Proceedings of the 17th International Symposium on
Mathematical Theory of Networks and Systems, Kyoto, Japan, 2006

Controllers for Asynchronous Sequential Machines with Infinite Cycles

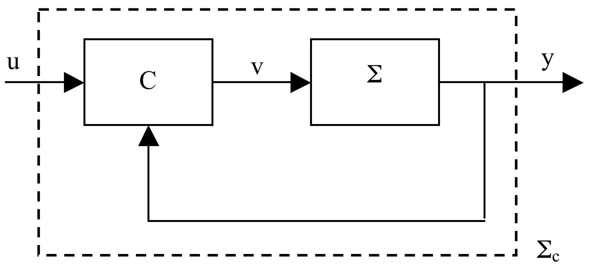
Niranjan Venkatraman and Jacob Hammer

Abstract—The existence of state-feedback controllers that eliminate the effects of infinite cycles on asynchronous sequential machines is considered. The main objective is to develop controllers that stop infinite cycles in an existing machine, while controlling the machine to match a prescribed model. Necessary and sufficient conditions for the existence of such controllers are stated in terms of a simple test applied to a certain numerical matrix.

I. Introduction

ASYNCHRONOUS sequential machines are common building blocks of high-speed computing equipment. Asynchronous machines can be afflicted by infinite cycles, which cause a machine to loop indefinitely among several of its states. Infinite cycles are caused by malfunctions, design flaws, component failures, or implementation flaws. This note presents state-feedback controllers that overcome the effects of infinite cycles on existing asynchronous machines. These controllers achieve two objectives: (i) the controlled machine does not linger in an infinite cycle; and (ii) the performance of the controlled machine matches a desired model.

(1)



Here, Σ is the faulty asynchronous machine being controlled, and C is an asynchronous machine serving as a controller. The closed loop machine is denoted by Σ_c. The controller can be designed so that the closed loop system will function properly before, as well as after, an infinite cycle appears in Σ. In other words, the controller can serve as a preemptive measure against malfunctions, improving system reliability. For machines with existing infinite cycles, the use of a controller is often economically more efficient than replacing a faulty machine. It is the only practical solution when the affected machine is inaccessible.

The existence of a controller depends on certain reachability properties of the faulty machine Σ. These properties can be characterized in terms of a numerical matrix of zeros and ones, called the "skeleton matrix" of Σ (§IV). The skeleton matrix determines whether the control objective can be achieved; its derivation is based on the theory of stable realizations of [15] and [16]. A stable realization represents every persistent status of a machine and the transitions from one persistent status to another.

This note continues [6], [7], [8], [9], [10], [11], [12], [4], and [5]. Studies dealing with other aspects of the control of discrete event systems can be found in [13], [2], [14], [1], [3], and others. The existing literature about infinite cycles of asynchronous machines deals with techniques for the design and implementation of machines that are free of infinite cycles. There seem to be no reports regarding the use of controllers to eliminate the effects of infinite cycles in an existing machine.

II. Terminology and Background

For a finite non-empty alphabet A, let A^* be the set of all finite strings of characters of A, and let A^+ be the set of all non-empty strings in A^*. Assume that A does not include the digits 0 and 1. The *length* $|w|$ of a string $w \in A^*$ is the number of characters of w. A *partial function* $f : S_1 \to S_2$ is a function defined over a subset of S_1.

An asynchronous machine Σ is a sextuple (A,Y,X,x_0,f,h), where A, Y, and X are alphabets, x_0 is the initial state, and $f : X{\times}A \to X$ and $h : X{\times}A \to Y$ are partial functions. Here, A is the *input alphabet,* Y is the *output alphabet*, and X is the set of *states*; f is the *recursion function* and h is the *output function*. A *valid pair* $(x,u) \in X{\times}A$ is a point at which f and h are defined.

The machine Σ accepts input strings $u := u_0 u_1 \ldots \in A^*$; in response, it generates a string of states $x_0x_1x_2 \ldots \in X^*$ and a string of output values $y_0y_1y_2 \ldots \in Y^*$, accord-

Niranjan Venkatraman is with the Department of Electrical Engineering, Northern Arizona University, Flagstaff, AZ 86011, USA (phone: 928-523-0373; e-mail: v.niranjan@ieee.org).

Jacob Hammer is with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA (phone: 352-392-4934; e-mail: hammer@mst.ufl.edu).

ing to

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

An input sequence is *permissible* if all pairs (x_k,u_k), $k \geq 0$, are valid pairs. The step counter k is incremented by one at every change of the input or of the state. The machine Σ is an *input/state machine* if the output is equal to the state, i.e., $y_k = x_k$, $k \geq 0$. An input/state machine Σ is represented by the triple (A,X,f), allowing for an arbitrary initial state.

A valid pair $(x,u) \in X \times A$ of Σ is a *stable combination* if $f(x,u) = x$, i.e., if the state x is a fixed point of f. A machine lingers at a stable combination until an input change occurs. A pair (x,u) that is not a stable combination is a *transient combination*. A state is *potentially stable* if it has a stable combination. States that are not potentially stable are invisible to the user, since the machine can never linger at them. It is customary to ignore states that are not potentially stable.

A transient pair (x,u) initiates a chain of transitions $x_1 = f(x,u)$, $x_2 = f(x_1,u)$, ..., where the input character u is kept fixed. If this chain of transitions ends, then there is an integer $q \geq 1$ such that the state $x' := f(x_q,u)$ of the chain satisfies $x' = f(x',u)$, i.e., (x',u) is a stable combination. Then, x' is called the *next stable state* of x with the input value u. If this chain of transitions does not terminate, then the pair (x,u) is part of an *infinite cycle*.

If the input of an asynchronous machine changes while the machine is undergoing transitions, the machine's response may become unpredictable, since the state at the time of the input change is unpredictable. To avoid this uncertainty, asynchronous machines are normally operated in *fundamental mode*, where only one variable of the machine is allowed to change at a time. In fundamental mode operation, a change of the input is allowed only while the machine is in a stable combination. For configuration (1), this means that the output of C must remain constant while Σ is undergoing transitions; and the output of Σ must remain constant while C is undergoing transitions.

Fundamental mode operation is impossible when a machine is in an infinite cycle, since, in order to take the machine out of the infinite cycle, the machine's input must be changed during the course of the cycle. The outcome of such an input change may be unpredictable. In this note, asynchronous machines operate in fundamental mode in all cases, except during an infinite cycle, as follows.

(2) DEFINITION. An asynchronous machine Σ operates in *semi-fundamental mode*, if it operates in fundamental mode when not in an infinite cycle. ♦

I. Generalized Realizations

Consider a machine Σ with the state set $X = \{x^1, x^2, ..., x^n\}$ and the recursion function f. An *infinite cycle* $\chi = \{x^{j1}, x^{j2}, ..., x^{jp}; a\}$ of Σ with the input character a and the states $x^{j1}, x^{j2}, ..., x^{jp}$ is described by

$$x^{jk+1} = f(x^{jk},a),\ k = 1, ..., p-1,$$
$$x^{j1} = f(x^{jp},a).$$

The *state set* of χ is $X(\chi) := \{x^{j1}, x^{j2}, ..., x^{jp}\}$.

The *length* ℓ of the infinite cycle χ is the number of distinct states it includes, i.e., $\ell = p$ in this case. When the length of an infinite cycle is 1, say $\chi = \{x; a\}$, then we have $x = f(x,a)$, i.e., (x,a) is stable combination of Σ. Thus, an infinite cycle of length 1 is a stable combination. *In the sequel, unless specifically stated otherwise, "infinite cycle" refers to an infinite cycle of length greater than 1.*

Let (x,u) be a valid pair of the machine $\Sigma = (A,X,Y,x_0,f,h)$, and assume it has a next stable state x'. The *stable recursion function* $s : X \times A \rightarrow X$ of Σ is defined by setting $s(x,u) := x'$ for every valid pair (x,u) that has a next stable state. When s is used as a recursion function, it induces the *stable-state machine* $\Sigma_{|s} = (A,X,Y,x_0,s,h)$.

When Σ has infinite cycles, being in an infinite cycle is clearly a persistent status of the machine, and this status is definitely experienced by the machine's user. The notion of stable-state machine is generalized to take this fact into account, as follows (see [15] and [16] for details).

(3) DEFINITION. Let Σ be an asynchronous machine with the state set $X = \{x^1, ..., x^n\}$ and $t > 0$ infinite cycles $\chi_1, ..., \chi_t$ of length greater than 1. With each infinite cycle χ_i, associate a new state x^{n+i}, called a *cycle state*. The set $\tilde{X} := \{x^1, ..., x^n, x^{n+1}, ..., x^{n+t}\}$ is *the augmented state set* of Σ; its elements are the *generalized states* of Σ.

A pair $(x,u) \in \tilde{X} \times A$ is a *generalized valid pair* of Σ if one of the following holds: (i) $x \in X$ and (x,u) is a valid pair of Σ; or (ii) $x = x^{n+i}$ for an integer $i \in \{1, ..., t\}$ and u forms a valid pair with each state of the infinite cycle χ_i.

A pair $(x,u) \in \tilde{X} \times A$ is a *generalized stable combination* if one of the following holds: (i) $x \in X$ and (x,u) is a stable combination of Σ; or (ii) $x = x^{n+i}$ for an integer $i \in \{1, ..., t\}$ and u is the input character of the infinite cycle χ_i. ♦

Thus, every persistent status of an asynchronous machine (a stable combination or an infinite cycle) is described by a generalized stable combination.

(4) EXAMPLE. Consider the machine Σ with the input set $A = \{a, b, c\}$, the state set $X = \{x^1, x^2, x^3\}$, and the transition function f given by the state transition table:

	a	b	c
x^1	x^1	x^3	x^1
x^2	x^2	x^3	x^3
x^3	x^2	x^2	x^3

This machine has one infinite cycle of length bigger than one: $\chi_1 := \{x^2, x^3; b\}$. With this infinite cycle, associate a cycle state x^4. The augmented state set of Σ is then $\tilde{X} = \{x^1, x^2, x^3, x^4\}$. The generalized stable combinations of Σ are (x^4,b), (x^1,a), (x^2,a), (x^1,c), and (x^3,c). ♦

We introduce now a new recursion function over the augmented state set $\tilde{X} = \{x^1, ..., x^{n+t}\}$ of a machine $\Sigma = (A,Y,X,x_0,f,h)$. Let $\chi_1, ..., \chi_t$ be the infinite cycles of Σ. First, define a partial function $s^e : X\times A \to \tilde{X}$ over all valid pairs $(x,u) \in X\times A$ of Σ by setting

$$s^e(x,u) = \begin{cases} s(x,u) \text{ if } (x,u) \text{ has a next stable state,} \\ x^{n+i} \text{ if } (f(x,u),u) \text{ is a pair of the infinite cycle } \chi_i. \end{cases}$$

Recall that $X(\chi)$ denotes the set of states included in an infinite cycle χ. Let u be an input character that forms valid pairs with all states of $X(\chi)$. Denote by $s^e[X(\chi),u]$ the image of the set $X(\chi)\times u$ through s^e, namely,

$$s^e[X(\chi),u] := \{x' \in \tilde{X} : x' = s^e(x,u) \text{ and } x \in X(\chi)\}.$$

Note that $s^e[X(\chi),u]$ can be a single state or a set of states, depending on $X(\chi)$ and on u. The following notion is critical to the control of asynchronous machines with infinite cycles. It defines a function whose values are subsets of $\tilde{X}$. When the value consists of a single element x of $\tilde{X}$, we shall drop the (formal) distinction between the element $x \in \tilde{X}$ and the subset $\{x\} \subset \tilde{X}$.

(5) DEFINITION. Let $\Sigma = (A,Y,X,x_0,f,h)$ be an asynchronous machine with the infinite cycles $\chi_1, ..., \chi_t$ and the augmented state set $\tilde{X} = \{x^1, ..., x^{n+t}\}$. Denote by $P(\tilde{X})$ the set of all subsets of $\tilde{X}$. The *generalized stable recursion function* $s : \tilde{X}\times A \to P(\tilde{X})$ of Σ is defined over all generalized valid pairs $(x,u) \in \tilde{X}\times A$ of Σ by

$$s(x,u) := \begin{cases} s^e(x,u) \text{ if } x \in X, \\ s^e[X(\chi_i),u] \text{ if } x = x^{n+i}, i = 1, ..., t. \end{cases}$$

The *generalized stable-state machine* $\Sigma_{|s} = (A,\tilde{X},s)$ of Σ is an input/state machine with the state set $\tilde{X}$ and the recursion function s. ♦

(6) EXAMPLE. A direct examination shows that the generalized stable recursion function s of the machine Σ of Example 4 is given by

	a	b	c
x^1	x^1	x^4	x^1
x^2	x^2	x^4	x^3
x^3	x^2	x^4	x^3
x^4	x^2	x^4	x^3

♦

Note that stable combinations of the generalized stable state machine $\Sigma_{|s}$ represent either stable combinations of the original machine Σ or infinite cycles of Σ. This leads to the following conclusion, which motivates the use of generalized stable state machines ([15], [16]).

(7) PROPOSITION. Semi-fundamental mode operation of a machine Σ is equivalent to fundamental mode operation of its generalized stable state machine $\Sigma_{|s}$. ♦

The following relates to a basic concept of Automata theory (compare to [11], [12], [4], and [5]).

(8) DEFINITION. Let $\Sigma = (A,Y,X,x_0,f,h)$ and $\Sigma' = (A,Y,X',\zeta_0,f',h')$ be two machines having the same input and the same output sets, and let $\Sigma_{|s}$ and $\Sigma'_{|s}$ be the generalized stable state machines of Σ and Σ', respectively. Two states $x \in X$ and $\zeta \in X'$ are *stably equivalent* ($x \equiv \zeta$) if the following is true: When $\Sigma_{|s}$ starts from the state x and $\Sigma'_{|s}$ starts from the state ζ, then (i) $\Sigma_{|s}$ and $\Sigma'_{|s}$ have the same permissible input strings; and (ii) $\Sigma_{|s}$ and $\Sigma'_{|s}$ generate the same output string for every permissible input string. Two machines Σ and Σ' are *stably equivalent* if their initial states are stably equivalent, i.e., if $x_0 \equiv \zeta_0$; in such case, we write $\Sigma = \Sigma'$. ♦

Stably equivalent machines appear identical to a user.

II. The Skeleton Matrix

Consider again the generalizes stable state machine $\Sigma_{|s} = (A,\tilde{X},s)$ with the generalized state set $\tilde{X} = \{x^1, ..., x^{n+t}\}$ and the generalized stable recursion function s. Recall that a *critical race* is a transition whose outcome is not uniquely predetermined. Assume that the input string $u \in A^+$, when applied at the generalized state x^j of $\Sigma_{|s}$, creates a critical race with the outcomes x^p and x^q, $p \neq q$. Assume further that there are input strings that take $\Sigma_{|s}$ from these states to a common target state x^s via deterministic transitions. Namely, assume that there are input strings u^1, $u^2 \in A^+$, where u^1 takes $\Sigma_{|s}$ from x^p to x^s deterministically, while u^2 takes $\Sigma_{|s}$ from x^q to x^s deterministically; i.e., $x^s = s(x^p,u^1) = s(x^q,u^2)$. Then, by using state-feedback control, we can induce a deterministic transition from x^j to x^s as follows: apply the input string u at the state x^j, and check the outcome. If the outcome is x^p, then continue with the input string u^1; if the outcome is x^q, then continue with the input string u^2. This leads to the deterministic (i.e., unique) outcome x^s, masking the effects of the critical race along the way. Note that all transitions are in semi-fundamental mode operation. To generalize, we need the following concept. (Let $\Pi_x : \tilde{X}\times A \to \tilde{X} : \Pi_x(x,u) = x$ be the standard projection onto $\tilde{X}$.)

(9) DEFINITION. Let Σ be an asynchronous machine inducing the generalized stable state machine $\Sigma_{|s} = (A,\tilde{X},s)$, where $\tilde{X} = \{x^1, ..., x^{n+t}\}$. A *feedback trajectory* from the generalized state x^j to the generalized state x^i is a list $\{S_0, S_1, S_2, ..., S_p\}$ of sets of valid pairs of $\Sigma_{|s}$ with the following properties:

(i) $S_0 = \{(x^j,u_0)\}$, where u_0 is a character of A;

(ii) $s[S_\alpha] \subset \Pi_x[S_{\alpha+1}]$, $\alpha = 0, ..., p-1$;

(iii) $s[S_p] = \{x^i\}$. ♦

(10) EXAMPLE. A feedback trajectory from x^1 to x^2 for the generalized stable state machine $\Sigma_{|s}$ of Example 6 is $S_0 = \{(x^1,b)\}$, $S_1 = \{(x^4,a)\}$, $S_2 = \{(x^2,a)\}$. ♦

The argument of the first paragraph of this section can be readily generalized to a proof of the next statement, which shows that feedback trajectories characterize the existence of feedback controllers.

(11) PROPOSITION. Let x^j and x^i be generalized states of a machine Σ. The following are equivalent.

(a) There is a state feedback controller that induces a deterministic transition from x^j to x^i in semi-fundamental mode operation.

(b) There is a feedback trajectory from x^j to x^i. ♦

The following notion plays an important role.

(12) DEFINITION. Let Σ be an asynchronous machine with the augmented state set $\tilde{X}$ and the generalized stable recursion function s. A generalized state $x' \in \tilde{X}$ is *stably reachable* from a generalized state $x \in \tilde{X}$ if there is a input string $u = u_0u_1\ldots u_k$ of Σ for which $x' \in s(x,u)$. ♦

To perform computations related to stable reachability, we need the following matrix (see [15] and [16] for details).

(13) DEFINITION. Let $\Sigma_{|s} = (A,\tilde{X},s)$ be a generalized stable state machine with the augmented state set $\tilde{X} = \{x^1, ..., x^{n+t}\}$. Denote by $s^*(x^i,x^j)$ the set of all input characters $u \in A$ for which $x^i \in s(x^j,u)$, and let N be a character not included in the alphabet A. Then, the *matrix of one-step generalized stable transitions* $R(\Sigma_{|s})$ is an $(n+t)\times(n+t)$ matrix whose (i,j) entry is given by

$$R_{ij}(\Sigma_{|s}) = \begin{cases} s^*(x^i,x^j) & \text{if } s^*(x^i,x^j) \neq \varnothing, \\ N & \text{otherwise, } i, j = 1, ..., n+t. \end{cases}$$ ♦

The matrix $R(\Sigma_{|s})$ characterizes all one step transitions of $\Sigma_{|s}$. Its (i,j) entry, if not N, consists of all (single) input characters that take $\Sigma_{|s}$ from x^j to a generalized stable combination with x^i. An (i,j) entry of N indicates that $\Sigma_{|s}$ cannot be driven from x^j to a generalized stable combination with x^i by applying a single input character.

Recall that A^* is the set of strings of characters of A. We describe now two operations on the matrix $R(\Sigma_{|s})$. First, an operation that mimics matrix addition ([11] and [12]). Let w_i be a subset of strings or the character N, $i = 1, 2$. The operation $\uplus$ of *unison* is defined by

$$w_1 \uplus w_2 := \begin{cases} w_1 \cup w_2 & \text{if } w_1 \subset A^* \text{ and } w_2 \subset A^*, \\ w_1 & \text{if } w_1 \subset A^* \text{ and } w_2 = N, \\ w_2 & \text{if } w_1 = N \text{ and } w_2 \subset A^*, \\ N & \text{if } w_1 = w_2 = N. \end{cases}$$

Here, N is treated like the empty set it represents. For $n\times n$ matrices, *unison* is entrywise: $C_{ij} := A_{ij} \uplus B_{ij}$, $i, j = 1, ..., n$.

Concatenation of strings $w_1, w_2 \in A^* \cup N$ is given by

$$\text{conc}(w_1,w_2) := \begin{cases} w_2w_1 & \text{if } w_1, w_2 \in A^*, \\ N & \text{if } w_1 = N \text{ or } w_2 = N. \end{cases}$$

More generally, let $W = \{w_1, w_2, \ldots, w_q\}$ and $V = \{v_1, v_2, \ldots, v_r\}$ be two subsets, whose elements are either strings of A^* or the character N. Define

$$\text{conc}(W,V) := \biguplus_{\substack{i=1,\ldots,q \\ j=1,\ldots,r}} \text{conc}(w_i,v_j).$$

Note that the concatenation result is either a subset of A^* or the character N, and N takes the role of a "zero". The *product* $Z := CD$ of two $n\times n$ matrices C and D is an $n\times n$ matrix with the entries

$$Z_{ij} := \biguplus_{k=1}^{n} \text{conc}(C_{ik},D_{kj}),\ i,j = 1, \ldots, n.$$

Using this product, we can define the powers

$$R^q(\Sigma_{|s}) := R^{q-1}(\Sigma_{|s})R(\Sigma_{|s}),\ q = 2, 3, ...$$

If not N, the (i,j) entry of $R^q(\Sigma_{|s})$ consists of all input strings that may take x^j to a generalized stable combination with x^i in q generalized stable transitions (these transitions may not be deterministic). Define

$$R^{(q)}(\Sigma_{|s}) := \biguplus_{p=1\ldots q} R^p(\Sigma_{|s}),\ q = 2, 3, ...$$

If not N, the (i,j) entry of $R^{(q)}(\Sigma_{|s})$ consists of all strings that may take the machine $\Sigma_{|s}$ from x^j to a generalized stable combination with x^i in q or fewer generalized stable transitions (these transitions may not be deterministic). The next fact's proof is similar to [[12] Lemma 3.9].

(14) LEMMA. Let Σ be an asynchronous machine with n states, t infinite cycles, and the generalized stable-state machine $\Sigma_{|s}$. The following two statements are equivalent:

(i) The generalized state x^i is stably reachable from the generalized state x^j.

(ii) The (i,j) entry of $R^{(n+t-1)}(\Sigma_{|s})$ is not N. ♦

Thus, the matrix $R^{(n+t-1)}(\Sigma_{|s})$ characterizes all transitions possible for the generalized stable-state machine $\Sigma_{|s}$.

In the matrix $R(\Sigma_{|s})$, a critical race is represented by the presence of an input character u in more than one entry of a column, as this indicates that the input character u transits the state corresponding to that column to several different outcomes. This leads to the following.

(15) PROPOSITION. Let $R(\Sigma_{|s})$ be the one-step matrix of stable transitions of a machine Σ with n states and t infinite cycles. Then, the following are equivalent for all input strings $u \in A^+$ and for all $j = 1, ..., n+t$.

(i) Applying u at the generalized state x^j results in a critical race.

(ii) The string u appears in more than one entry of column j of the matrix $R^{(n+t-1)}(\Sigma_{|s})$. ♦

The next algorithm characterizes all pairs of generalized states that are connected by a feedback trajectory. It operates on the matrix of one-step generalized stable transitions, gradually transforming it into a numerical matrix of zeros and ones. In the resulting matrix, an entry of 1 appears in position (i,j) if and only if there is a feedback trajectory from x^j to x^i. This characterizes all the ways in which state feedback controllers can affect a machine Σ.

We define a *meet* operation on strings of A^+ and the digits 0 and 1. Let ω be a character not in A. Set

$0 \wedge 0 := 0$, $0 \wedge 1 = 1 \wedge 0 := 0$, $1 \wedge 1 := 1$, $0 \wedge a = a \wedge 0 := 0$, $1 \wedge a = a \wedge 1 := \omega$, for all $a \in A^+$.

The *meet* of two vectors is defined entrywise, e.g.,

$$(1,1,0,a,0) \wedge (1,0,0,1,a) = (1,0,0,\omega,0).$$

(16) ALGORITHM. Let $R(\Sigma_{|s})$ be the matrix of one-step generalized stable transitions of Σ. Compute $R^{(n+t-1)}(\Sigma_{|s})$.

Step 1. Replace all entries of N in the matrix $R^{(n+t-1)}(\Sigma_{|s})$

by the digit 0; denote the resulting matrix by K^1.

Step 2. Perform (a), (b), (c), and (d) below for each $i, j = 1, ..., n+t$; then, continue to (e):

(a) If the (i,j) entry K^1_{ij} of K^1 includes a string of A^+ that does not appear in any other entry of the same column j, then:

(b) Delete the strings included in K^1_{ij} from all entries of column j of K^1.

(c) Replace all resulting empty entries by the digit 0.

(d) Replace entry K^1_{ij} by the digit 1.

(e) Denote the resulting matrix by $K'(1)$. Delete from $K'(1)$ all strings of A^+ whose length is greater than 1. Replace all empty entries by the digit 0. Denote the resulting matrix by $K(1)$. Set $k := 1, \alpha := 1$.

Step 3. If $k = n+t+1$, then perform the following:

(a) Set $K_\alpha(\Sigma) := K(k)$.

(b) If $\alpha \geq 2$ and $K_\alpha(\Sigma) = K_{\alpha-1}(\Sigma)$, then replace by 0 all entries of $K_\alpha(\Sigma)$ that are not 1; denote the resulting matrix by $K_g(\Sigma)$, and terminate the algorithm. Otherwise, replace α by $\alpha+1$, set $k := 1$, and continue to Step 4.

Step 4. If all entries of column k of the matrix $K(k)$ are 1 or 0, then set $K(k+1) := K(k)$, and repeat from Step 3 with the value $k+1$ for k. Otherwise, proceed to Step 5.

Step 5. (a) If the character $u \in A$ appears in column k of $K(k)$, then let $i_1, i_2, ..., i_q$ be the rows of column k of $K(k)$ that include u. Denote by $J(u)$ the meet of columns $i_1, i_2, ..., i_q$ of the matrix $K(k)$.

(b) If $J(u)$ has no entries other than 0 or 1, then delete u from all entries of column k of the matrix $K(k)$; set all empty entries to the value 0. Continue to (c).

(c) If $J(u)$ has no entries of 1, then return to Step 4. Otherwise, continue to (d).

(d) If $J(u)$ has entries of 1, let $j_1, ..., j_r$ be the entries of $J(u)$ having the value 1. Let $S(k)$ be the set of columns of $K(k)$ that consists of column k and of every column that has the digit 1 in row k. In $K(k)$, perform the following operations on every column of $S(k)$:

(1) Delete from the column all occurrences of input characters that appear in rows $j_1, ..., j_r$ of the column.

(2) Replace rows $j_1, ..., j_r$ of the column by the digit 1.

(3) If any entries of $K(k)$ remain empty, then replace them by the digit 0. Return to Step 4. ♦

(17) DEFINITION. The outcome $K_g(\Sigma)$ of Algorithm 16 is called the *generalized skeleton matrix* of Σ. ♦

(18) EXAMPLE. For the machine of Example 6,

$$R(\Sigma_{|s}) = \begin{pmatrix} \{a,c\} & N & N & N \\ N & a & a & a \\ N & c & c & c \\ b & b & b & b \end{pmatrix}$$

and Algorithm 16 yields

$$K_g(\Sigma) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}. ♦ \qquad (19)$$

The next fact indicates the significance of Algorithm 16.

(20) PROPOSITION. Let $K_g(\Sigma)$ be the generalized skeleton matrix of Σ, and let x^i and x^j be two generalized states of Σ. Then, the following are equivalent.

(i) There is a feedback trajectory from x^j to x^i.

(ii) The (i,j) entry of $K_g(\Sigma)$ is 1.

Proof. Let s be the generalized stable recursion function of Σ. Assume that there is a feedback trajectory $S_0, S_1, ... S_p$ from x^j to x^i. By the definition of feedback trajectory, $S_k = \{(x^{i(1,k)},u(1,k)), ..., (x^{i(q(k),k)},u(q(k),k))\}$, $k = 1, ..., p-1$, $S_0 = \{(x^j,u)\}$, $s[S_p] = x^i$, and $s(x^{i(r,k)},u(r,k)) \in \Pi_x[S_{k+1}]$, $r = 1, ..., q(k)$, $k = 0, ..., p-1$. Applying Step 5 of Algorithm 16 recursively at each of the steps $k = p, p-1, ..., 1$ (reverse order), while using the previous relations, shows that $K_g(\Sigma)$ has an entry of 1 in position (i,j). Thus, (i) entails (ii).

Conversely, assume that there is an entry of 1 in position (i,j) of $K_g(\Sigma)$. Consider first the matrix $K(1)$ of Algorithm 16. By construction, an entry of 1 in position (i,j) of $K(1)$ indicates the existence of a deterministic transition from x^j to x^i. Next, referring to Step 5 of Algorithm 16, let k_0 be the first value of k at which the condition of Step 5(a) of Algorithm 16 is satisfied. Then, Σ has the critical race $s(x^{k_0},u) = \{x^{i_1}, ..., x^{i_q}\}$. By the construction of $J(u)$ in Step 5(d) of Algorithm 16, it follows that an entry of 1 in row j of $J(u)$ indicates that there is a deterministic transition from each one of the generalized states $x^{i_1}, ..., x^{i_q}$ to the generalized state x^j. Recalling the first paragraph of section IV, this implies the existence of a feedback trajectory from x^{k_0} to x^j. Applying similar reasoning recursively at every cycle of Algorithm 16, we conclude that an entry of 1 in position (i,j) of $K_g(\Sigma)$ implies the existence of a feedback trajectory from x^j to x^i. Finally, if $\alpha \geq 2$ and $K_\alpha(\Sigma) = K_{\alpha-1}(\Sigma)$ in Step 3(b) of Algorithm 16, then any strings of A^+ present in $K_\alpha(\Sigma)$ involve critical races whose outcomes cannot be guided toward a single state via a feedback trajectory. Thus, (ii) implies (i). ♦

The generalized skeleton matrix plays a critical role in our discussion, reminiscent of the role played by the skeleton matrix in [11], [12], [4], and [5]. Combining Propositions 7, 11, and 20, leads us to one of the main results of this note: a characterization of all possibilities of controlling an asynchronous machine.

(21) THEOREM. Let Σ be a machine with the generalized skeleton matrix $K_g(\Sigma)$, and let x^i and x^j be two generalized states of Σ. The following are equivalent.

(a) There is a feedback controller that takes Σ from x^j to x^i in semi-fundamental mode operation.

(b) The (i,j) entry of $K_g(\Sigma)$ is 1. ♦

III. The Model Matching Problem

(22) PROBLEM. Let Σ and Σ' be input/state machines having the same input and the same output alphabets,

where Σ' is a stable-state machine. Find necessary and sufficient conditions for the existence of a controller C for which the generalized stable-state machine $\Sigma_{c|s}$ is stably equivalent to Σ' for all initial conditions. ♦

The controller C of Problem 22 makes the closed loop system imitate the behavior of Σ'. The machine Σ' is called the *model*, and it has no infinite cycles. Thus, when model matching is achieved, the controller C eliminates the effects of the infinite cycles of Σ, in addition to inducing desirable dynamical behavior.

Once the controller C has been activated, there is no more interest in transitions of Σ that either start or end at generalized stable combinations with cycle states, since these would indicate the presence of persistent infinite cycles in the closed loop system. Now, transitions that start at generalized stable combinations with cycle states are represented by the last t columns of $K_g(\Sigma)$, while transitions that terminate at generalized stable combinations with cycle states are represented by the last t rows of $K_g(\Sigma)$. Thus, for design purposes, the last t rows and the last t columns of $K_g(\Sigma)$ can be deleted (however, these are critical during the construction of $K_g(\Sigma)$).

(23) DEFINITION. Let Σ be an asynchronous machine with n states and t infinite cycles. The *skeleton matrix* $K(\Sigma)$ is obtained by deleting the last t rows and the last t columns of the generalized skeleton matrix $K_g(\Sigma)$. ♦

The entries of the skeleton matrix may still include transitions that pass transiently through infinite cycles. Only transitions that start or end at infinite cycles are eliminated.

(24) EXAMPLE. For $K_g(\Sigma)$ of (19),

$$K(\Sigma) = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}. ♦$$

We can now to state a solution of the Model Matching Problem 22. Clearly, simulating the behavior of the model Σ' means finding a controller C for which the generalized stable state machine $\Sigma_{c|s}$ has the same transitions as the model Σ'. All possible transitions of Σ' are represented by entries of 1 in its skeleton matrix $K(\Sigma')$. Consider, for example, one of these entries: suppose that $K(\Sigma')$ has an entry of 1 in position (i,j). This indicates that Σ' has a stable-state transition from the state x^j to the state x^i. Then, for $\Sigma_{c|s}$ to emulate Σ', the controller C must drive Σ to perform a similar transition in semi-fundamental mode, i.e., a transition from x^j to x^i. In view of Theorem 21, the latter is possible if and only if the skeleton matrix $K(\Sigma)$ has an entry of 1 in position (i,j). In other words, the model matching problem has a solution if an only if $K(\Sigma)$ has an entry of 1 in each position in which $K(\Sigma')$ has an entry of 1. As entries of $K(\Sigma)$ or of $K(\Sigma')$ can only be 0 or 1, this leads to the following. (Given two $n{\times}n$ numerical matrices A and B, the inequality $A \geq B$ is interpreted entrywise, i.e., $A_{ij} \geq B_{ij}$ for all $i, j = 1, ..., n$.)

(25) THEOREM. Let Σ be an input/state machine with the skeleton matrix $K(\Sigma)$, and let Σ' be a stable-state input/state machine with the skeleton matrix $K(\Sigma')$. Then, the following two statements are equivalent.

(i) There is a state-feedback controller C for which the closed loop system $\Sigma_{c|s}$ is stably equivalent to Σ', where $\Sigma_{c|s}$ operates in semi-fundamental mode.

(ii) The skeleton matrices satisfy $K(\Sigma) \geq K(\Sigma')$. ♦

Theorem 25 provides a simple necessary and sufficient condition for the existence of a solution of the model matching problem for machines with infinite cycles. The construction of appropriate controllers is described in [15]. Note that the closed loop system Σ_c has no persistent infinite cycles. Still, within the closed loop, the machine Σ may pass transiently through an infinite cycle on its way from one stable combination to another.

IV. References

[1] G. Barrett and S. Lafortune, "Bisimulation, the Supervisory Control Problem, and Strong Model Matching for Finite State Machines," *J. of Discrete Event Dynamic Systems*, vol. 8, no. 4, 1998, pp. 377–429.

[2] M. D. Dibenedetto, A. Saldanha, and A. Sangiovanni–Vencentelli, "Model matching for finite state machines," *Proc. of the IEEE Conference on Decision and Control*, vol. 3, 1994, pp. 3117–3124.

[3] M. D. Dibenedetto, A. Sangiovanni–Vincentelli, and T. Villa, "Model matching for finite–state machines," *IEEE Trans. on Automatic Control*, vol. 46, no. 11, 2001, pp. 1726–1743.

[4] X. J. Geng and J. Hammer, "Input/output control of asynchronous sequential machines", IEEE Trans. Automatic Control (to appear).

[5] X. J. Geng and J. Hammer, "Asynchronous sequential machines: input/output control", *Proc. of the 12th Mediterranean Conference on Control and Automation*, Kusadasi, Turkey, June 2004.

[6] J. Hammer, "On some control problems in molecular biology," *Proc. of the IEEE Conference on Decision and Control*, December 1994.

[7] J. Hammer, "On the modeling and control of biological signal chains," *Proc. IEEE Conf. on Decision and Control*, December 1995.

[8] J. Hammer, "On the corrective control of sequential machines," *International J. of Control*, vol. 65, pp. 249-276.

[9] J. Hammer, "On the control of incompletely described sequential machines," *International J. of Control*, vol. 63, no. 6, pp. 1005-1028.

[10] J. Hammer, "On the control of sequential machines with disturbances," *International J. Control*, vol. 67, no. 3, pp. 307-331.

[11] T. E. Murphy, X. J. Geng, and J. Hammer, "Controlling races in asynchronous sequential machines", *Proc. of the IFAC World Congress*, Barcelona, July 2002.

[12] T. E. Murphy, X. J. Geng, and J. Hammer, "On the control of asynchronous machines with races", IEEE Transactions on Automatic Control, vol. 48, no. 6, pp. 1073-1081.

[13] P. J. G. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control and Optimization*, vol. 25, no. 1, 1987, pp. 206-230.

[14] J. G. Thistle and W. M. Wonham, "Control of infinite behavior of finite automata," *SIAM J. Cont. Opt.*, vol. 32-4, 1994, pp. 1075-1097.

[15] N. Venkatraman and J. Hammer, "On the control of asynchronous sequential machines with infinite cycles," International Journal of Control (to appear).

[16] N. Venkatraman and J. Hammer, "Stable realizations of asynchronous sequential machines with infinite cycles," Proceedings of the 6th Asian Control Conference (to appear).