Proceeding of the IFAC World Congress, Barcelona, July 2002

CONTROLLING RACES IN ASYNCHRONOUS SEQUENTIAL MACHINES

by

Thomas E. Murphy†
Xiaojun Geng‡, and
Jacob Hammer‡

† Department of Mathematics and Computer Science, Georgia Southern University, Statesboro, GA 30460-8093, USA.
‡ Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611-6130, USA.

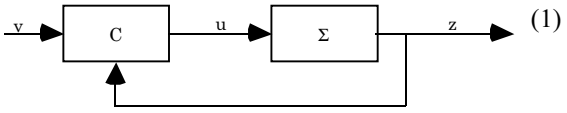
ABSTRACT

State feedback controllers can be used to restore predictable behavior of asynchronous sequential machines that are afflicted by a critical race. The present note presents necessary and sufficient conditions for the existence of such controllers. The controllers also assign a specified behavior to the controlled machine. *Copyright ©IFAC 2002*

Keywords: asynchronous sequential logic, critical races, finite state machines, control.

1. INTRODUCTION

A critical race is a fault of an asynchronous sequential machine. It causes the machine to exhibit unpredictable behavior. Critical races can occur because of malfunctions, design flaws, or implementation flaws. The present note deals with the development of state feedback controllers that control a race-afflicted machine Σ to render its response predictable and acceptable. These controllers provide an attractive option in cases where replacement of the entire machine is not practical or not economical. The control configuration is as follows.



(1)

Here, Σ is the defective machine and C is the controller. Both Σ and C are asynchronous sequential machines. Denote by Σ_c the input/output relation of the closed loop system (1).

The controller C eliminates the effects of critical races of Σ, and assigns to the closed loop system Σ_c a (predictable and) desirable behavior. The desirable behavior is specified in terms of a model that Σ_c has to match; C is called a *corrective controller*.

Consider a sequential machine Σ afflicted by a critical race having q possible outcomes. To model the race, represent Σ by a family $M = \{\Sigma^1, ..., \Sigma^q\}$ of q asynchronous machine models. Each model Σ^i represents the behavior of the machine Σ for one possible outcome of the race. The family M is called a *critical race family*.

The controller C, when it exists, controls Σ so that the closed loop system Σ_c has the same response, irrespective of which one of the models $\Sigma^1, ..., \Sigma^q$ is in effect. In this way, the response of the closed loop system becomes predictable, being independent of the outcome of the race.

In order to explain intuitively the way the controller C functions, recall that an asynchronous machine may have two kinds of states: *stable states* -- states at which the machine lingers until an input change occurs; and *unstable states* -- states through which the machine passes rapidly, ideally in zero time. When moving from one stable state to another, an asynchronous sequential machine may pass in rapid succession through several unstable states. Unstable states are not noticeable to the user, since the machine does not linger at them. In qualitative terms, the corrective controller C functions by transforming into unstable states all states at which the members of the critical race family M differ from each other. In this way, the differences among the various outcomes of the race become un-noticeable.

The existence of a corrective controller C depends on certain reachability properties. To represent these, a special matrix of zeros and ones, called the "skeleton matrix", is associated with each critical race family M. It characterizes all common state transitions of the members of M (see section 3 below). The existence of C can be determined by a

simple examination of the skeleton matrix. Due to space limitations, the complexity of the controller C is not addressed in this note.

Various aspects of the control of families of sequential machines are discussed in Hammer (1994, 1995, 1996a and b). The mathematical framework of the present discussion is based to some extent on Eilenberg (1974).

The notion of a critical race has been an important topic in the literature about digital circuit design for almost half a century. Some background on the subject can be found in Huffman (1954a, b, and 1957), Kohavi (1970), Maki and Tracey (1971), Datta and Bandyopadhyay (1988), Chu (1994), and Lavagno and Moon (1995). The problem of eliminating the effects of a race by using feedback controllers was first addressed in Murphy, Geng, and Hammer (2000).

Studies dealing with other aspects of the control of finite state machines can be found in Ramadge and Wonham (1987), Ozveren, Willsky, and Antsaklis (1991), Thistle and Wonham (1994), DiBenedetto, Saldanha, and Sangiovanni-Vincentelli (1994 and 1995), Barrett and LaFortune (1997), the references cited in these papers, and others. An important difference between these papers and the present discussion is that here the concentration is on the control of asynchronous machines. This brings to the forefront the distinction among stable and unstable states, and requires fundamental mode operation.

2. TERMINOLOGY AND BACKGROUND

A *finite state machine* Σ is a quintuple (A,Y,X,f,h), where A, Y, and X are finite non-empty sets, and $f : X\times A \to X$ and $h : X\times A \to Y$ are partial functions. Here, A is the set of *input values*, or the *input alphabet*; Y is the set of *output values*; and X is the *set of states*. The function f is called the *recursion function* (or the state transition function), and h is called the *output function*. A point $(x,u) \in X\times A$ for which the partial function f is defined is called a *valid pair*. The present note is restricted to *input/state* machines, i.e., to machines Σ whose output is their state. In this case, $Y = X$, and the output function h is not used. An input/state machine is then characterized by a triple (A,X,f).

An input/state machine Σ accepts an input sequence $u = (u_0, u_1, u_2, ...)$ of elements of the input set A. From an initial condition x_0, it creates in response a sequence of states $x_0, x_1, x_2, ... \in X$ according to the recursion

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

The integer k advances by one when a state transition or a change in an input value occurs.

A valid pair $(x,u) \in X\times A$ of the machine Σ is a *stable combination* if $f(x,u) = x$, i.e., a "fixed point" of f. A state x for which there is a stable combination is called a *potentially stable state*. States of Σ that are not potentially stable serve only as transition states, and the machine can never linger at them. Such states are omitted from the model.

2.1 The Fundamental Mode and Stable-State Machines.

To guarantee that there is no ambiguity in the response of an asynchronous machine, it is best to allow only one of the input or state variables to change value at any instant of time. A machine that operates under this restriction is said to operate in *fundamental mode*.

In fundamental mode, an input variable can change its value only after the machine has reached a stable combination. In the ensuing discussion, all asynchronous machines operate in fundamental mode. For controllers, feedback ascertains that the machine has reached a stable combination before the controller changes the input value of the machine.

When the state-input pair (x,u) is not a stable combination, the machine Σ will continue from this pair through a chain of state transitions. In fundamental mode, the input value u must be kept constant while this chain of transitions is in progress. The chain of transitions terminates if and only if a stable combination (x',u) with the same input value u is encountered. Then, x' is the *next stable state* of x with the input value u.

If there is no next stable state for x with the input value u, then the machine Σ has an infinite cycle. An infinite cycle cannot be terminated in fundamental mode, since one cannot change the input value while the cycle is in progress. Therefore, machines with infinite cycles are excluded from the present consideration. One can define a partial function $s : X\times A \to X$ by setting $s(x,u) := x'$ for every valid pair (x,u), where x' is the next stable state of x with the input value u. The function s is the *stable recursion function* of Σ. In many ways, the machine Σ is better described by s than by f, since s ignores unstable transitions. Using s instead of f yields the *stable-state machine* $\Sigma_{|s}$ induced by Σ.

Now, consider a string of input values $w = u_0u_1\ldots u_k$ applied to the system Σ from the initial state x. In fundamental mode, the input value u_0 is kept fixed until Σ reaches the next stable state $s(x,u_0)$. Then, the input value switches to u_1, and stays constant until the next stable state $(s(s(x,u_0),u_1)$ is reached. This process continues until the last stable state $s(...s(s(x,u_0),u_1),u_2)...,u_k)$ is reached. This defines a partial function s^*

$$s^*(x,w) := s(...s(s(x,u_0),u_1),u_2)...,u_k).$$

For convenience, the symbol s is also used for s^*.

Finally, consider two machines $\Sigma = (A,Y,X,f,h)$ and $\Sigma' = (A,Y,X',f',h')$ having the same input and output alphabets, and let $x \in X$ and $x' \in X'$ be two states. The states x and x' are *equivalent* if the following is true: When Σ starts from the initial condition x and Σ' starts from the initial condition x', then Σ and Σ' generate the same output string for every (permissible) input string. The machines Σ and Σ'

are *equivalent* if every state of Σ has a corresponding equivalent state of Σ', and every state of Σ' has a corresponding equivalent state of Σ. Such equivalence is indicated by writing $\Sigma = \Sigma'$.

2.2 Races and Race Families.

A state-input pair (r,v) for which the next stable state of the machine is unpredictable is called a *critical race pair* (e.g., Kohavi (1970), Unger (1995)). A critical race may appear because of a component malfunction, an implementation fault, or a design error. For a critical race pair (r,v), the next state can be one of several options, say $\rho_1, ..., \rho_q$, called the *outcomes* of the race.

To represent a critical race, build a family $M = \{\Sigma^1, ..., \Sigma^q\}$ of q sequential machines, all having the same input set, the same output set, and the same state set. The recursion function f_i of Σ^i is the same as the recursion function f of Σ at all points except at the critical race pair (r,v), at which $f_i(r,v) := \rho_i$, $i = 1, ..., q$. Then, M is a *critical race family*. Since in the current context a race must involve a state transition, one has $\rho_i \neq r$ for all $i = 1, ..., q$, and

$$f_i(x,u) = \begin{cases} f(x,u) \text{ for all } (x,u) \neq (r,v); \\ \rho_i \text{ for } (x,u) = (r,v), \text{ where } \rho_i \neq r, \end{cases}$$

$i = 1, ..., q$. Let r_i be the next stable state reached with the input value v from the outcome ρ_i of the race. To prevent an infinite cycle, one must have $r_i \neq r$ for all $i = 1, ..., q$.

2.3 The model-matching problem.

The control configuration (1) is *well posed* if it induces a unique input/output relation (e.g., Hammer (1996a)). The controller C is *proper* if the closed loop system (1) is well posed and operates in fundamental mode.

Let $M = \{\Sigma^1, ..., \Sigma^q\}$ be a critical race family of input/state asynchronous machines. For a member Σ^i of M, let Σ^i_c be the closed loop system obtained when Σ is replaced by Σ^i in (1). Denote by $\Sigma^i_{c|s}$ the stable-state machine induced by Σ^i_c. Now, let Σ' be a race-free stable-state machine having the same input set and the same state set as Σ.

MODEL MATCHING PROBLEM 2. Find necessary and sufficient conditions for the existence of a controller C such that $\Sigma^i_{c|s}$ is equivalent to Σ' (i.e., $\Sigma^i_{c|s} = \Sigma'$) for all $i = 1, ..., q$.

When it exists, the controller C eliminates the effects of the race and assigns a specified stable state behavior to the closed loop system.

3. REACHABILITY

Consider two states x and x' of the machine Σ. The state x' is *stably reachable* from the state x if there is an input string $u = u_0u_1...u_k$ of $\Sigma_{|s}$ for which $x' = s(x,u)$, where s is the stable recursion function of Σ.

Let $X = \{x_1, ..., x_n\}$ be the state set of $\Sigma_{|s}$. The *one-step stable transition matrix* $R(\Sigma)$ is an $n \times n$ matrix whose (i,j) entry $R_{ij}(\Sigma)$ is the set of all (single) input characters $u \in A$ satisfying $x_j = s(x_i,u)$. If $R_{ij}(\Sigma)$ is the empty set, write $R_{ij}(\Sigma) := N$, where N is a character not in A. It indicates that there is no one-step stable transition from x_i to x_j. The *one-step skeleton matrix* $S(\Sigma)$ is defined as follows: its (i,j) entry $S_{ij}(\Sigma)$ is

$$S_{ij}(\Sigma) := 0 \text{ if } R_{ij}(\Sigma) = N, \text{ and}$$
$$S_{ij}(\Sigma) := 1 \text{ if } R_{ij}(\Sigma) \neq N,$$

$i, j \in \{1, ..., n\}$. As seen, $S(\Sigma)$ is an $n \times n$ matrix of zeros and ones. Its (i,j) entry is 1 if and only if there is a one-step stable transition from x_i to x_j. Next, define an operation for skeleton matrices.

Let A, B be two $n \times n$ matrices of zeros and ones. The *combination* AB of A and B is again an $n \times n$ matrix of zeros and ones, whose (i,j) entry is

$$(AB)_{ij} := \max \{A_{ik}B_{kj} : k = 1, ..., n\}, \qquad (3)$$

in analogy to matrix multiplication. By this operation, build "powers" of the one-step skeleton matrix:

$$S^m(\Sigma) = S^{m-1}(\Sigma)S(\Sigma), \; m \geq 1.$$

Call $S^m(\Sigma)$ the *m-step skeleton matrix* of the machine Σ. It can be seen that the (i,j) entry of $S^m(\Sigma)$ is one if and only if it is possible to reach x_j from x_i in exactly m stable transitions.

For an integer $m \geq 2$, the matrix $S^{(m)}(\Sigma)$ is defined by setting its (i,j) entry to be

$$S^{(m)}_{ij}(\Sigma) := \max_{k = 1, ..., m} S^k_{ij}(\Sigma), \; i, j = 1, ..., n.$$

Thus, $S^{(m)}(\Sigma)$ is again a matrix of zeros and ones. It can be shown that

$$S^{(p)}(\Sigma) = S^{(n-1)}(\Sigma) \text{ for all } p \geq n-1. \qquad (4)$$

where n is the number of states. This implies that x_j is stably reachable from x_i (in any number of stable transitions) if and only if $S^{(n-1)}_{ij}(\Sigma) = 1$ (see Murphy, Geng, and Hammer (2000) for details). Call $S^{(n-1)}(\Sigma)$ the *skeleton matrix* of Σ, and denote

$$K(\Sigma) := S^{(n-1)}(\Sigma). \qquad (5)$$

Next, the *skeleton matrix* $K(M)$ of a critical race family $M = \{\Sigma^1, ..., \Sigma^q\}$ is defined by

$$K_{ij}(M) := \min \{K_{ij}(\Sigma^k) : k = 1, ..., q\}. \qquad (6)$$

Then, $K(M)$ is an $n \times n$ matrix of zeros and ones; the (i,j) entry of $K(M)$ is 1 if and only if x_j is stably reachable from x_i in all members of M. Given two $n \times n$ skeleton matrices K and K', write $K \geq K'$ when $K_{ij} \geq K'_{ij}$ for all $i, j = 1, \ldots, n$.

4. MODEL MATCHING.

The solution of the model-matching problem for a single machine is given by the following (compare to Murphy, Geng, and Hammer (2000)).

THEOREM 7. Let $\Sigma = (A,X,f)$ be an asynchronous input/state machine, and let $\Sigma' = (A,X,s')$ be a stable-state input/state machine. The following two statements are equivalent.

(i) There exists a controller C for which the stable-state machine $\Sigma_{c|s}$ induced by the closed loop system Σ_c is equivalent to Σ', where Σ_c is well posed and operates in fundamental mode.

(ii) The skeleton matrices of the machines Σ and Σ' satisfy $K(\Sigma) \geq K(\Sigma')$.

The proof of Theorem 7 is outlined in the Appendix below. The Theorem offers a necessary and sufficient condition for the existence of a state feedback controller that assigns a specified stable behavior to an asynchronous machine Σ. The condition is simple: compare the skeleton matrices of the given system and of the desired model.

Regarding the control of a critical race family, the following statement provides a comprehensive solution (compare to Murphy, Geng, and Hammer (2000)).

THEOREM 8. Let $M = \{\Sigma^1, ..., \Sigma^q\}$ be a critical race family of asynchronous input/state machines with state set X, and let Σ' be a stable-state input/state machine with the same state set X. Let $K(M)$ be the skeleton matrix of M, and let $K(\Sigma')$ be the skeleton matrix of Σ'. Then, the following two statements are equivalent.

(i) There is a controller C for which each one of the stable-state machines $\Sigma^1_{c|s}, \Sigma^2_{c|s}, ..., \Sigma^q_{c|s}$ is equivalent to Σ', where the closed loop systems $\Sigma^1_c, \Sigma^2_c, ..., \Sigma^q_c$ are all well posed and all operate in fundamental mode.

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

Theorem 8 is an elaborate consequence of Theorem 7; a proof is given in Murphy, Geng, and Hammer (2000). In crude terms, the controller C of the Theorem functions by transforming into unstable combinations all undesirable outcomes of the race, while driving all members of the critical race family M to a common stable combination after the race. In this way, the controller equalizes the stable state response of the closed loop system for all members of the family M.

The skeleton matrix $K(M)$ cannot be the zero matrix, since each state must be part of a stable combination. Consequently, it is always possible build a stable-state machine Σ' whose skeleton matrix is equal to $K(M)$. Considering this, Theorem 8 implies that the ambiguity of a critical race can always be eliminated through state feedback control.

5. EXAMPLE.

Consider an asynchronous input/state machine Σ having the input alphabet $A = \{a,b,c\}$ and the state set $X = \{x_0,x_1,x_2\}$. The machine has a critical race at the pair (x_0,c), with two possible outcomes: x_1 and x_2. Here is the table for the recursion function f of Σ.

Σ :

f	a	b	c
x_0	x_0	x_1	$\{x_1, x_2\}$
x_1	x_2	x_1	x_1
x_2	x_2	x_1	x_2

The machine Σ induces a critical race family $M = \{\Sigma^1, \Sigma^2\}$ of two members, whose stable recursion functions s^1 and s^2 are as follows.

Σ^1 $\qquad\qquad$ Σ^2

s^1	a	b	c	s^2	a	b	c
x_0	x_0	x_1	x_1	x_0	x_0	x_1	x_2
x_1	x_2	x_1	x_1	x_1	x_2	x_1	x_1
x_2	x_2	x_1	x_2	x_2	x_2	x_1	x_2

The one-step stable transition matrices are:

$$R(\Sigma^1)=\begin{pmatrix} \{a\} & \{b,c\} & N \\ N & \{b,c\} & \{a\} \\ N & \{b\} & \{a,c\} \end{pmatrix}, R(\Sigma^2)=\begin{pmatrix} \{a\} & \{b\} & \{c\} \\ N & \{b,c\} & \{a\} \\ N & \{b\} & \{a,c\} \end{pmatrix}$$

From these, the one-step skeleton matrices are

$$S(\Sigma^1) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad S(\Sigma^2) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

In view of (5), the skeleton matrices are

$$K(\Sigma^1) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad K(\Sigma^2) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

The skeleton matrix of the family M is by (6):

$$K(M) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

Choose the stable-state machine $\Sigma' := \Sigma^2_{|s}$ as the desired model to match, so that $K(\Sigma') = K(\Sigma^2)$. (This is an arbitrary choice; the matched model does not have to be a member of M.) A brief check shows that $K(M) = K(\Sigma')$. Consequently, the condition of Theorem 8 is satisfied, and a controller C exists which makes the closed loop respond always as Σ'. This controller eliminates the ambiguity caused by the race. An explicit design of the controller C is demonstrated in Murphy, Geng, and Hammer (2000).

APPENDIX

The present section contains a proof of Theorem 7. A more complete theory surrounding the proof is developed in Murphy, Geng, and Hammer (2000). First, the operation (3) of combining skeleton matrices has the following property, which can be verified directly.

PROPOSITION 9. Let A, B, C and D be $n\times n$ skeleton matrices. If $A \geq B$ and $C \geq D$, then $AC \geq BD$. ♦

Now, let $\Sigma' = (A,X,s')$ be a stable-state input/state machine. Problem 2 deals with the existence of a corrective controller C for which

$$\Sigma_{c|s} = \Sigma'. \qquad (10)$$

Let Ξ be the state set of the controller C. Then, the state set of the combination Σ_c is $X\times\Xi$. Define the projection $\pi_x : X\times\Xi \to X : \pi_x(x,\xi) := x$. Letting γ be the recursion function of $\Sigma_{c|s}$, it follows by machine equivalence that (10) is reduces to the following. For every valid pair (x,v) of Σ', there is a state $\xi \in \Xi$ for which (x,ξ,v) is a valid pair of $\Sigma_{c|s}$, and

$$\pi_x\gamma(x,\xi,v) = s'(x,v). \qquad (11)$$

The next result is the basis of Theorem 7 ($\beta \setminus \alpha$ denotes the difference of the sets α, β, i.e., the set of all elements of β that are not in α).

PROPOSITION 12. Let $\Sigma_{|s} = (A,X,s)$ be the stable-state machine induced by an input/state machine $\Sigma = (A,X,f)$. Choose k disjoint sets $x_1\times U_1$, $x_2\times U_2$, ..., $x_k\times U_k$ of valid pairs of Σ, where $x_1, ..., x_k \in X$ are states and $U_1, ..., U_k \subset A$ are sets of input characters. For each $i \in \{1, ..., k\}$, let $x_i' \in X$ be a state stably reachable by Σ from the state x_i. Then, there is a controller C for which $\Sigma_{c|s}$ is equivalent to a stable-state machine $\Sigma' = (A,X,s')$ whose recursion function s' satisfies

$s'[x_i\times U_i] = x_i'$ for all $i = 1,\ldots,k$, and
$s'(z,t) = s(z,t)$ for all $(z,t) \in X\times A \setminus (\bigcup_{i=1, ..., k} x_i\times U_i)$.

Furthermore, the closed loop system Σ_c is well posed and operates in fundamental mode.

Proof. Since x_i' is stably reachable from x_i, there is an input string $w_i \in A^+$ such that $s(x_i,w_i) = x_i'$, $i = 1, ..., k$. Let $m(i) := |w_i|$ be the length of the string and let $v_i^0, v_i^1, \ldots, v_i^{m(i)-1}$ be its characters, so that $w_i := v_i^0 v_i^1 \ldots v_i^{m(i)-1}$. With this input string, s generates a list of states as follows.

$$x_i^1 := s(x_i,v_i^0),$$
$$x_i^2 := s(x_i^1,v_i^1),\ldots,$$
$$x_i^{m(i)-1} := s(x_i^{m(i)-2},v_i^{m(i)-2})$$
$$x_i' := s(x_i^{m(i)-1}, v_i^{m(i)-1}),\ i = 1, \ldots, k.$$

Now, let $U(x_i) \subset A$ be the set of all input characters that form stable combinations with the state x_i. Define the sets

$$S := \bigcup_{i=1, ..., k} x_i\times U(x_i),$$
$$V := \bigcup_{i=1, ..., k} x_i\times U_i.$$

A controller $C = (A\times X,A,\Xi,\phi,\eta)$ that satisfies the requirements of Proposition 12 can then be constructed as follows.

(i) The state set Ξ of C has $2 + \Sigma_{i=1}^k m(i)$ states, denoted by

$$\Xi = \{\xi_0, \xi_1, \xi_1^1, \xi_1^2, ..., \xi_1^{m(1)}, \xi_2^1, ..., \xi_2^{m(2)}, \xi_k^1, ..., \xi_k^{m(k)}\}.$$

(ii) The initial state of the controller C is ξ_0. The controller moves to the state ξ_1 upon detection of a stable combination with one of the states $x_1, ... x_k$. This step comes to guarantee fundamental mode operation, since the system Σ must be in a stable combination with the state x_i before starting its move toward the state x_i' to match the required stable transition. To satisfy this requirement, the recursion function ϕ of C is defined as follows.

$$\phi(\xi_0,(z,t)) := \xi_0 \text{ for all } (z,t) \in X\times A \setminus S,$$
$$\phi(\xi_0,(x,u)) := \xi_1 \text{ for all } (x,u) \in S.$$

While in the state ξ_0, the controller C is transparent; it applies to Σ its own external input. Consequently, the output function η of the controller is defined at this state by

$$\eta(\xi_0,(z,t)) := t \text{ for all } (z,t) \in X\times A.$$

To define the function η at the state ξ_1, choose a character $u_i \in U(x_i)$, and set

$$\eta(\xi_1,(x_i,t)) := u_i \text{ for all } t \in A,\ i = 1, ..., k.$$

In this way, Σ remains in a stable combination with the state x_i as long as the controller is in the state ξ_1. This definition preserves fundamental mode operation, since the system Σ is in a stable combination when its input is set to u_i.

(iii) Suppose that the system Σ is in a stable combination with the state x_i, and an input value $u \in U_i$ appears. Then, C will start to apply to Σ the input string w_i, to initiate the process of bringing Σ to the state x_i'. To achieve this objective, define

$$\phi(\xi_1,(x_i,u)) := \xi_i^1 \text{ for all } u \in U_i,\ i = 1, ..., k;$$
$$\phi(\xi_1,(x_i,u)) := \xi_1 \text{ for all } u \in U(x_i) \setminus U_i,\ i = 1, ..., k;$$
$$\phi(\xi_1,(z,t)) := \xi_0 \text{ for all pairs } (z,t) \in X\times A \setminus (S \cup V).$$

When reaching the state ξ_i^1, the controller begins to generate the input string w_i for Σ, so set

$$\eta(\xi_i^1,(z,t)) = v_i^0 \text{ for all } (z,t) \in X\times A,\ i = 1, ..., k.$$

When this input is applied to Σ, it moves to the state x_i^1, where (x_i^1,v_i^0) is a stable combination by the definition of x_i^1.

(iv) From here on, the controller continues to generate the string w_i as input to Σ, as follows.

$$\phi(\xi_i^j,(x_i^j,u)) := \xi_i^{j+1} \text{ for all } u \in U_i,$$
$$\phi(\xi_i^j,(z,t)) := \xi_i^j \text{ for all } (z,t) \in X\times A \setminus x_i^j\times U_i,\ j = 1, 2, ..., m(i)–1,\ i = 1, ..., k.$$

The controller state ξ_i^{j+1} signifies that Σ has reached the state x_i^j, and is ready for the next character v_i^j of the input string w_i. Consequently, for $j = 1, 2, ..., m(i)–1$, set

$$\eta(\xi_i^{j+1},(z,t)) := v_i^j \text{ for all } (z,t) \in X\times A,\ i = 1, ..., k.$$

(v) Finally, for $j = m(i)$, assign

$$\phi(\xi_i^{m(i)},(x_i',u)) := \xi_i^{m(i)} \text{ for } u \in U_i,$$
$$\phi(\xi_i^{m(i)},(z,t)) := \xi_0 \text{ for all } (z,t) \in X\times A \setminus S,\ i = 1, ..., k.$$

$$\phi(\xi_i^{m(i)},(z,t)) := \xi_1 \text{ for } (z,t) \in S.$$

Since the sets $x_1 \times U_1$, $x_2 \times U_2$, ..., $x_k \times U_k$ are disjoint, there is no inconsistency in the definition of the recursion function ϕ.

A direct calculation shows that with this controller C, the stable recursion function γ of the closed loop system Σ_c satisfies (11). As pointed out during the construction in this proof, the closed loop system operates in fundamental mode. Finally, since Σ is an input/state system, it is strictly causal, and the closed loop system Σ_c is well posed (see Hammer (1996a) for details). ♦

Proof (of Theorem 7). Let s and s' be the stable recursion functions of Σ and of Σ', respectively, and let γ be the stable recursion function of Σ_c. Let $X = \{x_1, ..., x_n\}$ be the state set of Σ, and let Ξ be the state set of the controller C. Then, the state set of the controlled system Σ_C is $X \times \Xi$. Define the function

$$s_c := \pi_x \gamma.$$

Assume first that condition (i) of Theorem 7 is valid, i.e., that there is a controller C such that $\Sigma_{c|s}$ is equivalent to Σ'. Then, for every valid pair $(x,u) \in X \times A$ of Σ', there is a state $\xi \in \Xi$ such that $s_c(x,\xi,u) = s'(x,u)$. Now, x and $s_c(x,\xi,u)$ are states of Σ, say $x = x_i \in X$ and $s_c(x,u) = x_k \in X$. This means that one can write $x_k = s_c(x_i,\xi,u) = s'(x_i,u)$. Letting $S(\Sigma')$ be the one-step skeleton matrix of Σ', the equality $x_k = s'(x_i,u)$ implies that $S_{ik}(\Sigma') = 1$.

Further, since the controller C can access Σ only through the input of Σ, the equality $x_k = s_c(x_i,\xi,u)$ means that there is an input string $w \in A^+$ such that $x_k = s(x_i,w)$. By the definition of the skeleton matrix $K(\Sigma)$ of Σ, the last equality implies that $K_{ik}(\Sigma) = 1$. Thus, $K_{ik}(\Sigma) = 1$ if $S_{ik}(\Sigma') = 1$. In symbolic form, this means that $K(\Sigma) \geq S(\Sigma')$. Multiplying each side of this inequality by itself n–1 times and invoking Proposition 9, it follows that $K^{(n-1)}(\Sigma) \geq S^{(n-1)}(\Sigma') = K(\Sigma')$. In view of (4), it follows that $K(\Sigma) \geq K(\Sigma')$, which proves that condition (i) of Theorem 7 implies condition (ii) of Theorem 7.

Conversely, assume that condition (ii) of Theorem 7 is valid, i.e., that $K(\Sigma) \geq K(\Sigma')$. Now, by the definition of the skeleton matrix, one always has $K(\Sigma') \geq S(\Sigma')$. Combining the last two inequalities, it follows that $K(\Sigma) \geq S(\Sigma')$. This implies that, for every valid pair (x,u) of Σ', the state s'(x,u) is stably reachable from x by Σ. But then, Proposition 12 guaranties the existence of a controller C for which $\Sigma_{c|s} = \Sigma'$. Thus, condition (ii) implies condition (i), and our proof concludes. ♦

REFERENCES

Barrett, T., and LaFortune, S., (1997). Using bisimulation to solve discrete event control problems, *Proceedings of the American Control Conference*, pp. 2337-2341.

Chu, T., (1994). Synthesis of hazard-free control circuits from asynchronous finite state machines specifications, *Journal of VLSI Signal Processing*, **vol.7**, no.1-2, p. 61-84.

Datta, P.K., Bandyopadhyay, S.K. and Choudhury, A.K., (1988). A graph theoretic approach for state assignment of asynchronous sequential machines, *International Journal of Electronics*, **vol.65, no.6**, p. 1067-75.

DiBenedetto, M.D., Saldanha, A., and Sangiovanni-Vencentelli, A., (1994). Model matching for finite state machines, *Proceedings of the IEEE Conference on Decision and Control*, pp. 3117-3124; (1995) Strong model matching for finite state machines with non-deterministic reference model, *Proceedings of the IEEE Conference on Decision and Control*, pp. 422-426.

Eilenberg, S. (1974) *Automata, languages, and machines*, Academic Press, NY..

Hammer, J., (1994). On some control problems in molecular biology, *Proceedings of the IEEE Conference on Decision and Control*, December 1994; (1995) On the modeling and control of biological signaling chains, *Proceedings of the IEEE Conference on Decision and Control*, December 1995; (1996a) On corrective control of sequential machines, *International Journal of Control*, **Vol. 65**, pp. 249-276; (1996b) On the control of incompletely described sequential machines, *International Journal of Control*, **Vol. 63, No. 6**, pp. 1005-1028.

Huffman, D.A., (1954a) The Synthesis of Sequential Switching Circuits, *J. Franklin Inst.*, **vol. 257**, pp. 161-190; (1954b) The Synthesis of Sequential Switching Circuits, *J. Franklin Inst.*, **vol. 257**, pp. 275-303; (1957) The Design and Use of Hazard-Free Switching Networks, *Journal of the Association of Computing Machinery*, **vol. 4**, no. 1, pp. 47-62.

Kohavi, Z., (1970) *Switching and Finite Automata Theory*, McGraw-Hill Book Company, New York.

Lavagno, L., Moon, C. W., and Sangiovanni-Vincentelli, A., (1995). Efficient heuristic procedure for solving the state assignment problem for event-based specifications, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **vol. 14, no. 7**, p. 45-60.

Maki, G., and Tracey, J., (1971). A State Assignment Procedure for Asynchronous Sequential Circuits, *IEEE Transactions on Computers*, **C-20**, pp. 666-668.

Murphy, T.E., Geng, X.J., and Hammer, J., (2000). On the control of asynchronous machines with races, submitted for publication.

Ozveren, C.M., Willsky, A.S., and Antsaklis, P.J., (1991). Stability and stabilizability of discrete event systems, *J. ACM*, **Vol. 38**, pp. 730-752.

Ramadge, P.J.G., and Wonham, W.M., (1987). Supervisory Control of a Class of Discrete Event Processes, *SIAM Journal of Control and Optimization*, **vol. 25, no. 1**, pp. 206 - 230, Jan. 1987.

Thistle, J. G. and Wonham, W.M., (1994). Control of infinite behavior of finite automata, *SIAM Journal on Control and Optimization*, **v 32, n 4,** p 1075-1097.

Unger, S. H., (1995). Hazards, critical races, and metastability, *IEEE Trans. on Computers*, **vol. 44, no. 6**, p 754-768.