2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Determining A Purely Symbolic Transfer Function from Symbol Streams: Theory and Algorithms

Christopher Griffin
Oak Ridge National Laboratory
P.O. Box 2008, MS6418
Oak Ridge, TN 37831
Email: cgriffin@psu.edu

Richard R. Brooks and Jason Schwier
Holcombe Department of Electrical and Computer Engineering
Clemson University
PO Box 340915, Clemson, SC 29634
Email: rrb@acm.org, jschwie@clemson.edu

***Abstract*— Transfer function modeling is a *standard technique* in classical Linear Time Invariant and Statistical Process Control. The work of Box and Jenkins was seminal in developing methods for identifying parameters associated with classical (r, s, k) transfer functions.**

Discrete event systems are often *used* for modeling hybrid control structures and high-level decision problems. *Examples include* discrete time, discrete strategy repeated games. For these games, a *discrete transfer function in the form of* an accurate hidden Markov model of input-output relations *could be used to derive optimal response strategies.*

In this paper, we develop an algorithm *for* creating probabilistic *Mealy machines* that act as transfer function models for discrete event dynamic systems (DEDS). Our models are defined by three parameters, (l_1, l_2, k) just as the Box-Jenkins transfer function models. Here l_1 is the maximal input history lengths to consider, l_2 is the maximal output history lengths to consider and k is the response lag. Using related results, We show that our Mealy machine transfer functions are optimal in the sense that they maximize the mutual information between the current known state of the DEDS and the next observed input/output pair[1].

I. Introduction

Transfer function modeling is critical in minimum mean square error (MMSE) control [1]. The work of Box and Jenkins was seminal [2] and has been extended and enhanced over the years by several authors.

Contrast this with the discrete event control literature. Here, plant models are often developed by hand. This may be reasonable in some cases but for real-world applications controllers need to be synthesized for complex systems that are not fully known a priori. In particular, it is difficult to be certain that manually created models accurately reflect plant dynamics. This is especially true when system transitions follow probability distributions. If we could automatically derive a plant model whose outputs *are* observed responses to a known set of inputs, the resulting model of input-output relationships would be a discrete event transfer function. *This transfer function could be used to synthesize a discrete event controller that could optimize some objective function defined in terms of the DEDS.*

[1]This work was sponsored by the Office of Naval Research, Contract N00014-06-C-0022. Portions of Dr. Griffin's work were performed as a Eugene P. Wigner Fellow and staff member at the Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U.S. Department of Energy under Contract DE-AC05-00OR22725.

In this paper, we show to extend Crutchfield and Shalizi's CSSR algorithm [3]–[5] to identify an optimal Mealy Machine representation, when three parameters are supplied: l_1, the maximal input history length; l_2, the maximal output history length; and k the delay.

The remainder of this paper is divided as follows: In Section II we provide a formal notation for developing our algorithm. In Section III we extend CSSR to derive Mealy machine transfer functions. In Section IV we provide a sample application of this work, by identifying an adversary's strategy in the repeated Prisoner's Dilemma Game and deriving an optimal control strategy.

II. Mathematical Notation

In this section we *provide* the notation and preliminaries necessary for the *proposed approach*. Our notation is *derived from* Box and Jenkins [2] and symbolic dynamical systems [6] using time series *expressed as a string of* symbols.

A. Time Series of Symbols

Let $\mathcal{A}$ (the *input* alphabet) and $\mathfrak{A}$ (the *output* alphabet) be finite sets of symbols. A symbolized time series is a sequence: $\mathbf{x} = \ldots x(-2)x(-1)x(0)x(1)x(2)\ldots$, where $x(t)$ represents the symbol that occurred at discrete time t in $\mathbf{x}$. If $x(t)$ is undefined, then we assume it is ϵ the empty symbol. Following the work of Box and Jenkins, let $\mathcal{B}$ be the backshift operator, so that $(\mathcal{B}\mathbf{x})(t) = x(t-1)$.

The set of all finite lengths strings of symbols from $\mathcal{A}$ and the concatenation $(+)$ operation form a monoid with unit ϵ. We may therefore write: $\mathbf{x} = \sum_{t=-\infty}^{\infty} x(t)$.

Consider a finite sub-sequence $t = 0$ to $t = s$ of $\mathbf{x}$, this may be written as: $(1 + \mathcal{B} + \ldots \mathcal{B}^s)x(s) = A(\mathcal{B})x(s)$. Let $A_1(\mathcal{B}) = \sum_{j=1}^{l_1} \mathcal{B}^j$ and $A_2(\mathcal{B}) = \sum_{j=1}^{l_2} \mathcal{B}^j$.

We assume the following system dynamics

$$x(t) = \alpha\left(A_1(\mathcal{B})x(t)\right) \tag{1}$$

$$y(t) = \gamma(\mathcal{B}^k A_1(\mathcal{B})x(t), A_2(\mathcal{B})y(t)) \tag{2}$$

where α and γ are random functions with range in $\mathcal{A}$ and $\mathfrak{A}$ respectively parametrized by recent observations subsequences of $\mathbf{x}$ and $\mathbf{y}$. That is, the process dynamics are

Markovian and we will derive probabilities:

$$\Pr\left[x(t) = a_i | A_1(\mathcal{B})x(t)\right] \quad (3)$$

$$\Pr\left[y(t) = b_j | A_1(\mathcal{B})x(t), A_2(\mathcal{B})y(t)\right] \quad (4)$$

where $\mathcal{A} = \{a_1, \ldots, a_n\}$ and $\mathfrak{A} = \{b_1, \ldots, b_m\}$.

B. Probabilistic Automata

A *labeled transition system* (LTS) is a tuple $G = \langle Q, \mathcal{A}, \delta \rangle$, where Q is a finite set of states, $\mathcal{A}$ is a finite alphabet and $\delta \subseteq Q \times \mathcal{A} \times Q$ is a transition relation. The transition relation δ is deterministic when for all $q \in Q$ and for all $x \in \mathcal{A}$ there is at most one $q' \in Q$ such that $(q, x, q') \in \delta$.

A *probabilistic LTS* is a pair $\langle G, p \rangle$ where G is an LTS and $p : \delta \rightarrow [0, 1]$ is a probability function such that $\sum_{x \in \mathcal{A}, q' \in Q} p(q, \sigma, q') = 1 \quad \forall q \in Q$.

It is easy to see that if there is some initial probability distribution π_0 over Q, then the triple $\langle G, p, \pi_0 \rangle$ is a Markov chain with transitions labels.

A *polygenic Mealy machine* is a tuple $M = \langle Q, \mathcal{A}, \mathfrak{A}, \delta \rangle$ where Q and $\mathcal{A}$ are as above and $\mathfrak{A}$ is a second *output* alphabet and $\delta \subseteq Q \times \mathcal{A} \times \mathfrak{A} \times Q$ is a transition relation with input alphabet $\mathcal{A}$ and output alphabet $\mathfrak{A}$. Determinism of the transition relation is defined just as it was for an LTS. If we assign a probability function to the transition relation, then we obtain a probabilistic Mealy machine. (Note for the sake of brevity, we remove the term polygenic.)

III. Input/Output Transfer Function Modeling

We turn our attention to the problem of modeling a Mealy machine given two observation sequences: $\mathbf{x}$, the input and $\mathbf{y}$, the output. We will assume that the input signal is generated randomly and not as a function of the observed output. If not, we will be modeling the coupled dynamics of an existing control system and we will not obtain a true representation of the impact of a control signal on system output. This is consistent with classical transfer function modeling.

Let $\mathbf{w}$ be a subsequence of $\mathbf{x}$ and let $\mathbf{z}$ be a subsequence of $\mathbf{y}$. By $\binom{\mathbf{w}}{\mathbf{z}}$ we mean the pair of input/output sequences. Assume that we know there is a lag of k time observation symbols before an output is observed. Then, for example if we began generating input symbols $x(1)x(2)\cdots$, and the lag is 2, then the output $y(1)$ will occur as symbol $x(3)$ is generated. From now on, we assume that the two sequences they are appropriately aligned, so that $y(1)$ corresponds to the input $x(1)$, though $y(1)$ appears at time $k + 1$.

We assume the following when modeling symbolic input/output dynamics: (i) There is precisely one output symbol for every input symbol, though the type of output may vary as a function defined in Equation 2 and (ii) Both input and output may be randomly determined. In fact, input should be randomly determined to prevent pre-existing control relationships from coloring the identified behavior.

Algorithm A will produce the symbolic transfer function; it is similar to the CSSR algorithm presented by Crutchfield and Shalizi [3]–[5], with one exception. We have modified the algorithm to operate on an input and output symbol stream to generate a probabilistic Mealy machine. The CSSR algorithm generates only probabilistic finite state machines.

A. Computing the Distribution Functions

The following formulas can be used to compute $f_{a\mathbf{w}|\mathbf{x}}$, $f_{\binom{a\mathbf{w}}{A\mathbf{z}}|\mathbf{x},\mathbf{y}}$, $f_{q_i|\mathbf{x}}$ and $\tilde{f}_{q_i|\mathbf{x},\mathbf{y}}$ found in the algorithm. Let $\#(\mathbf{x}, \mathbf{y})$ be the number of times the sequence $\mathbf{x}$ is observed as a subsequence of $\mathbf{y}$. Define: $\tilde{\#}(\mathbf{w}, \mathbf{x}, \mathbf{z}, \mathbf{y})$ be the number of times that $\mathbf{w}$ appears a substring of $\mathbf{x}$ and $\mathbf{z}$ appears as a substring of $\mathbf{y}$ *and* they appear at the same times; i.e., $\mathbf{w} = w(t)w(t+1)\cdots w(t+n)$ and $\mathbf{z} = z(t)z(t+1)\cdots z(t+n)$. In this case, since we defined $\mathbf{x}$ and $\mathbf{y}$ to be appropriately indexed so that the lag was removed, then we have:

$$f_{a\mathbf{w}|\mathbf{x}}(a) = \Pr\left(a|\mathbf{w}, \mathbf{x}\right) = \frac{\#(\mathbf{w}a, \mathbf{x})}{\#(\mathbf{w}, \mathbf{x})}$$

$$f_{\binom{a\mathbf{w}}{A\mathbf{z}}|\mathbf{x},\mathbf{y}}\binom{a}{A} = \Pr\left(\binom{a}{A}|\mathbf{w}, \mathbf{z}, \mathbf{x}, \mathbf{y}\right) = \frac{\tilde{\#}(\mathbf{w}a, \mathbf{x}, \mathbf{z}A, \mathbf{y})}{\tilde{\#}(\mathbf{w}, \mathbf{x}, \mathbf{z}, \mathbf{y})}$$

$$f_{q_i|\mathbf{x}}(a) = \Pr(a|q_i, \mathbf{x}) = \frac{\sum_{\binom{\mathbf{w}}{\mathbf{z}} \in q_i} \#(\mathbf{w}a, \mathbf{x})}{\sum_{\binom{\mathbf{w}}{\mathbf{z}} \in q_i} \#(\mathbf{w}, \mathbf{x})}$$

$$f_{q_i|\mathbf{x},\mathbf{y}}\binom{a}{A} = \Pr\left(\binom{a}{A}|\mathbf{x}, \mathbf{y}\right) = \frac{\sum_{\binom{\mathbf{w}}{\mathbf{z}} \in q_i} \tilde{\#}(\mathbf{w}a, \mathbf{x}, \mathbf{z}A, \mathbf{y})}{\sum_{\binom{\mathbf{w}}{\mathbf{z}} \in q_i} \tilde{\#}(\mathbf{w}, \mathbf{x}, \mathbf{z}, \mathbf{y})}$$

Algorithm A: Mealey CSSR Algorithm

Input: Input sequence $\mathbf{x}$; Output sequence $\mathbf{y}$; Input Alphabet $\mathcal{A}$; Output Alphabet $\mathfrak{A}$; Integer l_1; Integer l_2; Integer k;

Initialization:

1) Define state q_0 and add $\binom{\epsilon}{\epsilon}$ to state q_0. Set $Q = \{q_0\}$; $T = \emptyset$; $N := 1$.

Splitting

1) Set

$$W = \left\{ \binom{\mathbf{w}}{\mathbf{z}} | \exists q \in Q \left(\binom{w}{z} \in q \ \wedge \ \binom{w}{z} \notin T \wedge |\mathbf{w}| < l_1 \ \wedge |\mathbf{z}| < l_2 \right) \right\}$$

   Let N be the number of states.
2) For each $\binom{w}{z} \in W$, for each $a \in \mathcal{A} \cup \{\epsilon\}$ and $A \in \mathfrak{A} \cup \{\epsilon\}$, if $a\mathbf{w}$ is a subsequence of $\mathbf{x}$ and, $A\mathbf{z}$ is a subsequence of $\mathbf{y}$ then
   a) Determine $f_{a\mathbf{w}|\mathbf{x}} : \mathcal{A} \rightarrow [0, 1]$, the probability distribution over the next input symbol.
   b) Determine $f_{\binom{a\mathbf{w}}{A\mathbf{z}}|\mathbf{x},\mathbf{y}} : \mathfrak{A} \rightarrow [0, 1]$ using lag k.
   c) Let $f_{q_i|\mathbf{x}} : \mathcal{A} \rightarrow [0, 1]$ be the unified state conditional input probability distribution; that is, the probability given the system is in state q_i, that the next control symbol will be a. Let $\tilde{f}_{q_i|\mathbf{x},\mathbf{y}} : \mathfrak{A} \rightarrow [0, 1]$ be the unified state conditional probability output probability distribution; that is, the probability given the system is in state q_i, that the next output symbol will be A.
   d) For each i, compare $f_{q_i|\mathbf{x}}$ with $f_{a\mathbf{x}|\mathbf{x}}$ and $\tilde{f}_{q_i|\mathbf{x},\mathbf{y}}$ with $f_{\binom{a\mathbf{w}}{A\mathbf{z}}|\mathbf{x},\mathbf{y}}$ using an appropriate statistical test with confidence level α. Add $\binom{a\mathbf{w}}{A\mathbf{z}}$ to the state that has the most similar probability distribution as measured by the p-value of the test. If all tests reject the null

hypothesis, then create a new state q_{N+1} and add $\binom{a\mathbf{w}}{A\mathbf{z}}$ to it. Set $N := N + 1$.

3) Add each element of W to T. If $W \neq \emptyset$, Goto 1.

Reconstruction

1) Let $N_0 = 0$
2) Let N be the number of states.
3) Repeat while $N \neq N_0$:
   a) For each $i \in 1, \ldots, N$: Set $r := 0$. Let M be the number of elements in state q_i. Choose an element $\binom{\mathbf{w}_0}{\mathbf{z}_0}$ from state q_i. Create state p_{ir} and add w_0 to it. For all elements $\binom{\mathbf{w}_j}{\mathbf{z}_j}$ $(j > 0)$ in state q_i:
      i) For each $a \in \mathcal{A}$ and for each $A \in \mathfrak{A}$ $\binom{\mathbf{w}_j a}{\mathbf{z}_j A}$ produces an element that is resident in another state q_s. Let $\delta(\binom{\mathbf{w}_j}{\mathbf{z}_j}, \binom{a}{A}) = q_s$.
      ii) For $l = 0, \ldots, r$, choose $\binom{\mathbf{w}}{\mathbf{z}}$ from p_{ir}. if $\delta(\binom{\mathbf{w}_j}{\mathbf{z}_j}, \binom{a}{A}) = \delta(\binom{\mathbf{w}}{\mathbf{z}}, \binom{a}{A})$ for all $\binom{a}{A} \in \mathcal{A} \times \mathfrak{A}$, then add $\binom{\mathbf{w}_j}{\mathbf{z}_j}$ to p_{ir}. Otherwise, create a new state p_{ir+1} and add $\binom{\mathbf{w}_j}{\mathbf{z}_j}$ to it. Set $r := r + 1$.
   b) Set $N_0 = N$
   c) Let N be the number of states in the current model.
4) Recompute the state probabilities and assign transitions using the δ function defined above.

B. Properties of Algorithm A

Proposition 1: The probabilistic Mealy machine returned by Algorithm A is minimally stochastic. Further, the states produced by Algorithm A are sufficient statistics for the future symbols produces by the process.

Proof: Let $\mathbf{x}$ and $\mathbf{y}$ be the input and output symbol streams as before. Then the string interleaving $\mathbf{z} = \mathbf{x} \times \mathbf{y}$ producing $x(1)y(1)x(2)y(2)\cdots$ is a string in the alphabet $\mathcal{A} \cup \mathfrak{A}$. If the dynamics given in Equation 2 hold for this system, then any probabilistic finite state machine derived using the CSSR Algorithm with $\mathbf{z}$ will also uncover these dynamics *but* will require an input length $\max\{l_1, l_2\}$ to do so. Algorithm A applies the CSSR Algorithm [4], [5] to $\mathbf{z}$ in a more efficient way, by ignoring non-existent correlations such as the impact the outputs have on the succeeding inputs. Since Algorithm A improves the efficiency of the CSSR Algorithm [5] when the system dynamics are as given in Equation 2, it follows from Corollary 1 and Theorem 4 of [3] that the resulting Mealy machine is minimally stochastic and that the states are sufficient statistics for *both* the inputs and the outputs. ■

Remark 1.1: We do not prove it, but the computational complexity of Algorithm A is similar to that of CSSR, since it [Algorithm A] is a derivative algorithm. Hence it is linear in the lengths of input and output, but exponential in the size of the alphabets $\mathcal{A}$ and $\mathfrak{A}$.

IV. Example

The Prisoner's Dilemma Game is a well known symmetric bimatrix game describing the behavior of two captured criminals. There are two strategies for each player, Collude (C) and Defect (D). [7] has an excellent overview. In the repeated Prisoner's Dilemma Game, two players repeat this game in an attempt to maximize their long run pay-off.

In Tit-for-Two-Tats, a forgiving strategy is used to avoid unending defection. If Player 1 plays C in round i, then Player 2 will play C in round $i + 1$. If Player 1 plays D in round i and had previously played C in round $i - 1$, then Player 2 still plays C in Round $i + 1$. Otherwise, Player 2 plays D.

If Player 2 is using a fixed strategy $\mathcal{S}$ such as Tit-for-Two-Tats, it may be possible to *game* the strategy thus deriving a better long run payoff for Player 1. We can use Algorithm A to identify Player 2's strategy assuming Player 1 begins by playing a random strategy. The derived transfer function could then be used to obtain an optimal game controller.

We used Algorithm A to derive a behavioral model when Player 1 plays a random strategy and Player 2 plays Tit-for-Two-Tats. The resulting model is shown in Figure 1. We used 25 games for modeling. Two history streams (the input and output) were fed into the algorithm and we set $l_1 = 2$, $l_2 = 0$ and $k = 1$; these are the correct parameters for the Tit-for-Two-Tats strategy. When we supplied incorrect parameters

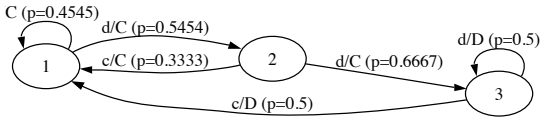

Fig. 1. A Mealy machine derived using $l_1 = 2$, $l_2 = 0$ and $k = 1$. (Note: In Step 2.d, we used the Kolmogorov-Smirnov test with p-value set of 0.05.)

$l_1 = 3, l_2 = 1$ and $k = 1$, the correct Mealy machine was still *found*. However, this was *not* the case when k was set to 2. In this case, the lag is incorrect and the outputs are not tied to the appropriate inputs. Clearly this indicates the necessity of identifying the lag correctly.

V. Future Work

It is worth noting that we have not provided any way of determining k, l_1 or l_2 from data. This is the subject of ongoing research and will be critical for exploring non-trivial control problems. We have already explored this problem for the CSSR algorithm [8] since [5] does not specify a method for determining the value of l_1 used there. We hope to apply some of the results we have obtained to this problem.

References

[1] E. D. Castillo, *Statistical Process Adjustment for Quality Control*. Wiley-Interscience, 2002.

[2] G. Box and G. Jenkins, *Time series analysis: Forecasting and control*. Holden-Day, 1970.

[3] C. R. Shalizi and J. P. Crutchfield, "Computational mechanics: Pattern and prediction, structure and simplicity," *J. Stat. Phys.*, vol. 104, no. 3/4, pp. 817–879, 2001.

[4] C. R. Shalizi, K. L. Shalizi, and J. P. Crutchfield, "Pattern discovery in time series, part i: Theory, algorithm, analysis, and convergence," Santa Fe Institute, Tech. Rep., 2002.

[5] C. Shalizi, K. Shalizi, and J. Crutchfield, "An algorithm for pattern discovery in time series," arXiv:cs.LG/0210025 v3, November 2002.

[6] M. Morse and G. Hedlund, "Symbolic dynamics," *American Journal of Mathematics*, vol. 60, no. 4, pp. 815–866, 1938.

[7] Wikipedia, "Prisoner's dilemma," http://en.wikipedia.org/wiki/Prisoner%27s_dilemma, August 30 2007.

[8] R. R. Brooks, J. Schwier, C. Griffin, and S. Bukkapatnam, "Zero knowledge hidden markov model inference," *submitted to IEEE Trans. Sys. Man and Cyber. A*, 2008.