2013 European Control Conference (ECC)
July 17-19, 2013, Zürich, Switzerland.

Model Matching Control for Composite Asynchronous Sequential Machines*

Jung–Min Yang[1], Seong–Jin Park[2], and Seong–Woo Kwak[3]

***Abstract*— Model matching for an asynchronous sequential machine is to design a feedback controller that compensates the closed-loop system so that it can match the behavior of a reference model. The objective of this paper is to address the model matching problem for a composite asynchronous sequential machine. Two input/state asynchronous machines in cascade connection form a composite machine, where the output of the front machine is transmitted as the input to the rear machine. Given a desired model, we present the existence condition and design procedure for an appropriate corrective controller that achieves model matching. An illustrative example is provided for verifying the applicability of the proposed scheme.**

I. INTRODUCTION

In the field of control systems engineering, model matching is referred to as designing a compensator or controller for a given system so that the response of the closed-loop system matches that of a prescribed reference model. For the past decades, extensive studies have been devoted to solving the model matching problem for continuous-time (e.g., [1], [2]) and discrete-event systems (e.g., [3], [4]).

The present work addresses the model matching problem for asynchronous sequential machines. Asynchronous sequential machines represent digital hardware or software programs that are operated without a global synchronizing clock. They are still being used as an important building block in many areas such as high-speed digital systems, parallel computing environments, digital communications, etc [5].

The research on model matching for asynchronous machines was initiated in [6] and later extended in [7]–[9]. The underlying approach is based on *corrective control* [10], a novel control scheme that applies automatic control theory to dynamics of asynchronous machines. A feedback controller, also having the form of an asynchronous machine, is placed in front of the considered asynchronous machine so as to compensate the behavior of the closed-loop system, while keeping the inner logic of the machine intact. The controller takes effect only when the machine starts to produce undesirable outcomes or model mismatch; otherwise, it simply passes external inputs to the machine. Corrective control fully utilizes the unique feature of asynchronous machines that their transient behavior is not noticeable from an outer user's viewpoint. In corrective control, hence, one can assert that model matching is realized if the *stable-state behavior* of the closed-loop system matches the input/output function of the model.

The objective of this paper is to develop a corrective control scheme for achieving model matching for composite asynchronous machines. The considered asynchronous machine consists of two single asynchronous machines—front and rear machines—that are connected with each other as cascade composition, where the output of the front machine serves as the input to the rear machine. The overall system also complies with asynchronous mechanism.

Compared to the former studies [6]–[9], our work requires a novel analysis tool to describe reachability and controllability for the following reasons. First, since the composite machine comprises two single asynchronous machines, its state has two-dimensional variables. In accordance with this feature, we should develop crucial properties needed for controller design such as stable reachability and intermediate states in the correction trajectory. Next, the controller does not have direct access to the rear machine because the rear machine is operated by the input made by the front machine. Hence, constraint on feasibility of a control input sequence will be more strict than the case of controlling a single asynchronous machine.

We first present the condition for the existence of an appropriate corrective controller that solves the model matching problem with respect to a give model. The controller existence depends on the potential reachability of the composite machine that can be utilized in the correction procedure. We also outline the design procedure of a controller module in which the interaction between the controller and the machine will be specified. An illustrative example is provided for verifying the applicability of the proposed scheme.

II. PRELIMINARIES

A. Model Matching Control System

Fig. 1 shows the configuration of the model matching control system used in this paper. Σ_1 and Σ_2 are two finite-state asynchronous machines that make cascade composition

*The work of J.–M. Yang was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012R1A2A2A01003419). The research of S.–J. Park 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. 2011-0007830). The research of S. W. Kwak 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. 2010-0021366).

[1]J.–M. Yang is with the Department of Electrical Engineering, Catholic University of Daegu, Gyeongsan, Gyeongbuk, 712-702, Republic of Korea. E-mail: jmyang@cu.ac.kr.

[2]S.–J. Park is with the Department of Electrical and Computer Engineering, Ajou University, Suwon, 443-749, Republic of Korea. E-mail: parksjin@ajou.ac.kr.

[3]S. W. Kwak is with the Department of Electronic Engineering, Keimyung University, Daegu, 704-701, Republic of Korea. E-mail:ksw@kmu.ac.kr (corresponding author).

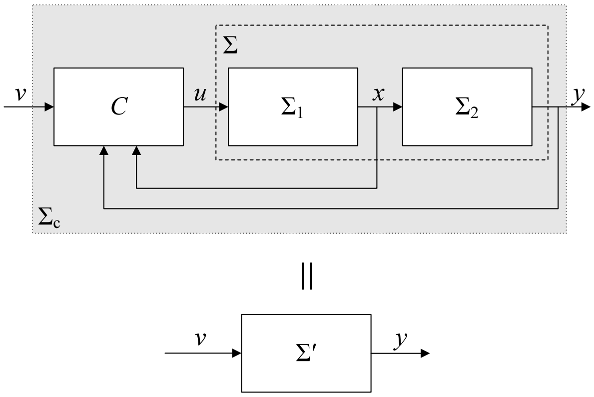

Fig. 1. Model matching control system for cascaded composition of two finite-state asynchronous machines.

with each other. We assume that both machines are of input/state type, that is, the current state of the machine is given as the output value. The composite asynchronous machine Σ is the input/ouptut machine, of which input and output are the same as that of Σ_1 and of Σ_2, respectively. C is the corrective controller that realizes model matching. C also has the form of an asynchronous machine. v is the external input, u is the input to Σ_1, x is the state (or output) of Σ_1 that is the input to Σ_2 as well, and y is the state (or output) of Σ_2. Both x and y are delivered to C as the feedback. Σ_c denotes the closed-loop system of C, Σ_1 and Σ_2.

The main objective is to present the existence condition and design procedure for a corrective controller C for which the stable-state behavior of the closed-loop system Σ_c matches that of a prescribed model Σ'. Here *matching behavior* means that Σ_c and Σ' show identical input/output functioning in terms of stable states [6], [9], i.e., when Σ_c and Σ' staying at the same stable state receive an external input v respectively, both move to a next state giving out the same output y (see Fig. 1).

When the system Σ shows the identical input/ouput behavior as that of Σ', the controller C conducts no control action; it just relays the external input v to the control input u. On the other hand, when an external input that will cause model mismatch is about to enter into the system, C suppresses the input value and generates a series of control inputs for which the behavior of the closed-loop system Σ_c is compensated to have a desired behavior. Note that this control activity, or corrective control as is often termed [10], [11], is made possible by virtue of the characteristics of asynchronous machines that their transient behavior is so fast that it is not noticeable from an outer user's viewpoint.

When working with asynchronous machines, it is important to abide by *fundamental mode operation* [12], an operating policy that prohibits the simultaneous change of two or more variables. This policy helps to prevent uncertainties arising from simultaneous changes in two or more variables. For the closed-loop system Σ_c of Fig. 1, fundamental mode operation implies the following (adapted from [13, Fact 1.1]).

Condition 1: The closed-loop system Σ_c of Fig. 1 operates in fundamental mode when the following conditions are valid:

(i) Among C, Σ_1, and Σ_2, whenever one machine is in transition, the other two are at stable states.

(ii) The external input v changes only when all of C, Σ_1, and Σ_2 are in stable states.

Condition 1.(i) must be implemented during the design of the controller C and machines Σ_1 and Σ_2, e.g., using the Huffman style asynchronous machines [5]. Condition 1.(ii), on the other hand, is a restriction on the operation of the closed-loop system Σ_c. Nevertheless, as transitions of asynchronous machines occur very quickly, 1.(ii) does not impose a burdensome requirement.

B. Composite Asynchronous Machine

First, let us represent the input/state asynchronous machine Σ_1 by the quadruple

$$\Sigma_1 := (A, X, x_0, f)$$

where A is the input set, X is the state set, $x_0 \in X$ is the initial state, and $f : X \times A \to X$ is the state transition function. A valid state–input pair $(x, u) \in X \times A$ is a stable combination of Σ_1 if $f(x, u) = x$ and x is a stable state with u; otherwise, it is a transient combination and x is a transient state with u. A transient combination induces a chain of transient transitions $x_1 := f(x, u)$, $x_2 := f(x_1, u), \ldots$, until Σ_1 reaches the *next stable state* $x_k = f(x_k, u)$ [12]. If there is no integer k for which (x_k, u) is a stable combination, then the system contains an infinite cycle. The present discussion assumes that none of the machines under considerations have infinite cycles. Owing to the lack of a synchronizing clock, transient transitions between (x, u) and (x_k, u) are executed instantaneously (in zero time, ideally). As a result, from an outer environment, Σ_1 seems to move directly from (x, u) to (x_k, u). The *stable recursion function* $s_1 : X \times A \to X$ [12] elucidates this feature of asynchronous machines:

$$s_1(x, u) := x_k$$

where x_k is the next stable state of (x, u). A chain of transitions from one stable combination to another, as described by s, is called a *stable transition*.

Similarly, Σ_2 is described as

$$\Sigma_2 := (X, Y, y_0, g).$$

Since the state of Σ_1 is transmitted as the input to Σ_2, X serves as the input set of Σ_2. Y is the state set of Σ_2, $y_0 \in Y$ is the initial state, and $g : Y \times X \to Y$ is the state transition function. $s_2 : Y \times X \to Y$ is the corresponding stable recursion function of Σ_2. It is convenient to extend s_1 and s_2 from input characters to sequences recursively. For $x \in X$ and $u_1 u_2 \cdots u_k \in A^+$, where A^+ is the set of all non-empty strings of characters of A, we define

$$s_1(x, u_1 u_2 \cdots u_k) := s_1(s_1(x, u_1), u_2 \cdots u_k).$$

Likewise, For $y \in Y$ and $x_1 x_2 \cdots x_k \in X$,

$$s_2(y, x_1 x_2 \cdots x_k) := s_2(s_2(y, x_1), x_2 \cdots x_k).$$

Cascade composition Σ is represented by an input/output asynchronous machine with the formulation [14]

$$\Sigma := (A, Y, X \times Y, (x_0, y_0), \delta, h)$$

where A is the input set, Y is the output set, $X \times Y$ is the two-dimensional state set, (x_0, y_0) is the initial state, $\delta : X \times Y \times A \to X \times Y$ is the state transition function, and $h : X \times Y \to Y$ is the output function (assuming that Σ is a Moore machine). Both δ and h are partial functions. Stable and transient combinations of Σ are defined in the same manner as the former case. A valid state–input pair $((x, y), u) \in X \times Y \times A$ is a stable combination if $\delta((x, y), u) = (x, y)$; otherwise, it is a transient combination.

To describe the transient behavior of Σ, assume that Σ has been at a stable combination (x_k, y_k) when the external input v changes to u_k for which $\delta((x_k, y_k), u_k) \neq (x_k, y_k)$. Assuming that no control action is done by C in Fig. 1, that is, by setting $u = v$, Σ_1 first goes through the stable transition in response to the input u_k, namely, $f(x_k, u_k) = x_k^1$, $f(x_k^1, u_k) = x_k^2, \ldots,$ until reaching the next stable state $x_{k+1} = s_1(x_k, u_k)$. Subsequently, Σ_2 undergoes its transitions by receiving the current state of Σ_1 as the input character. Note that according to Condition 1, Σ_2 must not initiate its transition until Σ_1 falls into a stable combination for preserving the principle of fundamental mode operation. This means that the transition of Σ_2 at the state y_k should not be defined with respect to all the transient states $x_k^1, x_k^2, \ldots,$ except for the next stable state x_{k+1}, which drives Σ_2 to its next stable state $y_{k+1} = s_2(y_k, x_{k+1})$. In mathematical terms, the stable recursion function $s : X \times Y \times A \to X \times Y$ of the composite machine Σ operates according to the recursion

$$\begin{aligned} s((x_k, y_k), u_k) &= (x_{k+1}, y_{k+1}) \\ &:= (s_1(x_k, u_k), s_2(y_k, s_1(x_k, u_k))) \end{aligned}$$

provided that none of the transient states between (x_k, u_k) and (x_{k+1}, u_k) make a valid pair of Σ_2 with the state y_k. Here, the integer k represents the "step counter" of Σ; it advances by one upon a change of the machine's input or state.

Since the output of Σ is equal to the current state of the machine Σ_2, the output function h is defined as the projection onto Y as follows.

$$h(x_k, y_k) := \Pi_y(x_k, y_k) = y_k.$$

Stable reachability between states is a crucial property required for materializing a corrective controller for model matching. In the machine Σ_1, a state $x' \in X$ is said to be stably reachable from another state $x \in X$ if there exists an input sequence $t \in A^+$ such that $x' = s_1(x, t)$ [6]. In the machine Σ_2, similarly, stable reachability from a state $y \in Y$ to another state $y' \in Y$ is equal to the existence of an input sequence $w \in X^+$ such that $y' = s_2(y, w)$. Extending this notion, we introduce the following definition of stable reachability for the composite machine Σ.

Definition 1: In the composite machine Σ, a state $(x', y') \in X \times Y$ is said to be stably reachable from another state $(x, y) \in X \times Y$ if there exists an input sequence $t = u_1 u_2 \cdots u_k \in A^+$ such that $s_1(x, t) = x'$ with $x_1 = s_1(x, u_1)$, $x_2 = s_1(x_1, u_2), \ldots, x' = s_1(x_{k-1}, u_k)$, and $s_2(y, w) = y'$ where $w = x_1 x_2 \cdots x'$.

The above definition implies that not only does the input string t steer Σ_1 from x to x', but the sequence of the induced stable states w of Σ_1 serves as the input sequence that drives Σ_2 from y to y'. Stable transitions of each machine must hold Condition 1 for guaranteeing fundamental mode operation.

III. MODEL MATCHING CONTROLLER

A. Existence Condition

Two asynchronous machines are *stably equivalent* [12] if their stable-state behaviors have equivalent functionality and seem identical from an outer user. Referring to Fig. 1, model matching between the machine Σ and the model Σ' means that the closed-loop system Σ_c is controlled by C so that its behavior can be stably equivalent to the model Σ' [6].

Without loss of generality, the prescribed input/state model Σ' is given as a stable-state machine, namely

$$\Sigma' := (A, Y, y_0, s')$$

where A, Y, and $y_0 \in Y$ correspond with those of Σ_1 and Σ_2, and $s' : Y \times A \to Y$ is the recursion function.

To explain the notion of model matching in detail, assume that the composite machine Σ stays at a stable combination (x_i, y_i) with the output $h(x_i, y_i) = y_i$, and that the model Σ' is at a stable combination with the state y_i. Assume further that the external input changes to a character $a \in A$ and enters into Σ and Σ'. Let $y_i^m := s'(y_i, a)$ be the next stable state of Σ'. If $s((x_i, y_i), a) = (x_i', y_i^m)$ with $h(x_i', y_i^m) = y_i^m$, i.e., if the rear machine Σ_2 transfers to the state y_i^m in response to the input a, the stable-state behavior of Σ continues to match that of Σ'. On the other hand, if the output of Σ is not equal to y_i^m, model matching would be violated if not corrected.

In this paper, we describe model mismatch between Σ and Σ' by the following set $D \subset X \times Y \times P(A) \times Y$, where $P(A)$ is the power set of A.

$$D := \{(x_i, y_i, A_i, y_i^m) | 1 \leq i \leq q\}.$$

$(x_i, y_i, A_i, y_i^m) \in D$ means that the model Σ' has the stable-state behavior $s'(y_i, a) = y_i^m$ for all $a \in A_i$, whereas the machine Σ staying at the stable state (x_i, y_i) transfers to a state (x', y') such that $y' \neq y_k^m$, in response to the input a. Note that for some i and j with $i \neq j$, y_i and y_j are not necessarily different with each other; in that case, either $x_i \neq x_j$ or $A_i \cap A_j = \emptyset$.

In the corrective control scheme, a basic controller module is designed to solve each model mismatch, and the overall controller is completed by assembling all the basic modules. Let C_i be the basic controller module that solves the model mismatch (x_i, y_i, A_i, y_i^m), $i = 1, \ldots, q$. The overall controller C is obtained by *join* operation "$\vee$" addressed in [8]:

$$C := C_1 \vee C_2 \vee \cdots \vee C_q.$$

In the case of a single asynchronous sequential machine, the existence of a model matching controller depends on whether the considered machine has stable reachability large enough for the machine to reach a desired state from a beginning state that would cause model mismatch [6], [11]. In terms of (x_i, y_i, A_i, y_i^m), the latter condition is said that y_i^m should be stably reachable from y_i. However, the present study differs from the former results in that the controlled machine is cascade composition of two asynchronous machines and thus the state of the desired model, i.e., $y \in Y$, is an element of the machine's two-dimensional state value $(x, y) \in X \times Y$.

Assume again that the composite machine Σ has been staying at the stable state (x_i, y_i), i.e., Σ_1 is at x_i and Σ_2 is at y_i, when the external input changes to an input character $a \in A_i$. Recalling that model matching is achieved if the closed-loop system Σ_c shows the same input/output behavior as the model, it follows that the control objective is to drive the machine Σ to a next stable state (x_i', y_i^m). In the framework of corrective control, the correction behavior is executed so fast that the closed-loop system Σ_c would seem to move from (x_i, y_i) directly to (x', y_i^m) in response to the input a.

Like the case of a single asynchronous machine, the existence condition for a corrective controller C is that (x_i', y_i^m) is stably reachable from (x_i, y_i) in the composite machine Σ. Using Definition 1, we describe the condition as follows:

Condition 2: The existence condition for a corrective controller that realizes model matching between Σ and Σ' with $D = \{(x_i, y_i, A_i, y_i^m) | 1 \le i \le q\}$:

$\forall i = 1, \ldots, q$, $\exists t_i = u_1 u_2 \cdots u_k \in A^+$ such that $s_1(x_i, t_i) = x_i'$ and $s_2(y_i, w_i) = y_i^m$ where $w_i = x^1 \cdots x^{k-1} x_i'$, $x^1 = s_1(x_i, u_1)$, $x^2 = s_1(x^1, u_2) \cdots x_i' = s_1(x^{k-1}, u_k)$.

B. Controller Design

Referring to Fig. 1, the corrective controller module C_i has three inputs $(x, y, v) \in X \times Y \times A$ and one output $u \in A$. Thus we describe C_i as a deterministic input/output asynchronous machine of the form

$$C_i := (X \times Y \times A, A, \Xi, \xi_0, \phi, \eta)$$

where Ξ is the state set, $\xi_0 \in \Xi$ is the initial state, $\phi : \Xi \times X \times Y \times A \to \Xi$ is the recursion function, and $\eta : \Xi \to A$ is the output function.

Provided that the machine Σ satisfies Condition 2, we now construct the controller module C_i for the model mismatch $(x_i, y_i, A_i, y_i^m) \in D$. In the beginning, C_i is at the initial state ξ_0. C_i remains in the initial state until detecting that Σ reaches the stable combination (x_i, y_i). Then C_i moves to a state ξ_t, called the transition state. Note that in fundamental mode operation, an input change can occur only when the machine stays at a stable combination. Hence, by transferring to the state ξ_t, C_i anticipates a possible entrance of an input character of A_i. If the external input changes to $v \notin A_i$ that causes no model mismatch, the controller module C_i returns to the initial state ξ_0. No particular control action is executed at ξ_0 and ξ_t. The function of C_i is to relay the external input v to the control input u without modifying it. To this end, we set the recursion function ϕ and the output function η as follows.

$$\begin{aligned}
&\phi(\xi_0, (x, y, v)) = \xi_0 \\
&\qquad \forall (x, y, v) \in X \times Y \times A \setminus \{(x_i, y_i)\} \times U(x_i, y_i) \\
&\phi(\xi_0, (x, y, v)) = \xi_t \quad (x, y, v) \in \{(x_i, y_i)\} \times U(x_i, y_i) \\
&\eta(\xi_0, (x, y, v)) = v \quad \forall (x, y, v) \in X \times Y \times A \\
&\eta(\xi_t, (x, y, v)) = v \quad \forall (x, y, v) \in X \times Y \times A
\end{aligned}$$

where $U(x_i, y_i) \subset A$ denotes the set of input characters that make a stable combination of Σ with the state (x_i, y_i). In view of Σ_1 and Σ_2, for every $u \in U(x_i, y_i)$, $s_1(x_i, u) = x_i$ and $s_2(y_i, x_i) = y_i$.

When the external input v switches to $a \in A_i$ at ξ_t, it would breach model matching if delivered to Σ. Hence C_i suppresses a and instead generates a series of control input characters. By Condition 2, there exists an input sequence

$$t_i := u_1 u_2 \cdots u_k \in A^+ \text{ s.t. } s((x_i, y_i), t_i) = (x_i', y_i^m).$$

C_i uses t_i as its control input sequence. Following the notations of Condition 2, we denote by $x^1, \ldots, x^{k-1} \in X$ all the intermediate stable states Σ_1 passes through with t_i, that is,

$$\begin{aligned}
&x^j = s_1(x^{j-1}, u_j) \\
&x^j = s_1(x^j, u_j), \ j = 1, \ldots, k \\
&x^0 := x_i, \ x^k := x_i'.
\end{aligned}$$

Moreover, denote by $y^1, \ldots, y^{k-1} \in Y$ the intermediate stable states Σ_2 passes through with the induced input sequence $w_i := x^1 x^2 \cdots x^k$ where $x^k = x_i'$, i.e.,

$$\begin{aligned}
&y^j = s_2(y^{j-1}, x^j) \\
&y^j = s_2(y^j, x^j), \ j = 1, \ldots, k \\
&y^0 := y_i, \ y^k := y_i^m.
\end{aligned}$$

Note that all (x^j, u_j)'s and (y^i, x^i)'s are stable combinations of Σ_1 and Σ_2, respectively. Σ_1 and Σ_2 may progress by some transient states between the adjacent stable states. A remarkable property of the corrective controller is that the controller is devoted to making these stable combinations seem transient in the closed-loop system by inserting k auxiliary states of the controller, termed $\xi_1, \ldots, \xi_k \in \Xi$, into the correction trajectory. Upon receiving the external input a, C_i moves to ξ_1 and provides Σ with the first control input character u_1. The front machine Σ_1 moves from x_i to $x^1 = s_1(x_i, u_1)$, the first intermediate state, generating the output x^1. Receiving x^1, the rear machine Σ_2 moves to its first intermediate state $y^1 = s_2(y_i, x^1)$. In dynamics of the composite machine Σ, the latter equals that Σ reaches the stable state (x^1, y^1). As soon as observing that the feedback value changes to (x^1, y^1), C_1 transfers to ξ_2 and produces the second control input u_2, and so on. This procedure advances forward until Σ reaches the desired state (x_i', y_i^m). To implement this control operation, we set ϕ and η as follows.

$$\begin{aligned}
&\phi(\xi_t, (x_i, y_i, a)) = \xi_1 \quad \forall a \in A_i \\
&\phi(\xi_t, (x_i, y_i, v)) = \xi_t \quad \forall v \in U(x_i, y_i) \setminus A_i \\
&\phi(\xi_t, (x_i, y_i, v)) = \xi_0 \quad \forall v \in A \setminus (U(x_i, y_i) \cup A_i).
\end{aligned}$$

$$\begin{aligned}
&\phi(\xi_j, (x^j, y^j, a)) = \xi_{j+1} \quad \forall j = 1, \dots, k-1 \\
&\phi(\xi_j, (x, y, a)) = \xi_j \quad \forall (x, y) \in X \times Y \setminus \{(x^j, y^j)\} \\
&\eta(\xi_j, (x, y, a)) = u_j \quad \forall (x, y) \in X \times Y.
\end{aligned}$$

When C_i arrives at ξ_k, the machine Σ finally reaches the desired state (x_i', y_i^m):

$$\begin{aligned}
&\phi(\xi_k, (x_i', y_i^m, a)) = \xi_k \\
&\phi(\xi_k, (x, y, v)) = \xi_0 \\
&\qquad \forall (x, v) \in X \times Y \times A \setminus \{(x^{k-1}, y^{k-1}, a), (x_i', y_i^m, a)\} \\
&\eta(\xi_k, (x, y, v)) = u_k \quad \forall (x, y, v) \in X \times Y \times A.
\end{aligned}$$

In this way, the controller module C_i keeps Σ at the stable combination $((x_i', y_i^m), u_k)$ so long as the external input remains a. When the external input switches from a to another character $v \notin A_i$, C_i resets to its initial state ξ_0 as assigned in the second line of the above equations. Clearly, all these assignments of ϕ and η preserve the principle of fundamental mode operation. Fig. 2 illustrates the interaction between the controller module C_i, Σ_1, and Σ_2.

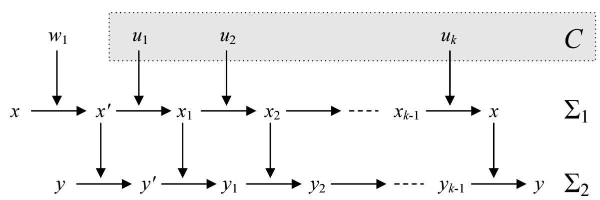

Fig. 2. Interaction between C_i, Σ_1, and Σ_2.

IV. EXAMPLE

Consider two input/state asynchronous machines Σ_1 and Σ_2 of which state flow diagrams are shown in Figs. 3 and 4. Their input and states sets are

$$\begin{aligned}
&A = \{a, b, c, d\} \\
&X = \{x_1, x_2, x_3, x_4\}, \ x_0 := x_1 \\
&Y = \{y_1, y_2, y_3, y_4\}, \ y_0 := y_1.
\end{aligned}$$

Applying the mechanism of cascade connection, we construct the composite asynchronous machine as illustrated in Fig. 5. For explaining dynamics of Σ, consider the initial state (x_1, y_1), that is, the state where the front machine Σ_1 staying at the stable combination (x_0, c) and the rear machine Σ_2 at (y_1, x_1). In Fig. 3, when the external input v changes to a, Σ_1 goes through the stable transition to the next stable state $x_3 = s_1(x_1, a)$ during which it traverses a transient state x_2. Associated with this stable transition, Σ_2 receives the input sequence $x_2 x_3$. Since only x_3 makes a valid pair with the present state y_2, fundamental mode is preserved at this transition. Upon receiving x_3, Σ_2 transfers to the next stable state $y_2 = s_2(y_1, x_3)$, and consequently the composite machine Σ reaches the corresponding stable combination $((x_3, y_2), a)$. Other transitions of Σ are interpreted in a similar way.

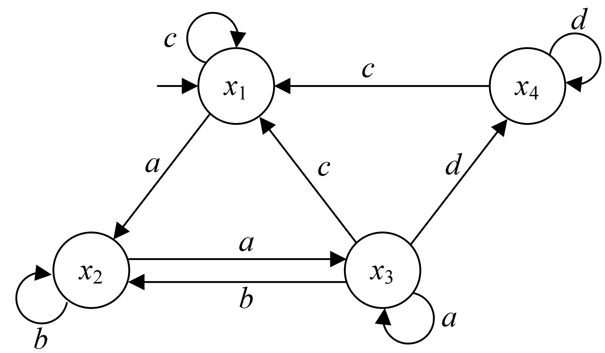

Fig. 3. Front machine Σ_1.

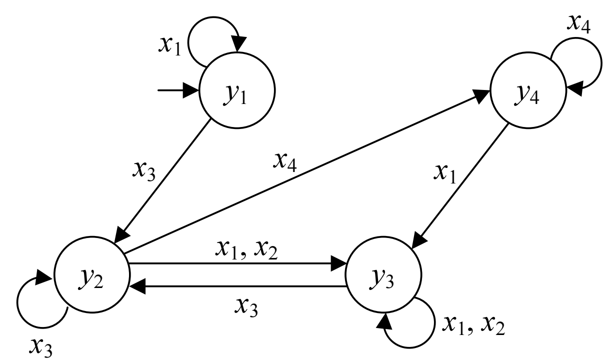

Fig. 4. Rear machine Σ_2.

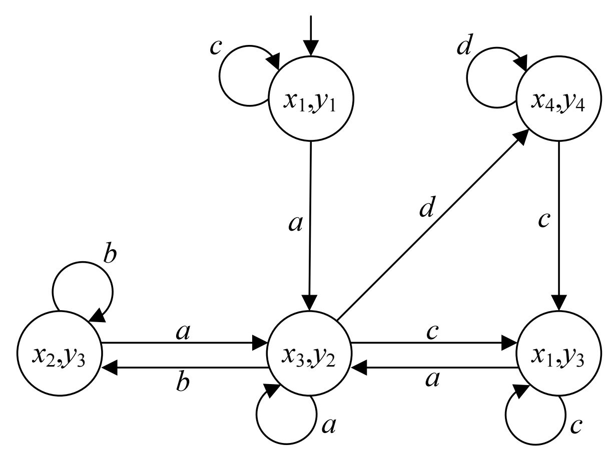

Fig. 5. Composite machine Σ.

The reference model Σ' is given in Fig. 6 where the input and state set are equal to A and Y, respectively. By comparing Fig. 5 and Fig. 6, we derive the following model mismatch:

$$D = \{(x_1, y_3, \{a\}, y_4), (x_2, y_3, \{a\}, y_4)\}.$$

The set D signifies that whereas the model Σ' goes from y_3 to y_4 in response to the input a, the composite machine Σ fails to match the behavior of the model. More specifically, the stable states of Σ with the element y_3 are (x_1, y_3) and (x_2, y_3), and from both states Σ reaches (x_3, y_2) that gives the incorrect output y_2 (see Fig. 5).

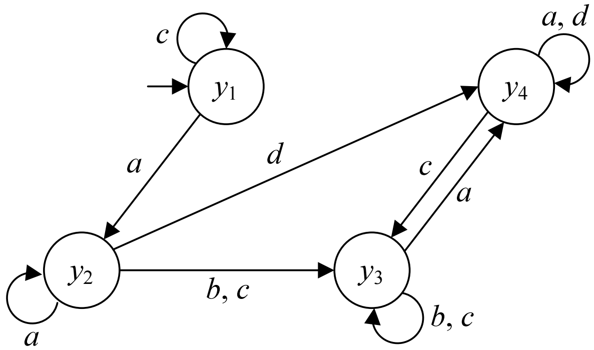

Fig. 6. Reference model Σ'.

We now investigate the existence condition for a corrective controller. We know from Fig. 5 that the desired state y_4 of Σ_2 makes a stable combination of Σ with the state x_4 of Σ_1. Moreover, a slight examination of Fig. 5 shows that from (x_1, y_3) and (x_2, y_3), respectively, we can find as follows an eligible control input sequence to Σ_1 with the induced input sequence to Σ_2.

$$(x_1, y_3, \{a\}, y_4) \;::\; t_1 = ad,\; w_1 = x_3 x_4$$
$$(x_2, y_3, \{a\}, y_4) \;::\; t_2 = ad,\; w_2 = x_3 x_4$$

where the notations follow those of Condition 2.

In this example, we present the controller module C_1 that corrects model mismatch $(x_1, y_3, \{a\}, y_4)$. According to the design procedure addressed in Section III-B, the state flow diagram and output function of C_1 are derived as shown in Fig. 7. At the initial state ξ_0, C_1 moves to the transition state ξ_t whenever receiving a stable combination with (x_1, y_3)—in this case (x_1, y_3, c). When the external input v changes to a at ξ_t, C_1 initiates the corrective control action by transferring to the first auxiliary state ξ_1 and generating the first control input a of t_1. Upon ensuring that Σ reaches (x_3, y_2), i.e., upon receiving the input pair (x_3, y_2), C_1 moves to the second auxiliary state ξ_2, and generates the second control input d. Finally, model matching is accomplished at ξ_2 when Σ reaches the desired state (x_4, y_4) in response to d. Note that the external input v remains a during this correction procedure, which will be conducted instantaneously.

V. CONCLUSION

We have presented a methodology for the design of corrective controllers that achieve model matching for composite asynchronous sequential machines comprising two

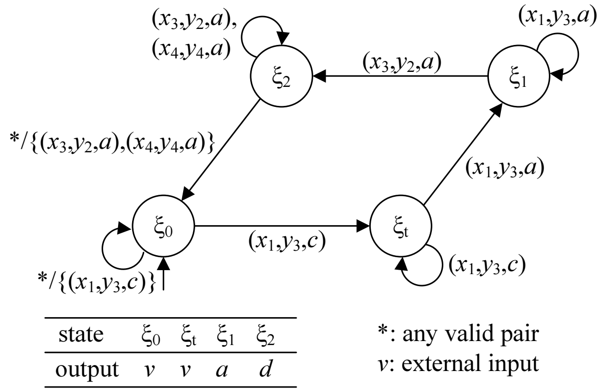

Fig. 7. Controller module C_1.

input/state machines in cascade connection. These controllers utilize the state feedback of each single asynchronous machine to conduct the correction procedure whenever necessary. The existence condition for an appropriate controller has been derived in the framework of composite asynchronous machines, and the design procedure for a controller module has been also outlined. Future studies will tackle the problem of model matching with the constraint that part of state feedback information is unavailable to the controller.

REFERENCES

[1] W. A. Wolovich, P. Antsaklis, and H. Elliott, "On the stability of solutions to minimal and nonminimal design problems," *IEEE Trans. Autom. Control*, vol. 22, no. 1, pp. 88–94, 1977.

[2] M. D. Di Benedetto and J. Grizzle, "Asymptotic model matching for nonlinear systems," *IEEE Trans. Autom. Control*, vol. 39, no. 8, pp. 1539–1550, 1994.

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

[4] N. Yevtushenko, T. Villa, R. K. Brayton, A. Petrenko, and A. L. Sangiovanni-Vincentelli, "Compositionally progressive solutions of synchronous FSM equations," *Discrete Event Syst.: Theory Appl.*, vol. 18, no. 4, pp. 51–89, 2008.

[5] J. Sparsø, and S. Furber, *Principles of Asynchronous Circuit Design – A Systems Perspective*, Kluwer Academic Publishers, 2011.

[6] T. E. Murphy, X. Geng, and J. Hammer, "On the control of asynchronous machines with races," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 1073–1081, 2003.

[7] X. Geng and J. Hammer, "Input/output control of asynchronous sequential machines," *IEEE Trans. Autom. Control*, vol. 50, no. 12, pp. 1956–1970, 2005.

[8] N. Venkatraman and J. Hammer, "On the control of asynchronous sequential machines with infinite cycles," *Int. J. Control*, vol. 79, no. 7, pp. 764–785, 2006.

[9] J.–M. Yang, "Model matching inclusion for input/state asynchronous sequential machines," *Automatica*, vol. 47, no. 3, pp. 597–602, 2011.

[10] J. Hammer, "On corrective control of sequential machines," *Int. J. Control*, vol. 65, no. 2, pp. 249–276, 1996.

[11] J.–M. Yang and S. W. Kwak, "Realizing fault-tolerant asynchronous sequential machines using corrective control," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 6, pp. 1457–1463, 2010.

[12] Z. Kohavi, *Switching and Finite Automata Theory*, 2nd ed. New York: McGraw-Hill, 1978.

[13] J.–M. Yang and J. Hammer, "Asynchronous sequential machines with adversarial intervention: the use of bursts," *Int. J. Control,* vol. 83, no. 5, pp. 956–969, 2010.

[14] E. A. Lee and P. Varaiya, *Structure and Interpretation of Signals and Systems*, 2nd ed., LeeVaraiya.org, 2011.