Proceedings of the 2011 IEEE International Conference on
Intelligent Computing and Intelligent Systems (ICIS 2011),
Guangzhou, China, Nov. 18-20, 2011

On the Control of Indeterminate Asynchronous Sequential Machines: An Algebraic Framework

Jung-Min Yang
Department of Electrical Engineering
Catholic University of Daegu
330 Kumrak, Hayang, Gyeongsan
Gyeongbuk 712-702, Republic of Korea

Tan Xing
Department of Electrical
and Computer Engineering
University of Florida
Gainesville, FL 32611-6130, USA

Jacob Hammer
Department of Electrical
and Computer Engineering
University of Florida
Gainesville, FL 32611-6130, USA

***Abstract*—An algebraic framework for controlling asynchronous sequential machines with unknown or unpredictable transitions is introduced. A semiring is used to characterize all compound transitions that have a predictable outcome. This semiring plays a critical role in model matching and in adaptive feedback control of indeterminate asynchronous machines.**

I. Introduction

Asynchronous sequential machines, or clockless logic circuits, serve important roles in high speed computing, in parallel computing, in the modeling of signaling chains in molecular biology, and in other applications. Often, the description of an asynchronous machine is not fully known: the machine may not have been tested exhaustively, or its response may be affected by interferences, malfunctions, or errors.

An *indeterminate transition* of an asynchronous machine is a transition whose outcome is not known a-priori; a *determinate* transition has a known outcome. A machine with indeterminate transitions is an *indeterminate machine.*

Recall that an asynchronous machine has two kinds of states: stable states (in which the machine lingers until an input change occurs) and transient states (through which the machine passes very quickly). A transition between stable states is a *stable transition*. A *compound transition* is a string of transitions between stable states. A compound transition may be determinate even if it includes indeterminate segments. By characterizing all determinate compound transitions of a machine, we find all tasks the machine can perform determinately. In this note, we present an algebraic framework that characterizes all determinate compound transitions through simple algebraic operations (section III).

In the control configuration of Figure 1,

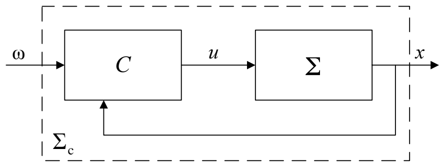

Fig. 1. Closed loop configuration

Σ is an indeterminate asynchronous machine to be controlled, while C is another asynchronous machine that serves as a controller. Our objective is to design C so that the closed loop machine Σ_c is a determinate machine. One important application is model matching, where Σ_c is required to emulate a specified determinate model Σ'. By 'emulate' we mean that Σ_c and Σ' have the same stable transitions and, hence, are indistinguishable to a user. We write $\Sigma_c = \Sigma'$ when Σ_c emulates Σ'. Using our algebraic framework, we derive in section V simple necessary and sufficient conditions for the existence of a controller C that achieves model matching.

Fundamental mode operation is an operating policy that prohibits simultaneous change of two or more variables in an asynchronous machine. This helps prevent uncertainties since, in an asynchronous environment, simultaneous change manifests itself as sequential change in unpredictable order and may lead to unpredictable outcomes.

Condition 1. The closed loop machine Σ_c of Figure 1 operates in fundamental mode when the following are valid:
(i) Σ is in a stable state while C is in transition.
(ii) C is in a stable state while Σ is in transition.
(iii) The external input ω changes only while Σ and C are both in stable states. □

Parts *(i)* and *(ii)* of Condition 1 are restrictions on the design of the controller C; part *(iii)*, on the other hand, is a restriction on the operation of the closed loop machine. As transitions of asynchronous machines occur very quickly, *(iii)* is not a burdensome requirement.

The paper is written within the framework of [7], [8], and [12], where the control of asynchronous sequential machines is considered. Studies dealing with other aspects of the control of sequential machines can be found in [9], [11], and [6], where the theory of discrete event systems is investigated; in [3], [4], [5], [2], [1], and [13], where issues related to control and model matching of sequential machines are studied; and in many other publications.

The paper is organized as follows. Section II introduces basic features of indeterminate asynchronous machines, while sections III and IV introduce an algebraic framework that helps characterize all determinate compound transitions of an indeterminate machine. The framework is applied in section V to the solution of the model matching problem. A comprehensive example runs through all sections of the paper.

978-1-61284-144-1/11/$26.00 ©2011IEEE

II. Indeterminate Machines

A. Basics

An input/state asynchronous machine is represented by a quadruple $\Sigma = (A, X, x^0, f)$, where A is the input alphabet, X is the set of states, x^0 is the initial state, and $f : X \times A \to X$ is a partial function called the recursion function. The machine operates according to

$$x_{k+1} = f(x_k, u_k), \qquad k = 0, 1, 2, \ldots,$$

where $x_0 := x^0$ is the initial state, $u_0 u_1 u_2 \ldots$ is the input sequence, and $x_1 x_2 \ldots$ is the sequence of states generated by the machine. The step counter k advances by one upon a change of input or state. A pair $(x, u) \in X \times A$ is *valid* if it is in the domain of f.

Often, values of the recursion function f are not precisely known due to malfunctions or errors, or due to incomplete characterization of the machine. In such cases, there may be several options for the machine's next state, and these are specified as a set of potential next states by regarding f as a set valued function. The set $f(x, u)$ consists then of all next state options of Σ.

Definition 2. If $f(x, u)$ includes more than one element, then (x, u) is an *indeterminate pair*; it induces an *indeterminate transition*. □

A valid pair (x, u) is a *stable combination* if $f(x, u) = x$; otherwise, (x, u) is a *transient combination*. At a stable combination, a machine lingers until an input change occurs; the machine passes quickly through transient combinations (ideally, in zero time). A transition from one stable combination to another may involve a chain of several transient combinations. Specifically, assume that Σ is at a stable combination (x, u'), when the input character changes to u. This may result in a chain of transitions $x_1 := f(x, u)$, $x_2 := f(x_1, u)$, ... If this chain terminates, then there is an integer $i \geq 1$ for which $x_i = f(x_i, u)$, and x_i is the *next stable state*. If the chain does not terminate, then Σ has an infinite cycle.

Convention 3. *Only machines without infinite cycles are considered in this note.* □

Thus, for machines considered here, every valid combination has a next stable state. The *stable recursion function s* of Σ is defined at every valid pair (x, u) by $s(x, u) := x'$, where x' is the next stable state (or the set of potential next stable states) of x with the input u. The stable recursion function describes users' experience with Σ, since transient states are traversed very quickly. Notwithstanding, transients do play an important role in the control of asynchronous machines, since the controller C of Figure 1 works by turning undesirable stable combinations of Σ into transient combinations of the closed loop machine Σ_c. This dismisses undesirable features of Σ (see [8]). For a string of input characters $u := u_0 u_1 \cdots u_q$, the shorthand notation $s(x, u) := s(s(\ldots s(s(x, u_0), u_1) \ldots), u_q)$ denotes the final stable state that Σ reaches when the input string u is applied starting at the state x. To preserve fundamental mode operation, the string u must be applied character-by-character, waiting after each character for Σ to reach its next stable state before applying another character.

B. The Adjusted Machine

Consider an indeterminate input/state asynchronous machine $\Sigma = (A, X, x^0, f)$ with the stable recursion function s, and assume that the indeterminate pairs of s form the set

$$U := \{(z^1, u^1), (z^2, u^2), \ldots, (z^r, u^r)\} \subseteq X \times A. \tag{1}$$

The set of potential next stable states $Z^i := s(z^i, u^i)$ of the pair (z^i, u^i) has $n(i) > 1$ members called *potential outcomes*:

$$Z^i := \{z^{i,1}, \ldots, z^{i,n(i)}\} \subseteq X. \tag{2}$$

The indeterminacy about the outcome of an indeterminate pair (z^i, u^i) can be formally resolved by associating a unique pseudo input character with every potential outcome, as follows. Let B be a sufficiently large alphabet disjoint from the input alphabet A. For each indeterminate pair (z^i, u^i), choose a distinct set A^i of $n(i)$ characters of B:

$$\begin{array}{l} A^i := \{u^{i,1}, u^{i,2}, \ldots, u^{i,n(i)}\} \subseteq B, \text{ where} \\ A^i \cap A^j = \varnothing \text{ for all } i \neq j \in \{1, \ldots, r\}. \end{array} \tag{3}$$

The *pseudo alphabet* is then the set $A' := \cup_{i=1,\ldots r} A^i$, and the *extended input alphabet* is $\tilde{A} := A \cup A'$. Now, we construct from Σ the following determinate machine.

Definition 4. The *adjusted stable recursion function* $s_a : X \times \tilde{A} \to X$ of Σ is

$$s_a(x,u) := \begin{cases} s(x,u) & \text{if } (x,u) \text{ is a determinate pair of } \Sigma; \\ z^{i,j} & \text{if } (x,u) = (z^i, u^{i,j}) \text{ for some} \\ & i \in \{1,\ldots,r\}, j \in \{1,2,\ldots,n(i)\}; \\ z^{i,j} & \text{if } (x,u) = (z^{i,j}, u^{i,j}) \text{ for some} \\ & i \in \{1,\ldots,r\}, j \in \{1,2,\ldots,n(i)\}; \\ \text{invalid} & \text{for all other } (x,u) \in X \times \tilde{A}, \\ & \text{including all } (x,u) \in U. \end{cases} \tag{4}$$

Then, $\Sigma_a := (\tilde{A}, X, x^0, s_a)$ is the *adjusted machine*. □

In (4), the first line makes Σ_a identical to Σ at determinate pairs of Σ; the second line formally resolves indeterminacy of each indeterminate pair (z^i, u^i) by associating a distinct pseudo input character $u^{i,j} \in \tilde{A}$ with every potential outcome $z^{i,j}$; the third line makes each pair $(z^{i,j}, u^{i,j})$ into a stable combination of s_a; and the fourth line turns all members of the set U of indeterminate pairs into invalid pairs of s_a. Then, the resulting adjusted machine Σ_a has no indeterminate pairs.

Example 5. Consider an asynchronous machine Σ with the input alphabet $A = \{a, b, c, d\}$, the state set $X = \{x^1, x^2, x^3, x^4\}$, the initial state $x^0 = x^1$, and the stable recursion function s given in the following table, where "–" denotes an invalid pair.

state	a	b	c	d
x^1	x^2	x^1	x^1	$-$
x^2	x^2	$-$	$\{x^1,x^4\}$	x^3
x^3	$\{x^2,x^4\}$	$-$	$-$	x^3
x^4	x^4	x^1	x^4	x^4

The indeterminate pairs are $U = \{(x^2,c),(x^3,a)\}$. Correspondingly, we introduce the pseudo character sets $A^1 := \{c^1,c^2\}$ and $A^2 := \{a^1,a^2\}$. These yield the pseudo input alphabet $A' = A^1 \cup A^2 = \{a^1,a^2,c^1,c^2\}$, and the extended input alphabet $\tilde{A} = \{a,b,c,d,a^1,a^2,c^1,c^2\}$. By (4), the adjusted stable recursion function s_a is then given by the following table.

state	a	b	c	d	a^1	a^2	c^1	c^2
x^1	x^2	x^1	x^1	$-$	$-$	$-$	x^1	$-$
x^2	x^2	$-$	$-$	x^3	x^2	$-$	x^1	x^4
x^3	$-$	$-$	$-$	x^3	x^2	x^4	$-$	$-$
x^4	x^4	x^1	x^4	x^4	$-$	x^4	$-$	x^4

The adjusted machine Σ_a is a determinate 'synthetic' machine. Of course, members of the pseudo character set A' cannot be applied as inputs to the real machine Σ, but they do serve a critical role in characterizing compound determinate transitions of Σ, as we discuss next.

III. A Semiring of Strings and Complete Sets

A. Basic Operations and Complete Sets

To work with the extended alphabet $\tilde{A}$, we introduce a semiring $\mathfrak{A}$ over the set $(\tilde{A})^*$ of all strings of characters of $\tilde{A}$. In this semiring, concatenation serves as multiplication and union serves as addition: the product of two strings $a,b \in (\tilde{A})^*$ is the concatenation ab, where a is the prefix; the sum of two subsets $c,d \subseteq (\tilde{A})^*$ is their union $c+d := c \cup d$. These operations combine with distributive laws

$$a(b+c) = ab+ac,$$
$$(a+b)c = ac+bc.$$

To examine the utility of the semiring $\mathfrak{A}$, consider an indeterminate pair (z^i,u^i) of an asynchronous machine Σ with the potential outcomes $\{z^{i,1},...,z^{i,n(i)}\}$, and let $A^i = \{u^{i,1},...,u^{i,n(i)}\}$ be the corresponding set of pseudo input characters. Then, by (4), the adjusted stable recursion function satisfies $z^{i,j} = s_a(z^i,u^{i,j})$ and $z^{i,j} = s_a(z^{i,j},u^{i,j})$, $j = 1,...,n(i)$, $i = 1,...,r$. Now, assume that Σ has a state z' with the following feature: for each one of the outcomes $z^{i,j}$, there is an input string $\alpha^j \in A^*$ (with no pseudo characters) for which $s(z^{i,j},\alpha^j) = z'$, $j = 1,...,n(i)$. Then, irrespective of which one of the states $z^{i,1},...,z^{i,n(i)}$ is the actual outcome of (z^i,u^i), we can always reach the state z' by using a state feedback controller: upon detecting the outcome $z^{i,j}$, the controller applies to Σ the input string α^j. This takes Σ to z' for every outcome of the indeterminate transition. More generally, every set of strings

$$\gamma^1(i) := \{\ u^{i,1}\alpha^1 + u^{i,2}\alpha^2 + \cdots + u^{i,n(i)}\alpha^{n(i)}\ \}, \qquad (5)$$

where $\alpha^1,\alpha^2,...,\alpha^{n(i)} \in A^*$ and $s_a(z^i,u^{i,1}\alpha^1) = s_a(z^i,u^{i,2}\alpha^2) = \cdots = s_a(z^i,u^{i,n(i)}\alpha^{n(i)})$, gives rise to a determinate stable transition from the indeterminate pair (z^i,u^i) to a common final stable state. The overall transition is determinate, despite the inclusion of indeterminate segments.

Next, assume that Σ was taken from a determinate pair (z,u) to the indeterminate pair (z^i,u^i) by an input string $\alpha \in A^*$. Then, using α as a prefix, we obtain the set of strings

$$\Gamma^1(i) := \{\alpha\gamma^1(i)\} \qquad (6)$$

which has two critical features:

(a) It includes a response to every outcome of the indeterminate pair (z^i,u^i); and

(b) All these responses take Σ to the same stable state.

Assuming that Σ has r indeterminate pairs $\{(z^1,u^1),...,(z^r,u^r)\}$, we define the family of *complete sets of order* 1 of Σ

$$\Gamma^1(\Sigma) := \cup_{i=1,...,r}\Gamma^1(i).$$

A slight reflection shows that features (a) and (b) remain valid when the strings $\alpha^1,\alpha^2,...,\alpha^{n(i)}$ of (5) include complete sets of order 1, i.e., when $\alpha,\alpha^1,\alpha^2,...,\alpha^{n(i)} \in (A \cup \Gamma^1(\Sigma))^*$. Doing so creates the family $\Gamma^2(i)$ of complete sets of order 2 associated with the indeterminate pair (z^i,u^i). The entire family of complete sets of order 2 of Σ is then

$$\Gamma^2(\Sigma) := \cup_{i=1,...,r}\Gamma^2(i).$$

In general, assuming that the family $\Gamma^p(\Sigma)$ of complete sets of order p of Σ has been created for an integer $p \geq 1$, define the set $\gamma^{p+1}(i)$ by

$$\gamma^{p+1}(i) := \{\ u^{i,1}\alpha^1 + u^{i,2}\alpha^2 + \cdots + u^{i,n(i)}\alpha^{n(i)}\ \}\ , \qquad (7)$$

where $\alpha^1,\alpha^2,...,\alpha^{n(i)} \in [A \cup \Gamma^p(\Sigma)]^*$ and $s_a(z^i,u^{i,1}\alpha^1) = s_a(z^i,u^{i,2}\alpha^2) = \cdots = s_a(z^i,u^{i,n(i)}\alpha^{n(i)})$. Then, the family $\Gamma^{p+1}(i)$ of all complete sets of order $p+1$ for the indeterminate pair (z^i,u^i) is

$$\Gamma^{p+1}(i) := \{\alpha\gamma^{p+1}(i)\}, \qquad (8)$$

where $\alpha \in [A \cup \Gamma^p(\Sigma)]^*$ takes Σ from a determinate pair to the indeterminate pair (z^i,u^i) and $\gamma^{p+1}(i)$ is given by (7). Finally, the family of all complete sets of order $p+1$ of Σ is

$$\Gamma^{p+1}(\Sigma) := \cup_{i=1,2,...r}\Gamma^{p+1}(i).$$

We use the notation $\Gamma^0(\Sigma) := \{\alpha \in A^*\}$, where α takes Σ from a determinate pair to a determinate pair.

Definition 6. A *complete set* of Σ is any member of the family $\Gamma(\Sigma) := \cup_{p=0,1,2,...}\Gamma^p(\Sigma)$. $\square$

A complete set includes a response to every outcome of every indeterminate transition encountered along its way, and all these responses lead to the same stable state. Complete sets are intimately related to state feedback, as follows.

Theorem 7. *Let Σ be an indeterminate machine with the adjusted machine $\Sigma_a = (\tilde{A},X,x^0,s_a)$, and let x and x' be two states of Σ. Then, the following are equivalent:*
(i) There is a state feedback controller C that takes Σ through a determinate transition from x to x' in fundamental mode.
(ii) There is a complete set γ satisfying $s_a(x,\gamma) = x'$.

Proof: (sketch) If *(i)* is valid, then C generates a response for every outcome of every indeterminate transition encountered along the way from x to x'; the cumulation of these responses forms a complete set. Conversely, if *(ii)* is valid, the complete set γ can be used to build a state feedback controller C that induces a determinate transition from x to x', as follows. At each indeterminate pair (z^i, u^i) encountered, γ prescribes the next input character $w^{i,j}$ that C must apply to Σ upon detecting the outcome $z^{i,j}$ of (z^i, u^i). ■

It can be shown that, for an asynchronous machine with n states, complete sets can always be shortened to a length not exceeding $n-1$ characters (see [12]).

B. Reduced Sets of Input Sequences

Let $S \subseteq (\tilde{A})^*$ be the set of all strings that take the adjusted machine Σ_a from a stable combination with a state x to a stable combination with a state x'. If S includes a complete set γ, then, by Theorem 7, there is a state feedback controller that implements a determinate transition from x to x'. Clearly, the presence or absence of additional members of S outside of γ is irrelevant in this regard. Thus, to simplify calculations, we ignore members of S outside of γ and augment the definition of addition in the semiring $\mathfrak{A}$ by the property

$$\gamma + \alpha = \gamma \text{ for any complete set } \gamma \text{ and any set } \alpha \subseteq (\tilde{A})^*. \quad (9)$$

Further, it can be verified that the following is valid.

Proposition 8. *Let x, x', and x'' be states, and let γ and γ' be complete sets, where γ takes Σ_a from a stable combination with x to a stable combination with x', while γ' takes Σ_a from a stable combination with x' to a stable combination with x''. Then, $\gamma\gamma'$ is a complete set taking Σ_a from a stable combination with x to a stable combination with x''.* □

Proposition 8 and (9) help us simplify expressions without compromising information about the existence of state feedback controllers that induce determinate transitions.

Example 9. According to Example 5, the set of input strings taking Σ_a from x^2 to x^1 is $S = \{c^1 + c^2b + da^2b + \cdots\}$. As $c^1 + c^2b$ is a complete set of strings by (7), we can use (9) to write $S = \{c^1 + c^2b\}$, a significant simplification. □

Definition 10. A set of input strings $S \subseteq (\tilde{A})^+$ is *reducible* if it can be simplified into a complete set by using (9) and Proposition 8; otherwise, S is *irreducible*. A reducible set is in *reduced form* when it is expressed as a complete set. □

In view of Theorem 7, reducible sets represent transitions that can be implemented in determinate form by a state feedback controller. Therefore, the problem of determining the existence of such state feedback controllers can be resolved through simple algebraic manipulations within the ring $\mathfrak{A}$.

IV. Stable Reachability

A. Determinate and Indeterminate Transitions

As seen in [8], the matrix of stable transitions plays an important role in the solution of control problems. To extend its definition to indeterminate machines, consider an indeterminate machine $\Sigma = (A, X, x^0, f)$ with the state set $X = \{x^1, \ldots, x^n\}$ and the adjusted machine $\Sigma_a = (\tilde{A}, X, x^0, s_a)$. Denote by $(\tilde{A})^{(i)}$ the set of all strings of i or fewer characters of the extended alphabet $\tilde{A}$. For two integers $p, q \in \{1, ..., n\}$ and an integer $i > 0$, define the set of strings

$$\alpha^i(p,q) := \{u \in (\tilde{A})^{(i)} \mid s_a(x^p, u) = x^q\},$$

and let N be a character not in the alphabet $\tilde{A}$.

Definition 11. The *adjusted matrix of stable transitions $R^a(\Sigma)$* has the entries

$$R^a_{pq}(\Sigma) := \begin{cases} \alpha^{n-1}(p,q) & \text{if } \alpha^{n-1}(p,q) \neq \varnothing, \\ N & \text{else;} \end{cases}$$

the *one-step adjusted matrix of stable transitions $R^a(\Sigma, 1)$* has the entries

$$R^a_{pq}(\Sigma, 1) := \begin{cases} \alpha^1(p,q) & \text{if } \alpha^1(p,q) \neq \varnothing, \\ N & \text{else,} \end{cases}$$

$p, q = 1, 2, ..., n$. □

We extend the semiring $\mathfrak{A}$ to include the character N:

$$N\alpha = \alpha N = N \text{ for all } \alpha \in (\tilde{A})^+,$$
$$N + \alpha = \alpha + N = \alpha \text{ for all } \alpha \in (\tilde{A})^+.$$

Then, with the operations in $\mathfrak{A}$, we can use the usual definition of matrix multiplication to obtain the powers $(R^a(\Sigma, 1))^i, i = 1, 2, ...$ and construct the combination

$$(R^a(\Sigma))^{(i)} := R^a(\Sigma, 1) + (R^a(\Sigma, 1))^2 + \cdots + (R^a(\Sigma, 1))^i.$$

It can be seen that

$$R^a(\Sigma) = (R^a(\Sigma, 1))^{(n-1)}. \quad (10)$$

Example 12. From Example 5, we have $R^a(\Sigma, 1) =$

$$\begin{pmatrix} \{b + c + c^1\} & \{a\} & N & N \\ \{c^1\} & \{a + a^1\} & \{d\} & \{c^2\} \\ N & \{a^1\} & \{d\} & \{a^2\} \\ \{b\} & N & N & \left\{ \begin{array}{c} a + c + d \\ +a^2 + c^2 \end{array} \right\} \end{pmatrix}.$$

Then, $R^a(\Sigma)$ can be obtained by (10). □

A statement analogous to the following is proved in [8].

Proposition 13. *Let Σ_a be an adjusted asynchronous machine with the state set $X = \{x^1, ..., x^n\}$, the extended input alphabet $\tilde{A}$, and the adjusted matrix of stable transitions $R^a(\Sigma)$. Then, the following two statements are equivalent.*

(i) There is an input string $u \in (\tilde{A})^+$ that takes Σ_a from a stable combination with the state x^i to a stable combination with the state x^j in fundamental mode operation.

(ii) $R^a_{ij}(\Sigma) \neq N$. □

The adjusted matrix of stable transitions can be simplified:

Definition 14. The *reduced matrix of stable transitions $R(\Sigma)$* is obtained by expressing every reducible entry of $R^a(\Sigma)$ in reduced form; irreducible entries are left unchanged. □

By Proposition 13, the reduced matrix of stable transitions characterizes all determinate transitions that can be implemented by state feedback (see [12] for details):

Corollary 15. *Let Σ be an asynchronous machine with the state set $X = \{x^1, x^2, ..., x^n\}$ and the reduced matrix of stable transitions $R(\Sigma)$. Then, the following are equivalent for any states $x^i, x^j \in X$.*
(i) There is a state feedback controller that takes Σ through a determinate transition from a stable combination with x^i to a stable combination with x^j in fundamental mode operation.
(ii) $R_{ij}(\Sigma)$ is a complete set. □

Example 16. Using Example 12, the reduced matrix of stable transitions is

$$R(\Sigma) = \begin{pmatrix} \{b\} & \{a\} & \{ad\} & R_{14}(\Sigma) \\ \{\gamma_1\} & \{a\} & \{d\} & R_{24}(\Sigma) \\ \{\gamma_2\} & \{\gamma_3\} & \{d\} & R_{34}(\Sigma) \\ \{b\} & \{ba\} & \{bad\} & \{a\} \end{pmatrix},$$

where $\gamma_1 := c^1 + c^2 b$, $\gamma_2 = a^1\gamma_1 + a^2 b$, and $\gamma_3 = a^1 + a^2 ba$ are complete sets; $R_{14}(\Sigma), R_{24}(\Sigma)$, and $R_{34}(\Sigma)$ are irreducible. □

The information in the reduced matrix of stable transitions can be further condensed as follows (compare to [8]).

Definition 17. Let $\Sigma = (A, X, x^0, f)$ be an asynchronous machine with n states and the reduced matrix of stable transitions $R(\Sigma)$. Let Δ be a character not in $\tilde{A} \cup \{N\}$. Then, the *skeleton matrix* $K(\Sigma)$ is an $n \times n$ matrix with the entries $(i, j = 1, 2, ..., n)$

$$K_{ij}(\Sigma) := \begin{cases} 1 & \text{if } R_{ij}(\Sigma) \text{ is a complete set,} \\ 0 & \text{if } R_{ij}(\Sigma) = N, \\ \Delta & \text{if } R_{ij}(\Sigma) \text{ is irreducible.} \end{cases}$$

In $K(\Sigma)$, transitions indicated by 1 can be implemented in determinate form by a state feedback controller operating in fundamental mode; transitions indicated by 0 are impossible; and transitions indicated by Δ are indeterminate – they may or may not be possible, depending on the outcomes of indeterminate transitions along the way.

Example 18. Based on Example 16, the skeleton matrix is

$$K(\Sigma) = \begin{pmatrix} 1 & 1 & 1 & \Delta \\ 1 & 1 & 1 & \Delta \\ 1 & 1 & 1 & \Delta \\ 1 & 1 & 1 & 1 \end{pmatrix}. \square$$

V. Model Matching

Consider a machine $\Sigma = (A, X, x^0, f)$ connected to a state feedback controller C as in Figure 1, and let $\Sigma' = (A, X, x^0, s')$ be a specified determinate model. Let $K(\Sigma)$ and $K(\Sigma')$ be the skeleton matrices of Σ and Σ', respectively. When Σ has no indeterminate transitions, [8, Theorem 5.1] states that a state feedback controller C satisfying $\Sigma_c = \Sigma'$ exists if and only if

$$K(\Sigma) \geq K(\Sigma'). \tag{11}$$

When Σ has indeterminate transitions, $K(\Sigma)$ may include entries of the character Δ, which indicate unpredictable outcomes. When an entry of Δ in $K(\Sigma)$ appears opposite an entry of 1 in $K(\Sigma')$, model matching cannot be guaranteed. On the other hand, an entry of Δ in $K(\Sigma)$ opposite an entry of 0 in $K(\Sigma')$ has no direct bearing on model matching, since the transition it represents is not required to match Σ'. These considerations can be incorporated into (11) by defining Δ as a number satisfying

$$0 < \Delta < 1. \tag{12}$$

With the assignment (12), the following is true.

Theorem 19. *Let $\Sigma = (A, X, x^0, f)$ be an asynchronous machine with the skeleton matrix $K(\Sigma)$, and let $\Sigma' = (A, X, x^0, s')$ be a determinate stable state machine with the skeleton matrix $K(\Sigma')$. Then, the following are equivalent:*
(i) There is a state feedback controller C satisfying $\Sigma_c = \Sigma'$, where Σ_c operates in fundamental mode.
(ii) $K(\Sigma) \geq K(\Sigma')$. □

Theorem 19*(ii)* provides a convenient way to determine whether model matching is possible.

Acknowledgement

The work of J.-M. Yang was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0005116).

References

[1] G. BARRETT and S. LAFORTUNE [1998], "Bisimulation, the supervisory control problem, and strong model matching for finite state machines," *Discrete Event Dynamic Systems: Theory and Application*, vol. 8, no. 4, pp. 377–429.

[2] M. D. DIBENEDETTO, A. SALDANHA and A. SANGIOVANNI-VINCENTELLI [1994], "Model matching for finite state machines," *Proc. IEEE Conf. on Decision and Control*, 1994, pp. 3117–3124.

[3] J. HAMMER [1994], "On some control problems in molecular biology," *Proc. IEEE Conf. on Decision and Control*, pp. 4098–4103, Dec. 1994.

[4] J. HAMMER [1995], "On the modeling and control of biological signal chains," *Proc. IEEE Conf. on Decision and Control*, pp. 3747–3752, Dec. 1995.

[5] J. HAMMER [1996], "On the control of incompletely described sequential machines," *Int. J. Control*, vol. 63, no. 6, pp. 1005–1028.

[6] R. KUMAR, S. NELVAGAL, and S. I. MARCUS [1997], "A discrete event systems approach for protocol conversion," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 7, no. 3, pp. 295–315.

[7] T. E. MURPHY, X. J. GENG, and J. HAMMER [2002], "Controlling races in asynchronous sequential machines," *Proceeding of the IFAC World Congress*, Barcelona, July 2002.

[8] T. E. MURPHY, X. GENG, and J. HAMMER [2003], "On the control of asynchronous machines with races," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 1073–1081, 2003.

[9] P. J. G. RAMADGE and W. M. WONHAM [1987], "Supervisory control of a class of discrete event processes," *SIAM Journal on Control and Optimization*, vol. 25, no. 1, pp. 206–230.

[10] M.-D. SHIEH, C.-L. WEY, and P. D. FISHER [1993], "Fault effects in asynchronous sequential logic circuits," *IEE Proc.-E*, vol. 140, no. 6, pp. 327–332, 1993.

[11] J. G. THISTLE and W. M. WONHAM [1994], "Control of infinite behavior of finite automata," *SIAM J. Control and Opt.*, vol. 32, no. 4, pp. 1075–1097.

[12] J.-M. YANG, T. XING, and J. HAMMER [2011], "Adaptive control of asynchronous sequential machines with state output," submitted.

[13] N. YEVTUSHENKO, T. VILLA, R. BRAYTON, A. PETRENKO, and A. SANGIOVANNI-VINCENTELLI [2008], "Compositionally progressive solutions of synchronous FSM equations," *Discrete Event Dynamic Systems: Theory and Application*, vol. 18, no. 1, pp. 51–89.