporal windows, thus building an automata whose discrete transitions correspond to significant organizational events in the system's gene regulation, and whose continuous dynamics correspond to periods of concerted co-expression. The construction of such an automata requires a long preprocessing or clustering phase which results in an automaton with a number of locations proportional to the number of distinct temporal windows.

The two problems stated above are both related to the creation of a compact or compressed representation of the biological system. On the one hand, grouping like genes together and considering the collective continuous dynamics of clusters of co-expressed genes allows one to reduce the complexity of the resulting automata by simplifying the dynamical laws. On the other hand, generating temporal windows allows one to reduce the number of locations from the order of the number of time points down to the order of the number of time windows. These considerations are directly related to compressing the original time series data both in the number of genes and in the number of experiments, and a variety of bi-clustering techniques have been explored for this purpose [16]. We will discuss two methods: one directly exploiting correlation among gene expressions and consecutive time points (through Principal Component Analysis, or PCA), and the other method emphasizing "lossy compression" of hybrid automata by building on rate distortion theory and graph search; these approaches show how one can ultimately go from clustered data of reduced dimension to a "reasonably faithful" hybrid automata model. First however, we step through a number of intuitively simple, but successively more complex, examples of representations of our time series data using hybrid automata. Next, we introduce several key ideas from information theory as well as our clustering algorithm. Finally, we will present a hybrid automata constructed from our time series that represents a significantly compressed version of the original data.

Let M be an $m \times n$ matrix of biological time series data, where $G_1,\ldots,$ G_n are the genes under consideration and $D_1,\ldots,D_m$ are the dates that the samples were captured on. We can define a semi-algebraic hybrid automaton

H representing M in various ways; we illustrate them in ascending order of complexity, beginning with a couple of trivial examples.

The most simple way to construct a hybrid automata from our time series data is to have a single location and a continuous representation of each gene's expression profile. Thus, for our our $m \times n$ matrix of expression values, we have n polynomials, where each of the n genes is represented by a polynomial of degree m (i.e. the number of time points). In this way we completely capture all of the information in our expression matrix without loss, in fact we can reconstruct our matrix of expression measurements exactly from this representation. Note that in this case we get an automaton without edges, since there is only a single location. Thus, there are no guard or reset conditions. This construction is close to the classical approach of using one system differential equations.

Rather than having a single location with polynomial representations of degree m for each gene, we could instead have m locations, one for each time point, and each of these could have linear dynamics for each gene. Clearly this is also completely equivalent to our original data and represents it without loss, in fact the representation can be seen as a simple distribution of the rows of our expression data across the m locations. In this case the discrete graph underlying the automaton is simply a chain and the automaton is linear.

The two examples above either use completely continuous or completely discrete representations of the dynamics, and are incapable of taking advantage of the hybrid nature of the dynamics, where it exists. Thus, one may be able to avoid automata of prohibitive complexity, by using a representation that reduces the dimensionality of the underlying data and yields automata with fewer locations and simpler dynamical laws. We can progress toward this goal by considering coordinated genes within suitably sized (initially, uniform length) windows of time and by letting the number of locations in our automata equal the number of windows under consideration. Next, we describe such an example with uniformly sized time windows; a version with nonuniform adaptively sized windows will be discussed below.

- the continuous variables are $G = \langle G_1, \ldots, G_n \rangle$ and $G' = \langle G'_1, \ldots, G'_n \rangle$;
- the directed graph $\langle \mathcal{V}, \mathcal{E} \rangle$ has $L = \lceil \frac{m}{h} \rceil$ locations $v_1, \ldots, v_L$ and its edges are defined as $\mathcal{E} = \{\langle v_i, v_{i+1} \rangle \mid 1 \leq i < L\}$;
- for each $v_i \in \mathcal{V}$ and each $e_i = \langle v_i, v_{i+1} \rangle \in \mathcal{E}$ we have: $Inv(v_i)[G] \stackrel{\text{def}}{=} \text{true}$; $Dyn(v_i)[G, G', T] = \wedge_{j=1}^{n} G'_j = p_{(i,j)}(T)$, where $p_{(i,j)}$ is the polynomial of degree at most h connecting the values of G_j at $D_{h*(i-1)+1}, \ldots, D_{h*i+1}$; $Act(e_i)[G, G'] = \wedge_{j=1}^{n} G_j = g_{(i,j)}$, where $g_{(i,j)}$ is the expression level of G_j at D_{h*i+1}; $Reset(e_i)[G, G'] = \wedge_{j=1}^{n} G'_j = G_j$ is the identity.

In the activation conditions we have implicitly assumed that the biological system has no memory. In fact, the activation considers only the final values of a state and not the trajectory which leads to these values. More sophisticated constraints are necessary to model systems with memory. The proposed automaton has a number of locations which depends on the number of dates and on the degree of the dynamical laws, and a number of variables which is proportional to

the number of genes under consideration. Again, the discrete graph underlying the automaton is simply a chain, but the automaton is not linear.

Example 1. Let us consider three genes G_1, G_2, and G_3 for which we have measured the following expression levels:

D	0	1	2
G_1	0.25	0.20	0.42
G_2	0.49	0.41	0.80
G_3	0.10	0.20	0.30

In Figure 1, we depict the hybrid automaton, built by applying the naïve methods, described above, with $h = 1$ and without time windows. The dynamics are written inside the locations, while the resets and activations are represented on the edges. The incoming edge on the left provides the initialization conditions which can be used to obtain the trace which corresponds to the expression levels measured in the matrix.

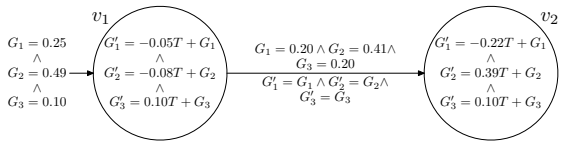

Fig. 1. The automaton of Example 1

Notice, that the fact that we are using only polynomial constraint is not too restrictive since: non-polynomial functions can be approximated with polynomials; polynomials can always interpolate finite sets of data.

In the following sections, we will explore methods to improve the automaton construction discussed above, by exploiting correlations, performing dimensionality reduction via correlation coefficients, Principal Component Analysis (PCA) [13], information theory [19], and exploiting clustering techniques [11].

2.1 Reductions via Correlations on Genes

Given a gene expression level matrix M having m rows corresponding to the dates and n columns corresponding to the genes we can interpret each column of M as a random variable and compute the correlation coefficients between pairs of genes. As a result we get an $n \times n$ symmetric matrix $Corr$, such that $Corr[i, j]$ is the correlation between G_i and G_j ranging in the interval $[-1, 1]$. We can now use the absolute values of the elements of $Corr$ as similarity measures (or equivalently set the distance between G_i and G_j to $d(G_i, G_j) = 1 - |Corr[i, j]|$). These similarity measures can be used to to cluster the genes. There are different clustering techniques which can be used (see, e.g., [11, 9]) leading to different results.

However, we do not use clustering techniques to infer properties of the biological system under investigation, but only to build a compact hybrid automaton representing it. The analysis of the automaton will then help us to understand the system behaviors. Another possibility is that of clustering the genes using PCA, i.e., using their correlation coefficients with respect to the new coordinate system. In both cases we obtain classes of (highly correlated) genes.

In each class of correlated genes we can choose a representative gene which is closer to all of the elements of the class and construct a hybrid automaton whose continuous variables are only the representative genes. The values of the non-representative genes can be approximated at any time from the representative ones exploiting their linear relationships. Alternatively, we could compute a cluster average as a continuous variable of a new fictitious representative gene, and use it to approximate the behavior of the non-representative genes, encoded through linear relationships. For the sake of simplicity, we focus on the former representation, here, and relegate the more complex treatment to the full paper.

Example 2. Let us consider the genes G_1, G_2, and G_3 of Example 1. The correlation coefficient of G_1 and G_2 is 0.99, while G_3 is less correlated both with G_1 and G_2 (0.74 and 0.75, respectively). If we apply hierarchical clustering on correlations we obtain a class with G_1 and G_2 together and another class containing G_3 only. Applying a clustering based on PCA in this case we obtain the same result. Hence, we can construct a hybrid automaton which has only G_1 and G_3 as continuous variables, i.e., the automaton of Figure 1 with G_2 deleted, and at any time we can infer the value of G_2 from G_1 ($G_2 \approx 2 * G_1$).

2.2 Reductions via Correlations on Dates

We would next like to exploit dates-correlations to cluster dates, yielding better adaptive time windows for the construction of our automaton . However, if we analyze what happens if we transpose our gene expression matrix M, i.e., we consider the dates as random variables, and compute the correlation coefficients, we notice that this not only provides a better time segmentation, but also, a compact symbolic representation of the transcriptomic dynamics of genes.

Considering the dates as random variables means that each observation represents the values of a gene at each date. We have to imagine a coordinate system in which each axis corresponds to a date. In this system we can plot a point for each gene: the coordinates of this point are the expression levels of the gene at the different dates. When two dates are highly linearly correlated it is sufficient to know the expression levels of the genes at the first data to approximate the levels at the second one. If more than two dates are highly correlated, then the levels at one of them are sufficient to reconstruct the levels at all the other dates. In particular, if the random variables (dates) $D_i, \ldots, D_{i+r}$ are highly correlated,

then we can relate them through a linear system of the form

$$\begin{cases} \widehat{G}(D_i) &= f_0(q) \\ \widehat{G}(D_{i+1}) &= f_1(q) \\ \dots & \\ \widehat{G}(D_{i+r}) &= f_r(q) \end{cases} \tag{1}$$

where $\widehat{G}$ is a symbolic variable gene expression, q is a parameter and the f_j's are linear function in q. If we know that the expression level of the gene G_j at D_{i+s} is $g_{s,j}$, then we can use it to determine the corresponding value of q, i.e., $q_{s,j} = f_s^{-1}(g_{s,j})$. Now by substituting the $q_{s,j}$ to q in the equation corresponding to the date D_u, we can approximate the expression level $g_{u,j}$ of G_j at D_{i+u}, i.e., $g_{u,j} \approx f_u(q_{s,j})$.

Since we are not only interested in the expression levels at the measured data, but we would like to reconstruct all the genes time evolution, we can apply interpolation techniques to obtain a dynamical law. To keep the presentation simple we discuss here the case of linear interpolation (see [3] for more sophisticated interpolation methods). We have that the expression level of the gene G_j at time t, where $D_{i+a} \leq t \leq D_{i+a+1}$, for some $a \in [0, r-1]$ can be approximated with:

$$\frac{f_{i+a+1}(q_{s,j}) - f_{i+a}(q_{s,j})}{D_{i+a+1} - D_{i+a}}(t - D_{i+a}) + f_{i+a}(q_{s,j}) \tag{2}$$

Hence, we can construct our hybrid automaton by using a single location for dates which are highly correlated and in these locations the dynamical laws are the same for all the genes and are given by system (1) together with expression (2). This means that our automata will now have a single $\widehat{G}$ variable able to represent all the genes. In the case in which there are blocks of non consecutive dates which are correlated we can still use one location for all of them and introduce a loop in the discrete topology of the automaton. In order to simplify the notation we present only the definition for the case of adjacent correlated dates (the general case is presented in Example 3). Let M be a gene expression matrix of dimension $m \times n$. Let us assume that we cluster the dates exploiting their correlation coefficients as follows: $Cl_1 = \{D_1, \dots, D_{d_1}\}$, $Cl_2 = \{D_{d_1+1}, \dots, D_{d_2}\}$, $\dots$, and $Cl_{cl} = \{D_{d_{(cl-1)}+1}, \dots, D_m\}$. The *dates reduced* automaton H representing M is $HD(M) = (G, G', \mathcal{V}, \mathcal{E}, Inv, \mathcal{F}, Act, Reset)$, where:

- $G = \langle \widehat{G} \rangle$ and $G' = \langle \widehat{G}' \rangle$;
- $\langle \mathcal{V}, \mathcal{E} \rangle$ has cl locations $v_1, \dots, v_{cl}$ and $\mathcal{E} = \{\langle v_i, v_{i+1} \rangle \mid 1 \leq i < cl\}$;
- for each v_i corresponding to $Cl_i = \{D_a, \dots, D_b\}$, where for each $a \leq c \leq b$ it holds $\widehat{G}(D_c) = f_c(q)$ and for each $e_i = \langle v_i, v_{i+1} \rangle$ we have: $Inv(v_i)[\widehat{G}] = \text{true}$, $Act(e_i)[\widehat{G}] = \vee_{j=1}^{n} \widehat{G} = M[b, j]$, $Reset(e_i)[\widehat{G}, \widehat{G}'] = \vee_{j=1}^{n} (\widehat{G} = M[b, j] \wedge \widehat{G}' = M[b+1, j])$, and

$$\begin{aligned} Dyn(v_i)[\widehat{G}, \widehat{G}', T] = \bigvee_{a \leq c < b} (D_c - D_a \leq T \leq D_{c+1} - D_a \wedge \\ \widehat{G}' = \tfrac{f_{c+1}(f_a^{-1}(\widehat{G})) - f_c(f_a^{-1}(\widehat{G}))}{D_{c+1} - D_c} T + f_c(f_a^{-1}(\widehat{G}))) \end{aligned}$$

In the above automaton we have reduced the states from m to cl without increasing the complexity of the involved formulæ. Notice that since the f's are linear, their inverses, f^{-1}'s, are still linear and the automaton is semi-algebraic. Moreover, it is important to notice that inside each state/location the continuous dynamics of the genes are all regulated by a single law. In fact, what changes from one gene to another is only the value of $\widehat{G}$. This drastically reduces the complexity of the analysis in many cases. Imagine for instance that we wish to check the following property: *Each time a gene reaches an expression level lower than low it never increases again enough to reach the expression level low′.* In each state v_i we can check this property at the same time for all the genes. We only have to write a first-order formula representing the values of $\widehat{G}$ which violate the property, and then check that all the initialization values of the genes that are outside of this set. In this sense we can say that the reduction based on dates correlation reduce both the number of states and of variables, which were our main objectives.

Due to the non-determinism introduced in the discrete jumps the date reduced automaton H correctly approximates the behaviors observed in M only, provided that in M there is not a date in which two genes have the same value. This assumption is not restrictive in the real cases.

Example 3. Let us consider the following transposed gene matrix:

D	0	1	2	3	4	5	6
G_1	1	2	3	4	8	4	5
G_2	2	3	4	1	2	3	4
G_3	3	4	5	3	6	1	2

This is a toy example in which we have a perfect correlation on the dates 0, 1, 2, 5, and 6 and a perfect correlation on the dates 3 and 4. The automaton we can build generalizing the technique to use also clusters of non adjacent dates is depicted in Figure 2. We label each edge only with the reset constraint, since the activation one can be reconstructed from it.

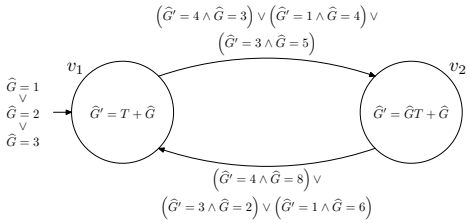

Fig. 2. The automaton of Example 3

3 Rate Distortion Theory and Extensions

In the above discussion we considered various ways of reducing the dimensionality of the data and deriving an automaton that captured the dynamics of this new compressed data set. As stated above, this can take the form of clustering the genes and subsequently using one gene from each cluster to approximate the others, or of considering windows of time to reduce both the number of locations and the number of variables in our hybrid automata. Finally, one could also reduce the complexity of the model used to represent the continuous dynamics, for example, one could use lower order polynomials or splines rather than polynomials of high degree. Each of these methods of simplifying our hybrid automata results in a distortion or disagreement between our model and the raw data. For instance, clustering forces us to live with discrepancies between the approximated profiles and the actual data vectors. What we really desire is a formalism to represent such distortions precisely, allowing us to specify an objective function that we can minimize, thus obtaining an optimal partition of our data and a low complexity automaton. We look to information theory for such a formalism and find it in the rate distortion theory of Shannon and Kolmogorov [8].

In rate distortion theory, one desires a compressed representation Z of a random variable X that minimizes some measure of distortion between the data elements $x \in X$ and their prototypes $z \in Z$. Taking $I(Z;X)$, the mutual information between Z and X, to be a measure of the compactness or degree of compression of the new representation, and defining a distortion measure $d(x,z)$ that measures distance between cluster prototypes and data elements, one can frame the problem as a trade-off between compression and average distortion. The main idea is that one balances the desire to achieve a compressed description of the data with the precision of the clustering, as measured by the average distortion, and finds the appropriate balance that maintains enough information while eliminating noise and inessential details.

In rate distortion theory, this trade-off is characterized mathematically as an optimization problem: $\mathcal{F}_{min} = I(Z;X) + \beta\langle d(x,z)\rangle$, where average distortion is defined as $\langle d(x,z)\rangle = \sum_{x,z} p(x)p(z|x)d(x,z)$ and is simply the weighted sum of the distortions between the data elements and their prototypes. More recently, Slonim et al. [20] have discussed a modification to rate distortion clustering for which only relations between data elements are used in the distortion function, rather than explicit mention of cluster prototypes. We have used a similar approach as a component in our graph search based approach to the time course segmentation problem.

Moving beyond classical rate distortion theory, we will need to generalize the problem further. In this generalized picture, we are presented with a family of time-course data all sampled from the same dynamical system; for example, k matrices of dimension $m \times n$. These matrices may be thought of as describing essentially the same dynamics, but corrupted by measurement noise, or affected by unmodeled/unmodelable environmental conditions. We may wish to introduce a notion of "distorted bisimulation", generalizing the idea of classical bisimulation, by allowing for certain constraints on allowable bisimulation. In this setting, it

makes perfect sense to ask for a minimal complexity hybrid-automata representation of the datasets, subject to a constrained "distorted bisimulation".

This general notion will be explored in more detail in the full paper, but here, we focus on the most immediate problem of compressing (with loss) a given time-course data set by means of a semi-algebraic hybrid automaton. Returning to our earlier discussion, in the specialized setting, we note that the functional above captures the compression-precision trade-off inherent in the clustering problem and when combined with a shortest path graph search algorithm (as described below), it allows one to use an iterative method to find a numerical solution to the time course segmentation problem. The trade-off is controlled by the Lagrange parameter β that mitigates the trade-off between compression and preservation of relevant information, as β becomes large we focus on precision, as β tends to zero we focus more on compression. Setting the clustering problem up in this way allows us to find both an optimal windowing of our data, and optimal clusters of genes within the windows. From this compressed representation, we can create a hybrid automaton having minimal disagreement with the original data.

3.1 Reductions via Rate Distortion

We would like to cluster our data in both the genes and in time, that is, we would like a procedure that yields windows in time and that captures intervals of concerted gene activity in which the genes are clustered into a small number of groups of co-expressed elements. From such a compressed representation, we can produce an automaton whose number of locations is the number of time windows, and for which the dynamical laws are less complex because we derive our continuous dynamics from the clustered data rather than from individual genes. We briefly discuss a method that performs this type of compression.

Let $D = \{D_1, D_2, \ldots, D_m\}$ be the time points at which a given system is sampled, and l_{min} and l_{max} be the minimum and maximum window lengths respectively. For each time point $D_a \in D$, we define a candidate set of windows starting from D_a as $S_{D_a} = \{W_a^b | l_{min} \leq D_b - D_a \leq l_{max}\}$, where W_a^b is the time window containing the dates $D_a, D_{a+1}, \ldots, D_b$. Each of these windows may then be clustered and labeled with a score based on its length and the cost associated with the clustering functional defined in (3). Following scoring, we formulate the problem of finding the lowest cost windowing of our time series in terms of a graph search problem and use a shortest path algorithm to generate the final set of (non-overlapping) time windows that fully cover the original series.

To score the windows, we use a variant of rate distortion clustering, using a distortion function defined between pairs of data elements. We aim to maximize compression (by minimizing the mutual information between the clusters and data elements), while at the same time forcing our clusters to have minimal distortion (as described in [20]). We perform rudimentary model selection by iterating over the number of clusters while optimizing (line search) over beta. This procedure, while somewhat expensive, results in a fairly complete sampling of the rate-distortion curves. Essentially, we trace the various solutions for different model sizes while tuning β, and choose the simplest model that achieves

minimal cost in the target functional. In this way we obtain for each window a score that is the minimum cost in terms of model size and model fit, based on the trade-off between compression and precision. This method is computationally expensive and run times can be substantial, for this reason we have developed an implementation that can take advantage of parallel hardware.

Once the scores are generated, we pose the problem of finding the lowest cost tiling of the time series as a graph search problem. We consider a graph $G = (V, E)$ for which the vertices are time points $V = \{D_1, D_2, \ldots, D_m\}$, and the edges represent windows with associated scores. Each edge $e_{ab} \in E$ represents the corresponding window W_a^b from time point D_a to time point D_b, and has an initially infinite positive cost. The edges are labeled with the costs for the windows they represent, each edge e_{ab} gets assigned a cost $(F_{ab} * length)$ where F_{ab} is the minimum cost found by the clustering procedure and length is the length of the window $(b - a)$. Our original problem of segmenting the time series into an optimal sequence of windows can now be formulated as finding the minimal cost path from the vertex D_1 to the vertex D_m. The vertices on the path with minimal cost represent the points at which our optimal windows begin and end. We use a shortest path algorithm and generate a windowing that segments our original time series data into a sequence of optimal windows which perform maximal compression in terms of the clustering cost functional. We are now in a position to sketch one possible way to construct hybrid automata that have a compact representation in time and that reflect clusters with respect to gene expression. Further, we construct our models to have minimal distortion with respect to the original data. We accomplish this by clustering using the method just discussed and then building an automata with the same number of locations as windows and simplified dynamical laws constructed from the clustered genes.

Hence, for each cluster we can choose a representative gene which minimizes the distance to all of the other genes in the cluster and construct a hybrid automaton whose continuous variables correspond to those of the representative genes. Further, our time windows naturally provide a means to simplify the dynamics of our model by exploiting correlations in time. Our graph based approach allows for a convenient method of locating repeated segments in the data that are correlated, i.e., loops in our automata can be readily located. We will provide a complete characterization of this construction in the forthcoming paper, but note that our clustering procedure provides a method to optimally partition the data such that minimum distortion hybrid automata may be constructed.

4 Experimental Results and Conclusions

We now apply the techniques presented in previous sections to build a simple model of the metabolism of peach fruit. We measured the expression profiles of two classes of genes, ARF and RAB, along a period of 42 days, starting 72 days after flowering and sampling the genes every week. Gene expressions profiles were collected using real time PCR [5, 10, 14]. In particular, we considered 13 and 20 genes for the ARF and RAB families, respectively. Each sample

consists in the average of 3 measurements normalized with respects to Ubiquitin Conjugating Enzyme level. We analyze the data appling the techniques described in Sections 2.1 and 2.2. A hierachical clustering based on the function $d(X,Y) = (1 - |Corr(X,Y)|)$ for ARF genes is reported in Figure 3. We choose as distance between two clusters, C_1 and C_2, the minimum distance between $X \in C_1$ and $Y \in C_2$. The label of the circle shaped nodes represent the distance between subgraphs. Requiring a correlation of at least 70% we obtained 3 and 5 gene clusters for ARF and RAB, respectively. Applying the clustering on the

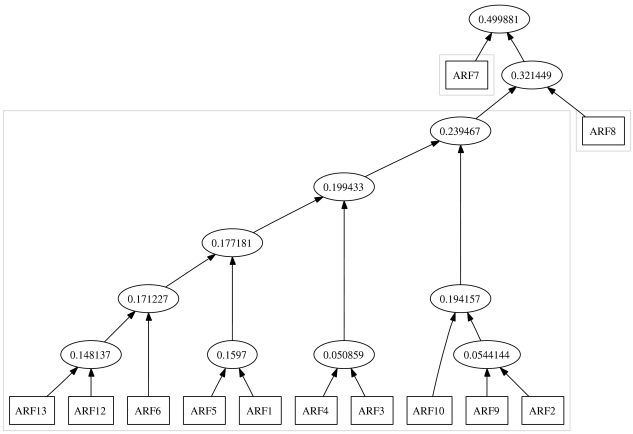

Fig. 3. The cluster hierachy of the ARF gene correlations.

date correlations we noticed a higher correlation: requiring a correlation of at least 98% we obtained 1 date cluster for ARF genes, while requiring a correlation of at least 93% for RAB genes, we got 2 date clusters. Hence, in the case of RAB genes we can built an automaton having just 2 discrete locations and the variables $\widehat{G}$, $\widehat{G}'$ and T which represent the evolution along 7 dates of 20 genes.

In conclusion, we emphasize that we have only established the preliminary foundations of a theory, aiming at the questions of how experimental data collected in biology may be treated rigorously within a semi-algebraic hybrid automata framework. We have hinted at its deep connection to dimensionality reduction and classical rate-distortion theory, but have relegated its complete treatment to the full paper. However, once such a framework has been created, it opens the field to many new questions. Namely, the following: How does one compare the dynamics of several closely related systems, e.g., a wild-type, mutant and a double-mutant? Can one factor the dynamics so that the final automaton may be viewed as product of several component automata, where most of the component modules remain unchanged over evolutionary time? How can the interaction between two or more biological systems (e.g., host-pathogen, host-vector-parasite, or an ecology) be modeled as products of hybrid automata constructed from different datasets?

References

1. R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho. Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems. In R. L. Grossman, A. Nerode, A. P. Ravn, and H. Richel, editors, *Hybrid Systems*, LNCS, pages 209–229. Springer, 1992.
2. H. Anai. Algebraic Approach to Analysis of Discrete-Time Polynomial Systems. In *European Control Conference (ECC'99)*, 1999.
3. Z. Bar-Joseph, G. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon. A New Approach to Analyzing Gene Expression Time Series Data. In *Proc. of Int. Conference on Computational biology (RECOMB'02)*, pages 39–48. ACM Press, 2002.
4. Ziv Bar-Joseph. Analyzing time series gene expression data. *Bioinformatics*, 20(16):2493–2503, 2004.
5. S. A. Bustin. Absolute quantification of mRNA using real-time reverse transcription polymerase chain reaction assays. *Journal of Mol. Endoc.*, 25:169–193, 2000.
6. A. Casagrande, V. Mysore, C. Piazza, and B. Mishra. Independent dynamics hybrid automata in systems biology. In *Proc. of the First International Conference on Algebraic Biology (AB'05)*, pages 61–73. Universal Academy Press, Inc., 2005.
7. A. Casagrande, C. Piazza, and B. Mishra. Semi-Algebraic Constant Reset Hybrid Automata - SACoRe. In *Proc. of Conference on Decision and Control (CDC'05)*, pages 678–683. IEEE Computer Society Press, 2005.
8. T. M. Cover and J. A. Thomas. *Elements of information theory.* Wiley-Interscience, New York, NY, USA, 1991.
9. S. Datta and S. Datta. Comparisons and validation of statistical clustering techniques for microarray gene expression data. *Bioinformatics*, 19(4):459–466, 2003.
10. C. Gachon, A. Mingam, and B. Charrier. Real-time PCR: what relevance to plant studies? *Journal of Experimental Botany*, 55(402):1445–1454, 2004.
11. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
12. R. Jekins and S. Pennington. Arrays for protein expression profiling: towards a viable alternative to two-dimensional gel electrophoresis? *Prot.*, 1(1):13–29, 2001.
13. I. T. Jolliffe. *Principal component analysis.* Series in statistics. Springer, 1986.
14. M. Kubista, J. M. Andrade, M. Bengtsson, A. Forootan, J. Jonák, K. Lind, R. Sindelka, R. Sjöback, B. Sjögreen, L. Strömbom, A. Ståhlberg, and N. Zoric. The real-time polymerase chain reaction. *Mol. Aspects of Medicine*, 27:95–125, 2006.
15. G. Lafferriere, G. J. Pappas, and S. Sastry. O-minimal Hybrid Systems. *Mathematics of Control, Signals, and Systems*, 13:1–21, 2000.
16. S. Madeira and A. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Trans. on Comp. Biology and Bioinformatics*, 1:24–45, 2004.
17. C. Piazza, M. Antoniotti, V. Mysore, A. Policriti, F. Winkler, and B. Mishra. Algorithmic algebraic model checking i: The case of biochemical systems and their reachability analysis. In K. Etessami and S. K. Rajamani, editors, *Proc. of Int. Conference on Computer Aided Verification (CAV'05)*, volume 3576 of *LNCS*. Springer, 2005.
18. M. Schena, D. Shalon, R.W. Davis, and Brown P.O. Quantitative monitoring of gene expression patterns with a complementary dna microarray. *Science*, 270(5235):467–70, 1995.
19. C. E. Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27:379–423, 1948.
20. N. Slonim, G. S. Atwal, G. Tkacik, and W. Bialek. Information-based clustering. *Proc Natl Acad Sci USA*, 2005.