International Journal of Control
Vol. 83, No. 5, May 2010, 956–969

Taylor & Francis
Taylor & Francis Group

Asynchronous sequential machines with adversarial intervention: the use of bursts

Jung-Min Yang[a] and Jacob Hammer[b]*

[a]*Department of Electrical Engineering, Catholic University of Daegu, 330 Kumrak, Hayang, Gyeongsan, Gyeongbuk, 712-702, Republic of Korea;* [b]*Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA*

(*Received 30 June 2009; final version received 21 November 2009*)

Feedback controllers can automatically counteract the effects of adversarial interventions on the operation of asynchronous sequential machines. The use of bursts – fast outbursts of characters generated by a controlled machine during transition – helps broaden the conditions under which such controllers exist. Necessary and sufficient conditions for the existence of state feedback controllers that employ bursts to counteract the effects of adversarial interventions are presented. Design techniques for such controllers are also described.

Keywords: asynchronous sequential machines; feedback control; adversarial intervention; disturbance rejection

1. Introduction

Asynchronous sequential machines are sequential logic circuits that operate without a clock. They are employed in the construction of computing machines, industrial controllers, traffic control systems and related applications. In addition, asynchronous sequential machines play an important role in the analysis and design of massively parallel computing systems and in the modelling of signalling chains in molecular biology (e.g. Hammer 1994). An important issue in the operation of asynchronous sequential machines is the possible impact of adversarial interventions on proper function. Examples of such interventions are provided by attempts of rogue computer operators to infiltrate computing networks, or by the impact of viruses, bacteria or prions on biological cell function. This article explores the possibility of designing feedback controllers that automatically counteract adversarial interventions and restore affected machines to normal operation.

Specifically, consider an asynchronous machine Σ with two inputs: a legitimate input – the *control input* – and a subversive input – the *adversarial input*. The objective is to develop controllers that automatically counteract the effects of commands received through the adversarial input. The situation is described in Figure 1.

Here, the asynchronous machine Σ is controlled by another asynchronous machine C, which serves as a controller. As depicted in the figure, the machine Σ has two inputs: the control input u and the adversarial input w. The adversarial input represents attempts to interfere with the operation of the machine. The closed loop machine shown in the diagram is denoted by Σ_c. Note that the controller has no direct access to the adversarial input.

A command received at the adversarial input w may cause the machine Σ to undergo state transitions, and these are detected by the controller C. If possible, the controller automatically reacts by entering a command string into the control input u of Σ to reverse these transitions and take Σ back to the state it occupied before the adversarial event. Being an asynchronous machine, the controller's reaction is very quick (ideally, in zero time). Thus, when the controller is successful, users of the closed loop machine Σ_c remain unaware of adversarial interferences.

A brief review of some general features of asynchronous machines is in order. An asynchronous machine may occupy a *stable state* – a state at which the machine lingers until an input change occurs, or a *transient state* – a state through which the machine passes quickly on its way from one stable state to another. Often, a transition from one stable state to another takes the machine through several transient states. A *burst* is a fast string of output characters created by an asynchronous machine during a transition from one stable state to another. When the output of the machine is its state, a burst consists of the sequence of transient states the machine passes during a transition.

In Figure 1, the machine Σ provides its state as output, and C is a state feedback controller equipped

*Corresponding author. Email: hammer@mst.ufl.edu

ISSN 0020–7179 print/ISSN 1366–5820 online

DOI: 10.1080/00207170903502831
http://www.informaworld.com

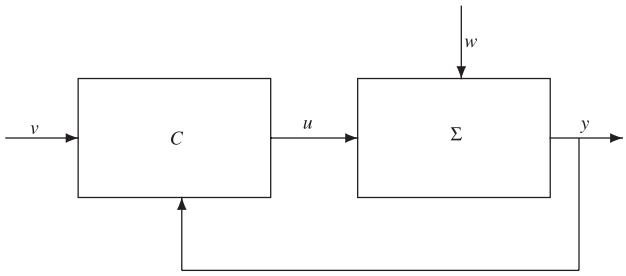

Figure 1. The basic configuration.

with a shift register that records bursts generated by Σ. Endowing C with tools to process bursts enhances its ability to counteract adversarial interventions at the cost of a moderate increase in complexity. Controllers that counteract adversarial interventions without utilising bursts are discussed in Yang and Hammer (2008a, b). There, the controller is activated when the controlled machine Σ reaches a stable state, and the action of the controller is based entirely on the current stable state of Σ; information about the string of transient states that Σ might have passed on its way to the latest stable state, i.e. information about the burst of Σ, is not recorded and not utilised. In contrast, the article introduces a new class of controllers that record bursts generated by the controlled machine Σ and utilise them in the control process. This results in a more powerful class of controllers; indeed, we demonstrate in Example 3.4 that the use of bursts does broaden the conditions under which adversarial interventions can be counteracted.

Other studies on the use of feedback controllers to overcome adversities in the operation of asynchronous sequential machines include Murphy, Geng, and Hammer (2002, 2003), where state feedback controllers are developed to overcome the impact of critical races on asynchronous sequential machines; Venkatraman and Hammer (2006a, b, c), where state feedback controllers are used to overcome the effects of infinite cycles on asynchronous sequential machines; and Peng and Hammer (2008, 2010), which considers the more intricate problem of designing output feedback controllers that overcome the effects of critical races on asynchronous sequential machines.

When operating an asynchronous sequential machine, one must avoid situations where two or more input variables change value at the same time, as this may cause an unpredictable outcome. In this regard, it is most common to enforce *fundamental mode operation* – a policy whereby only one input variable may change value at any instant of time (e.g. Kohavi 1970). For the composite machine of Figure 1, fundamental mode operation implies the following.

Fact 1.1: The composite machine Σ_c of Figure 1 operates in fundamental mode if and only if all the following requirements are met:

(i) Σ is in a stable state when C undergoes transitions;
(ii) C is in a stable state when Σ undergoes transitions; and
(iii) the variables v and w change value only when Σ and C are both in a stable state, and then only one at a time.

In fundamental mode operation, corrective action of the controller C can start only after the machine Σ has reached a stable state, and, while C performs its corrective action, the external inputs v and w must stay constant. The latter is not a burdensome requirement since, being an asynchronous machine, C acts very quickly. In practice, asynchronous machines are almost always operated in fundamental mode, and so are all the machines considered in this article.

Studies dealing with other aspects of the control of sequential machines can be found in Ramadge and Wonham (1987) and Thistle and Wonham (1994), where the theory of discrete event systems is investigated; in Dibenedetto, Saldanha, and Sangiovanni-Vincentelli (1994), Hammer (1994, 1995, 1996a, b, 1997) and Barrett and Lafortune (1998), where issues related to control and model matching for sequential machines are considered; in the references cited in these publications, and elsewhere. Note that the references listed in this paragraph do not take into consideration specialised issues related to the function of asynchronous machines, such as the implications of stable states, transient states, and fundamental mode operation.

The article is organised as follows. Section 2 reviews notation and general background. The main discussion starts in Section 3, with the introduction of a notion that is critical to the existence of controllers – the notion of detectability. This notion determines the feasibility of fundamental mode operation of the closed loop machine, and hence determines whether adversarial interventions can be counteracted in a reliable deterministic fashion. The remaining sections of the article employ detectability to derive necessary and sufficient conditions for the existence of controllers that overcome the effects of adversarial interventions. An illustrative example is immersed throughout the text to demonstrate notions and techniques.

2. Asynchronous sequential machines

Our discussion is within the general framework of Murphy, Geng, and Hammer (2002, 2003), Geng and

Hammer (2004, 2005), Venkatraman and Hammer (2006a, b, c) and Yang and Hammer (2008a, b). Given an alphabet D, denote by D^* the set of all strings of characters of D and by D^+ the set of all non-empty strings of characters of D. For a string $z := z_1 z_2 \in D^+$ formed by the concatenation of two strings $z_1, z_2 \in D^*$, the string z_1 is called a *prefix* of z. The string z_1 is a *strict prefix* of z if neither z_1 nor z_2 are empty strings.

An asynchronous sequential machine Σ with two inputs is represented by a sextuple $\Sigma = (A \times B, Y, X, x^0, f, h)$, where A is the control input alphabet, B is the adversarial input alphabet, Y is the output alphabet, X is a set of n states, x^0 is the initial state, $f: X \times A \times B \to X$ is a partial function serving as the *recursion function* and $h: X \to Y$ is the *output function* of Σ. The machine operates according to the recursion

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w_k), \\ y_k &= h(x_k, u_k, w_k), \quad k = 0, 1, 2, \ldots, \end{aligned} \tag{1}$$

where $u_0, u_1, u_2, \ldots$ is the control input sequence, $w_0, w_1, w_2, \ldots$ is the adversarial input sequence, $x_0, x_1, x_2, \ldots$ is the resulting sequence of states and $y_0, y_1, y_2, \ldots$ is the sequence of output characters. When $y_k = x_k$ for all integers k, i.e. when the output function h is the identity function, the machine Σ is an *input/state machine*. This article concentrates on the control of input/state machines. An input/state machine is characterised by the triple $\Sigma = (A \times B, X, f)$; the initial state x^0 plays no particular role in our discussion, so we will often ignore it.

A triplet $(x, u, w) \in X \times A \times B$ is a *valid combination* if the recursion function f is defined at it. Similarly, a pair $(x, u) \in X \times A$ or $(x, w) \in X \times B$ is *valid* if there are characters $w \in B$ or, respectively, $u \in A$ such that (x, u, w) is a valid combination. A *stable combination* is a valid combination that satisfies the equality $x = f(x, u, w)$, i.e. a 'fixed point' of f; the state x of a stable combination is called a *stable state*. According to (1), the asynchronous machine Σ rests at a stable combination until a change occurs at its control input or at its adversarial input.

When (x, u, w) is not a stable combination, it initiates a chain of transitions $x_0 = x$, $x_1 = f(x_0, u, w)$, $x_2 = f(x_1, u, w), \ldots$, which may or may not terminate. If this chain of transitions terminates, then there is an integer $i \geq 0$ such that $x_i = f(x_i, u, w)$; in such case, (x_i, u, w) is a stable combination, and x_i is called the *next stable state* of x with the input (u, w). If this chain of transitions does not terminate, then the triplet (x, u, w) is part of an *infinite cycle* (e.g. Kohavi 1970). This article concentrates on machines with no infinite cycles. Thus, in our case, every valid combination (x, u, w) has a next stable state.

The *stable recursion function* s of Σ is a partial function $s: X \times A \times B \to X$ given, for every valid combination (x, u, w), by $s(x, u, w) := x'$, where x' is the next stable state of x with the input (u, w). The stable recursion function gives rise to the *stable-state machine* $\Sigma_{|s} := (A \times B, X, s)$. When an input string $\alpha = \alpha_0 \alpha_1 \cdots \alpha_m \in (A \times B)^+$ is applied to the machine Σ, the resulting stable state is

$$s(x, \alpha) := s(s(s(x, \alpha_0), \alpha_1) \ldots, \alpha_m). \tag{2}$$

Of course, to preserve fundamental mode operation, only one component of α can change at each step.

Consider now the case where the machine Σ is at a stable combination (x, a'), where $x \in X$ and $a' \in A \times B$, when a change occurs in one of the input characters. Let $a \in A \times B$ be the new input pair and let $x' := s(x, a)$ be the next stable state of Σ. Assume that this change makes the machine Σ undergo a string of $i \geq 1$ transitions $x_1 := f(x, a)$, $x_2 := f(x_1, a), \ldots, x_i := f(x_{i-1}, a) = x'$, so that $x_i = x'$. Then, the string

$$b(x, a) := x_1 x_2 \cdots x_i$$

is the *burst* created by the pair (x, a). We define $b(x, a) := \varnothing$ (the empty set) when (x, a) is not a valid pair. As the machine Σ is asynchronous, a burst occurs very quickly (ideally, in zero time).

Consider again the machine $\Sigma = (A \times B, X, f)$. Often, not all characters of the adversarial input alphabet B are actively being used by the adversarial input agent. Let $\Omega \subset B$ the set of all adversarial input characters that may actually appear at the adversarial input of Σ. When Ω is a proper subset of B, some characters of B are never used by the adversarial agent. The set Ω is called the *adversarial uncertainty*.

The controller C of Figure 1 has two inputs: one is the burst produced by the machine Σ and the other is the external input character v of the closed loop machine. Now, letting X be state set of Σ, the burst of Σ is a member of the set X^* of strings of states. The external input alphabet of the closed loop machine is usually taken as the input alphabet A of the controlled machine Σ. Then, the input alphabet of C is $X^* \times A$. The controller C drives the machine Σ, and hence its output alphabet is equal to the input alphabet A of Σ. As a result, we can write $C = (X^* \times A, A, \Xi, \xi^0, \phi, \eta)$, where Ξ is the state set, ξ^0 is the initial state, ϕ is the recursion function, and η is the output function of C. Below, we denote by $\Sigma_{c|s}$ the stable-state machine induced by the closed-loop machine Σ_c of Figure 1. The main objective of our discussion can now be stated as follows.

Problem 2.1: *Control objective*: Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with adversarial uncertainty Ω, and consider the closed loop

configuration of Figure 1. Find necessary and sufficient conditions for the existence of a controller C for which the stable state closed-loop machine $\Sigma_{c|s}$ remains unaffected by adversarial intervention and operates in fundamental mode. If such a controller exists, describe its design.

For a user, the observed behaviour of an asynchronous machine is its stable state behaviour, since transients of asynchronous machines disappear very quickly (ideally, in zero time). Thus, when the stable state response $\Sigma_{c|s}$ remains unaffected by adversarial activity, the controller C achieves automatic protection against adversarial intervention.

3. Detectability

True to the nature of an adversarial input or a disturbance, no direct information is available about the particular adversarial input character that is active at any given time. The only a priori information available is that the adversarial input character is a member of the adversarial uncertainty Ω. When Ω includes more than one character, the exact value of the adversarial input character may be uncertain. This uncertainty raises important issues about the operation of the composite machine Σ_c of Figure 1. Especially critical is the impact of such uncertainty on fundamental mode operation.

Recall from Fact 1.1 that, in order to achieve fundamental mode operation, the controller C cannot react to an adversarial intervention until the machine Σ has reached its next stable state. Thus, it must be possible for C to determine whether or not Σ has reached its next stable state, despite uncertainty about the adversarial input. As the controller has access only to the burst of Σ and to the control input value u, this leads us to the following notion, which generalises a concept of Yang and Hammer (2008a).

Definition 3.1: Let $\Sigma = (A \times B, X, f)$ be an input/state asynchronous machine with adversarial uncertainty Ω. Assume that Σ is in a stable combination at the state x when the control input character switches to u. Let b be the burst induced by the resulting stable transition. Then, (x, u) is a *detectable pair* if it is possible to determine from b and u whether or not Σ has reached its next stable state.

Needless to say, fundamental mode operation of the composite machine of Figure 1 is possible only at detectable pairs, since at non-detectable pairs one cannot determine from available data whether a stable state has been reached. The determination of whether or not a given pair is detectable depends on the information that is available about the adversarial input, as we discuss next.

It is often possible to deduce more information about the adversarial input than the a priori information provided by the adversarial uncertainty Ω. One source of such information is the current stable combination of the machine Σ. Indeed, let s be the stable recursion function of Σ, and assume that Σ is in a stable combination at an initial state x^0 with the control input value u. Then, the set of all adversarial input characters compatible with this information is given by

$$\omega(x^0, u) := \{w \in \Omega : s(x^0, u, w) = x^0\}. \tag{3}$$

More generally, assume that Σ is in a stable combination at the state x with the control input character u. When x is not an initial state, further information about the adversarial input character can be derived from historical data about the response of the machine Σ along its way to the state x. Denote by $\upsilon(x, u)$ the set of all adversarial input characters that are compatible with the information currently available. We refer to $\upsilon(x, u)$ as the *residual adversarial uncertainty* and we proceed now to calculate it. Note that at an initial state $x = x^0$, we have $\upsilon(x^0, u) = \omega(x^0, u)$.

Assume that the control input character of Σ switches from u to the character u', and let x' be the next stable state of Σ. In fundamental mode operation, the adversarial input character w remains constant during the resulting transition, which may consist of $q \geq 1$ transient steps: $x_1 := f(x, u', w)$, $x_2 := f(x_1, u', w), \ldots, x_q := f(x_{q-1}, u', w) = x'$. Denote by

$$b(x, u', w) := x_1 \cdots x_q \tag{4}$$

the burst created by this transition. As the set of all possible adversarial input characters is given by the residual adversarial uncertainty $\upsilon(x, u)$, the set of all bursts that could result from switching the control input character from u to u' is given by

$$B(x, u, u') := \{b(x, u', w) : w \in \upsilon(x, u)\}. \tag{5}$$

Example 3.2: Consider an input/state asynchronous machine Σ with the control input alphabet $A = \{a, b, c\}$, the adversarial input alphabet $B = \{\alpha, \beta, \gamma\}$, the state set $X = \{x^1, x^2, x^3\}$, the adversarial uncertainty $\Omega = B$, and the state flow diagram of Figure 2.

Assume that Σ is in a stable combination at the initial state x^1 with the control input character b. Then, direct inspection shows that $\omega(x^1, b) = \{\beta, \gamma\}$. Assume further that the control input character changes to a, potentially starting state transitions. An examination of Figure 2 yields $b(x^1, a, \beta) = x^2x^3$ and $b(x^1, a, \gamma) = x^3x^2$. Thus,

$$B(x^1, b, a) = \{x^2x^3, x^3x^2\}. \tag{6}$$

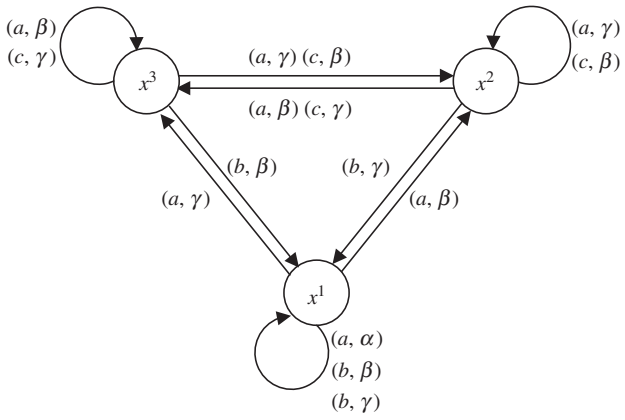

Figure 2. State flow diagram of Σ.

In these terms, we obtain the following characterisation of detectability (compare to Peng and Hammer (2008, 2010), where output feedback controllers for asynchronous machines with critical races are discussed).

Theorem 3.3: *Let* $\Sigma = (A \times B, X, f)$ *be an asynchronous input/state machine resting in a stable combination at the state* x, *when the control input character switches from* u *to* u'. *Let* $\upsilon(x, u)$ *be the residual adversarial uncertainty and let* $B(x, u, u')$ *be the set of bursts* (5). *Then, the following are equivalent*:

(i) *The pair* (x, u') *is detectable*.
(ii) *No member of* $B(x, u, u')$ *is a strict prefix of another member*.

Proof: We use the notation of (4) and (5). First, by contradiction, assume that (ii) is not valid. Then, there are adversarial input characters $w, w' \in \upsilon(x, u)$ for which $b(x, u', w)$ is a strict prefix of $b(x, u', w')$. Let x' be the stable state that Σ reaches at the end of the burst $b(x, u', w)$, i.e. $b(x, u', w) = \cdots x'$, so that (x', u', w) is a stable combination. Now, since $b(x, u', w)$ is a strict prefix of $b(x, u', w')$, the state x' is reached before the end of the burst $b(x, u', w')$. Denoting by x'' the stable state reached at the end of the burst $b(x, u', w')$, we have $b(x, u', w') = \cdots x' \cdots x''$, so that (x', u', w') is not a stable combination in this case. Thus, when encountering the end of the burst $b(x, u', w)$, there are two possibilities: either (a) the adversarial input is w and Σ is in a stable combination; or (b) the adversarial input is w', and Σ has not reached a stable combination yet. Consequently, as the control input character u is the same in both cases, it is impossible to tell whether or not Σ is in a stable state without knowing the adversarial input character. Hence, the pair (x, u') is not detectable, and (i) implies (ii).

Conversely, assume that (ii) is valid and select an adversarial input character $w \in \upsilon(x, u)$. Then, the burst $b(x, u', w)$ is not a strict prefix of any other burst in the set of all possible bursts $B(x, u, u')$. Consequently, at the end of the burst $b(x, u', w)$, the machine Σ has reached the end of a transition string, and hence must be in a stable combination. As this is valid for all adversarial input characters $w \in \upsilon(x, u)$, it follows that the end state of a burst in $B(x, u, u')$ always indicates a stable combination. Thus, (ii) implies (i), and our proof concludes. □

Example 3.4: Consider the machine Σ of Example 3.2. Assume that Σ is in a stable combination with (x^1, b) when the control input switches to a. We can see from (6) that no member of $B(x^1, b, a)$ is a strict prefix of another member. Hence, by Theorem 3.3, (x^1, b) is detectable. However, this pair is not detectable in the sense of Yang and Hammer (2008a), since the same state can appear as a stable state and as a transient state in the transitions.

So far, we have examined the possibility of detecting the next stable state of the machine Σ after a switch of the control input character. Similar issues arise after a switch of the adversarial input character, as we discuss next. Consider a situation where the machine Σ is in a stable combination (x, u, w) when the adversarial input character switches to w'; here, w' can be any member of the adversarial uncertainty Ω of the machine Σ. This change has the potential of initiating an unauthorised transition of Σ, a transition which the controller C of Figure 1 must counteract. To preserve fundamental mode operation, C must wait until Σ has reached its next stable state before taking any counteraction. As a result, we are faced again with a situation where the controller must determine from the burst and the control input value whether Σ has reached its next stable state. This brings us to the following notion.

Definition 3.5: Let $\Sigma = (A \times B, X, f)$ be an input/state asynchronous machine with adversarial uncertainty Ω. Assume that Σ is in a stable combination with the state x and the control input u, when a switch of the adversarial input character occurs. Then, the pair (x, u) is *adversarially detectable* if it is possible to determine from the control input and the burst of Σ whether or not Σ has reached its next stable state. The machine Σ is *adversarially detectable* if every valid pair (x, u) of Σ is adversarially detectable.

Necessary and sufficient conditions for adversarial detectability are closely analogous to the conditions for detectability listed in Theorem 3.3. First, for a state x and a control input character u, consider the set of bursts

$$B_a(x, u) := \{b(x, u, w) : w \in \Omega\}, \tag{7}$$

i.e. the set of all bursts that may result from a switch of the adversarial input character.

Example 3.6: Continuing with Example 3.2, suppose that Σ is in a stable combination at the state x^1 and the control input character a. Referring to Figure 2, the adversarial input character can either stay at α or switch to one of the characters β or γ; the set of all possible resulting bursts is

$$B_a(x^1, a) = \{b(x, a, \alpha), b(x, a, \beta), b(x, a, \gamma)\} = \{x^1, x^2x^3, x^3x^2\}. \quad (8)$$

The proof of the following statement is similar to the proof of Theorem 3.3.

Theorem 3.7: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with adversarial uncertainty Ω. Assume that Σ is in a stable combination at the state x with the control input character u, when a change at the adversarial input occurs. Let $B_a(x, u)$ be the set of bursts* (7). *Then, the following two statements are equivalent*:

(i) *The pair (x, u) is adversarially detectable.*
(ii) *No member of $B_a(x, u)$ is a strict prefix of another member.*

Example 3.8: Consider again the machine Σ of Example 3.2. From Example 3.6, we can see that no member of $B_a(x^1, a)$ is a strict prefix of another member. Hence, the pair (x^1, a) is adversarially detectable. A similar examination of the remaining stable combinations of Σ shows that they are all adversarially detectable. Therefore, Σ is an adversarially detectable machine.

When the machine Σ is adversarially detectable, the controller C of Figure 1 can always determine whether or not Σ has reached its next stable state after an adversarial intervention has occurred. This is accomplished as follows. The controller keeps track of the latest stable state x of the machine Σ and of its current control input value u. When C detects a burst b of Σ without a corresponding change of the control input, it compares the progressing burst b to members of the set $B_a(x, u)$ of possible bursts. Note that the set $B_a(x, u)$ is known a priori, as it is determined by the known recursion function of the machine Σ. By condition (ii) of Theorem 3.7, the next stable state of Σ is reached when the progressing burst becomes equal to a member of $B_a(x, u)$. At that point, the controller can start to counteract the adversarial transition without impeding fundamental mode operation.

Note that only single-step stable adversarial transitions need to be considered, since the controller C reacts immediately after each single-step transition. All single-step adversarial transitions are characterised by the following matrix.

Definition 3.9: Let $\Sigma = (A \times B, X, f)$ be an adversarially detectable asynchronous machine with the state set $X = \{x^1, x^2, \ldots, x^n\}$, the adversarial uncertainty Ω, and the stable recursion function s. Let $U(x^i)$ be the set of all control input characters that form stable combinations with the state x^i. Then, the *adversarial skeleton matrix* $K^a(\Sigma, \Omega)$ is an $n \times n$ matrix of zeros and ones with the entries

$$K^a_{ij}(\Sigma, \Omega) = \begin{cases} 1 & \text{if } x^j = s(x^i, u, w) \text{ for some} \\ & \quad u \in U(x^i) \text{ and } w \in \Omega, \\ 0 & \text{else,} \end{cases}$$

$i, j = 1, 2, \ldots, n$.

Example 3.10: Consider the machine Σ of Example 3.2. A brief examination of Figure 2 shows that Σ has the stable pairs (x^1, a), (x^1, b), (x^2, a), (x^2, c), (x^3, a) and (x^3, c). Considering the effects of the adversarial inputs in the figure, we obtain the following adversarial skeleton matrix:

$$K^a(\Sigma, \Omega) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}.$$

By examining the outcome of an adversarial transition, it is often possible to reduce the uncertainty about the adversarial input, as we discuss next.

4. Uncertainty

4.1 *Uncertainty after an adversarial transition*

Consider an asynchronous input/state machine $\Sigma = (A \times B, X, f)$ with adversarial uncertainty Ω. Assume that Σ is in a stable combination at an adversarially detectable pair $(x, u) \in X \times A$, when a change at the adversarial input causes Σ to move to a stable combination at the state x'. Let w' be the new adversarial input character. A priori, w' can, of course, be any member of Ω. However, after x' has been reached, we can deduce more information about the value of w'. Indeed, let b be the burst generated by Σ during this stable transition, and denote by $b(x, u, w)$ the burst of Σ induced by a triple $(x, u, w) \in X \times A \times B$. Then, the subset

$$\upsilon(x, u, b) := \{w \in \Omega : b(x, u, w) = b\} \quad (9)$$

includes all adversarial input characters that are compatible with the available data, namely with the starting state x, the control input character u and the burst b. Thus, $\upsilon(x, u, b)$ characterises the uncertainty about the adversarial input character that is compatible with the latest step data.

4.2 *Residual adversarial uncertainty*

The reaction of the controller C of Figure 1 to an adversarial transition consists of applying a string of control input characters to the machine Σ in an attempt to undo the adversarial transition. This string of control input characters guides Σ through a sequence of stable and detectable state transitions, eventually ending at the state Σ occupied before the adversarial transition occurred. The controller applies the control input string one character at a time; after each character, the controller waits until Σ has reached its next stable state, and then applies to Σ the next control input character of the string. The resulting chain of transitions of Σ forms a single stable transition of the closed loop machine Σ_c; it is completed very quickly (ideally, in zero time). The adversarial input character remains constant during this process (fundamental mode operation). In the course of this chain of transitions, further information may be gleaned about the adversarial input character. The uncertainty about the adversarial input character may be reduced at each step of the chain when the outcome of the step becomes known, as we discuss next.

Consider again the asynchronous machine $\Sigma = (A \times B, X, f)$ with the state set $X = \{x^1, x^2, \ldots, x^n\}$, the adversarial uncertainty Ω, and the stable recursion function s. Assume that Σ is at a stable combination $(x^i, u_0, w) \in X \times A \times B$, when a control input string $u = u_0 u_1 u_2 \cdots u_t \in A^+$ is applied, while keeping the adversarial input fixed at the character $w \in \Omega$. Denote by $\alpha := w|u$ the combined input string. Suppose that α takes Σ from a stable combination with the state x^i to a stable combination with the state x^j, through the stable states $x_0 := x^i$, $x_1 = s(x_0, u_1, w)$, $x_2 = s(x_1, u_2, w), \ldots, x_t = s(x_{t-1}, u_t, w) = x^j$. For an integer $p \in \{1, 2, \ldots, t\}$, let $b_p(\alpha)$ be the burst generated by Σ on its way from x_{p-1} to x_p along our chain of transitions. Then, using the notation of (9), the set of all adversarial input characters that are compatible with the data about the single step p is given by

$$\upsilon(x_{p-1}, u_p, b_p(\alpha)) = \{w \in \Omega : b(x_{p-1}, u_p, w) = b_p(\alpha)\}. \tag{10}$$

Now, let $\upsilon_p(\alpha)$ be the residual uncertainty at the end of step p of our input string, i.e. the set of adversarial input characters that are compatible with the entire information available about Σ at the end of step p. This information would include data about steps $1, 2, \ldots, p$ as well as data about the adversarial input character that was available prior to step 1 of the current transition chain.

Recall that our objective here is to counteract adversarial transitions. Thus, a controlled transition from x^i to x^j would be in response to an adversarial transition from x^j to x^i. Denoting by $b_a(j, i)$ the burst registered by the controller during the adversarial transition from x^j to x^i, it follows from (9) that, immediately after the adversarial transition, we can infer that the adversarial input character belongs to the set

$$\upsilon_{ij}(u_0) := \upsilon(x^j, u_0, b_a(j, i)). \tag{11}$$

Using (10), we conclude that the residual adversarial uncertainty at the end of step p of our input string is given by the recursion

$$\begin{gathered}\upsilon_0(\alpha) := \upsilon_{ij}(u_0), \\ \upsilon_p(\alpha) = \upsilon_{p-1}(\alpha) \cap \upsilon(x_{p-1}, u_p, b_p(\alpha)), \quad p = 1, 2, \ldots, t.\end{gathered} \tag{12}$$

This proves the following.

Lemma 4.1: *The set of adversarial input characters* $\upsilon_p(\alpha)$ *of* (12) *forms the residual adversarial uncertainty at the end of step p of a chain of stable transitions induced by the input string* $\alpha = w|u_0 u_1 u_2 \cdots u_t \in \Omega|A^+$.

The residual adversarial uncertainty characterises the information available to the controller about the adversarial input character before it commences the next step of the control input string.

4.3 *The extended matrix of stable transitions*

Recall the mode of operation of the composite machine of Figure 1: the controller C activates immediately after detecting a burst of the machine Σ that has occurred without a corresponding change at the control input, i.e. immediately after an adversarial transition. The controller then creates a string of control input characters that takes Σ through a chain of stable transitions back to the state it occupied before the adversarial transition. During this chain of transitions (which occurs, ideally, in zero time), the adversarial input remains constant. Thus, we concentrate on the effect of the control input while the adversarial input is constant.

Let s be the stable recursion function of Σ, and let Ω be its adversarial uncertainty. A state x' of Σ is *stably reachable* from a state x in the presence of an adversarial input character w if there is a control input string $u \in A^+$ such that $x' = s(x, u, w)$. We construct now a matrix that characterises the stable reachability features of the machine Σ, starting with some notation (compare to Yang and Hammer (2008a)).

For a string $w|u \in \Omega|A^+$, where $u = u_0 u_1 \cdots u_k$, define the projection $\Pi_i^c : B|A^+ \to A$ onto the i-th

control input character by setting

$$\Pi_i^c w|u := \begin{cases} u_i & \text{if } i \le k, \\ u_k & \text{if } i > k, \end{cases}$$

$i = 0, 1, 2, \ldots$. Now, let $X = \{x^1, x^2, \ldots, x^n\}$ be the state set of Σ, and consider two states $x^i, x^j \in X$. The set of all control input strings $u \in A^+$ that take Σ from a stable combination with x^i to a stable combination with x^j in the presence of the adversarial input character $w \in \Omega$ is

$$\sigma(w, x^i, x^j) := \{w|u \in \Omega|A^+ : x^j = s(x^i, u, w) \quad \text{and} \quad (x^i, \Pi_0^c w|u, w) \text{ is a stable combination}\}.$$

Aggregating over all possible adversarial input characters, we obtain the set of input strings

$$\rho_{ij} := \bigcup_{w \in \Omega} \sigma(w, x^i, x^j) \subset \Omega|A^+.$$

Example 4.2: An examination of Figure 2 shows that, for the machine Σ of Example 3.2, we have $\rho_{21} = \{\gamma|ab, \beta|cab\}$.

Next, denote by $\Pi^a : B|A^+ \to B : (w|u) \mapsto w$ the projection onto the adversarial input character. Assume now that an adversarial transition from the state x^j to the state x^i has occurred, ending at a stable combination at x^i with the control input character u_0. Then, by (11), the adversarial input character must belong to the set $\upsilon_{ij}(u_0)$. The set of all control input strings that take Σ back from x^i to x^j in the presence of an adversarial input character from $\upsilon_{ij}(u_0)$ is given by

$$R_{ij}^*(\Sigma, \Omega) := \{\alpha \in \rho_{ij} : \Pi^a \alpha \in \upsilon_{ij}(\Pi_0^c \alpha)\}, \quad i, j = 1, 2, \ldots, n. \tag{13}$$

We call $R^*(\Sigma, \Omega)$ the *extended matrix of stable transitions* of the machine Σ. Each entry of $R^*(\Sigma, \Omega)$ is a set of strings $w|u \in \Omega|A^+$, where u takes Σ from a stable combination with x^i to a stable combination with x^j in the presence of the adversarial input character w. In each entry, the adversarial input characters are restricted to those that are consistent with the starting state and the initial control input character. The following statement is a direct consequence of the construction.

Lemma 4.3: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous machine with adversarial uncertainty Ω, and let $R^*(\Sigma, \Omega)$ be the extended matrix of stable transitions of Σ. Then, the following two statements are equivalent for all $i, j = 1, 2, \ldots, n$:*

(i) *The entry $R_{ij}^*(\Sigma, \Omega)$ includes a string $w|u$.*
(ii) *The state x^j is stably reachable from the state x^i in the presence of the adversarial input character w.*

5. Detectable feedback paths

Our next objective is to determine whether the extended matrix of stable transitions includes control input strings that can be implemented by a feedback controller. If so, this will allow us to construct a feedback controller that can automatically reverse adversarial transitions. To this end, consider an asynchronous input/state machine $\Sigma = (A \times B, X, f)$ with adversarial uncertainty Ω and extended matrix of stable transitions $R^*(\Sigma, \Omega)$. Let $X = \{x^1, x^2, \ldots, x^n\}$ be the state set of Σ, and examine a string $\alpha = w|u \in R_{ij}^*(\Sigma, \Omega)$. Write $u = u_0 u_1 \cdots u_q$ and let $x_p := s(x^i, u_0 u_1 \cdots u_p, w)$ be the stable state of the machine Σ at the end of step p of the string, where $p \in \{0, 1, \ldots, q\}$, $x_0 := x^i$, and $x_q := x^j$. Then, the residual adversarial uncertainty $\upsilon_p(\alpha)$ of (12) characterises the information available about the adversarial input character at the end of step p. Consequently, in order for the controller to operate in fundamental mode, the pair (x_p, u_{p+1}) must be detectable with respect to the residual adversarial uncertainty $\upsilon_p(\alpha)$ at all steps $p = 0, 1, \ldots, q-1$.

Now, let $b_p(\alpha)$ be the burst generated by Σ during its transition from x_{p-1} to x_p. Then, the sequence of bursts generated by the machine Σ along the string of transitions up to step p is

$$b_1^p(\alpha) := \begin{cases} \varnothing & \text{for } p = 0, \\ \{b_1(\alpha), b_2(\alpha), \ldots, b_p(\alpha)\} & \text{otherwise.} \end{cases} \tag{14}$$

At the step p, the controller cannot distinguish between different adversarial input characters that produce the same string of bursts $b_1^p(\alpha)$. As a result, for all such adversarial input characters, the controller must produce the same next control input character u_{p+1}. This is, of course, a fundamental feature of feedback controllers and is a consequence of causality.

Before continuing, we need some notation. Denote by $\Pi^c : B|A^+ \to A^+ : (w|u) \mapsto u$ the projection onto the control input string, and, for a string $\alpha = w|u_0 u_1 \cdots u_q \in B|A^+$ and an integer $p \ge 0$, denote by

$$\alpha_{|p} := \begin{cases} w|u_0 u_1 \cdots u_p & \text{for } p \le q, \\ w|u_0 u_1 \cdots u_q & \text{for } p > q \end{cases}$$

the truncation to the first p characters of the control input string.

Now, given a set of strings $S \subset B|A^+$, an integer $p \ge 0$ and a string $\alpha \in S$, denote by $S(\alpha, p)$ the set of all strings of S which, up to step p, have the same control input characters and produce the same string of bursts as α; namely

$$S(\alpha, p) := \begin{cases} \{a \in S : \Pi_0^c a = \Pi_0^c \alpha\} & \text{for } p = 0, \\ \{a \in S : \Pi^c a_{|p} = \Pi^c \alpha_{|p} \\ \quad \text{and } b_1^p(a) = b_1^p(\alpha)\} & \text{for } p > 0. \end{cases} \tag{15}$$

Note that strings in $S(\alpha, p)$ may have different adversarial input characters. The following notion is critical to our discussion, as it singles out a structural feature that underlies the existence of controllers that automatically counteract adversarial transitions (compare to Venkatraman and Hammer (2006c) and Yang and Hammer (2008a)).

Definition 5.1: Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with the state set $X = \{x^1, x^2, \ldots, x^n\}$, the stable recursion function s, and the initial adversarial uncertainty Ω. Assume that Σ is in a stable combination at the state x^i with the control input value u_0. A subset $S \subset R^*_{ij}(\Sigma, \Omega)$ is a *detectable feedback path* from x^i to x^j if the following conditions are satisfied for every element $\alpha \in S$ and every integer $p \geq 0$:

(i) $\Pi^c_0 S$ consists of the single character u_0;
(ii) the set $\Pi^c_{p+1} S(\alpha, p)$ consists of a single character; and
(iii) the pair $(s(x^i, \alpha_{|p}), \Pi^c_{p+1}\alpha)$ is detectable with respect to the residual uncertainty $\upsilon_p(\alpha)$ of (12).

The character u_0 is called the *initial control input character* of S.

As we show later, the presence of a detectable feedback path is equivalent to the existence of a controller that counteracts an adversarial transition. In the meanwhile, we provide an example of the construction of detectable feedback paths.

Example 5.2: Referring to the machine Σ of Example 3.2, we have seen in Example 4.2 that $\rho_{21} = \{\gamma|ab, \beta|cab\}$. Assume that an adversarial transition has occurred, taking Σ from the state x^1 to the state x^2 while producing the burst $b_a(1,2) = x^3x^2$ and ending at the stable pair (x^2, a). Then, $u_0 := a$ must be the initial control input character of a control input string that reverses this adversarial transition and returns Σ to the state x^1. Using (11), we obtain $\upsilon_{21}(a) = \upsilon(x^1, a, x^3x^2) = \{\gamma\}$; also, from (13), we get that $\gamma|ab \in R^*_{21}(\Sigma, \Omega)$. Now, set $S := \{\gamma|ab\}$. As S consists of a single string, conditions (i) and (ii) of Definition 5.1 are clearly valid.

Regarding condition (iii) of Definition 5.1, note that we have to examine only the case $p = 0$, since counting of the control input characters starts from 0 and the string $\gamma|ab$ is only two control characters long. Now, $(s(x^2, \gamma|ab_{|0}), b) = (s(x^2, \gamma|a), b) = (x^2, b)$ and $\upsilon_0(\gamma|ab) = \{\gamma\}$; referring to (5), the set of bursts $B(x^2, a, b)$ includes in this case the single member $b(x^2, b, \gamma)$. Thus, (x^2, b) is detectable with respect to $\upsilon_0(\gamma|ab) = \{\gamma\}$, and hence $S = \{\gamma|ab\}$ forms a detectable feedback path from x^2 to x^1.

The next statement indicates the equivalence between the presence of detectable feedback paths and the existence of controllers that automatically counteract adversarial transitions. Note that the presence of detectable feedback paths is a structural feature of the controlled machine Σ and can be determined by an examination of the recursion function of Σ. Consequently, the next statement forms the basis of a computational procedure for ascertaining the existence of controllers that counteract adversarial transitions.

Theorem 5.3: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with adversarial uncertainty Ω, state set $X = \{x^1, x^2, \ldots, x^n\}$ and extended matrix of stable transitions $R^*(\Sigma, \Omega)$. Assume that Σ underwent an adversarial transition from the state x^j to the state x^i in the presence of the control input character u_0. Then, the following two statements are equivalent for all $i, j \in \{1, 2, \ldots, n\}$:*

(i) *There is a controller $C(i,j)$ that takes Σ back from a stable combination with x^i to a stable combination with x^j in fundamental mode operation.*
(ii) *The entry $R^*_{ij}(\Sigma, \Omega)$ includes a detectable feedback path with initial control input character u_0.*

Proof: Assume first that (i) is valid, and let $S \subset B|A^+$ be the set of all input strings of Σ that the controller $C(i,j)$ may generate in the process of steering Σ from x^i back to x^j. Then, as the transition back starts from a stable combination of Σ with the control input character u_0, we have that u_0 is the initial character of every string of S, namely $u_0 = \Pi^c_0 S$; hence, condition (i) of Definition 5.1 is valid. Further, as $C(i,j)$ is a feedback controller, the output character of $C(i,j)$ is determined by past bursts of Σ and by past and present output characters of $C(i,j)$. In other words, an equal burst history of Σ combined with an equal output history of $C(i,j)$ must result in the same output character of $C(i,j)$, validating condition (ii) of Definition 5.1. Finally, fundamental mode operation of the composite machine $\Sigma_{C(i,j)}$ requires that all steps through which Σ is taken by $C(i,j)$ must be detectable, as stated in condition (iii) of Definition 5.1. Thus, all conditions of Definition 5.1 are valid, and S is a detectable feedback path.

Conversely, assume that condition (ii) of the theorem is valid, and let $S \subset R^*_{ij}(\Sigma, \Omega)$ be a detectable feedback path with initial control input character u_0. Define a controller $C(i,j)$ as follows:

(a) The initial output character of $C(i,j)$ is the character u_0.
(b) Using recursion, assume that the action of the controller $C(i,j)$ has been defined up to a step $p \geq 0$, and let $\alpha \in S$ be any string for which steps

$0, 1, \ldots, p$ correspond to this controller action. Then, the next character generated by $C(i,j)$ is the single member of the set $\Pi^c_{p+1}S(\alpha, p)$ (condition (ii) of Definition 5.1).

By condition (iii) of Definition 5.1, all stable transitions of Σ induced by $C(i,j)$ are detectable. Thus, the closed loop machine $\Sigma_{C(i,j)}$ operates in fundamental mode. Finally, the fact that $S \subset R^*_{ij}(\Sigma, \Omega)$ implies that any string generated by $C(i,j)$ takes the machine Σ from a stable combination with the state x^i to a stable combination with the state x^j, and our proof concludes. □

Theorem 5.3 provides a necessary and sufficient condition for the existence of a controller that overcomes adversarial interventions. This condition is stated in terms of structural features of the controlled machine Σ, namely the existence of detectable feedback paths. Our next objective is to show that the controller $C(i,j)$ of the proof of Theorem 5.3 can always be implemented with a finite state set. To this end, denote by $\#Z$ the cardinality of a set Z. Also, define the *length* of a string $\alpha = w|u \in B|A^+$ as the length of the control input string u, i.e. $|\alpha| := |u|$.

Definition 5.4: Let $\Sigma = (A \times B, X, f)$ be an asynchronous machine with the adversarial uncertainty Ω and the extended matrix of stable transitions $R^*(\Sigma, \Omega)$. Denote $\kappa := (\#X)(\#\Omega)$. Then, the *matrix of stable transitions* $R(\Sigma, \Omega)$ is obtained from $R^*(\Sigma, \Omega)$ by deleting all entries of length exceeding κ.

The significance of the matrix of stable transitions originates from the following fact.

Lemma 5.5: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with adversarial uncertainty Ω and state set $X = \{x^1, x^2, \ldots, x^n\}$. Let $R^*(\Sigma, \Omega)$ be the extended matrix of stable transitions of Σ and let $R(\Sigma, \Omega)$ be the matrix of stable transitions. Then, the following two statements are equivalent for all $i, j \in \{1, 2, \ldots, n\}$:*

(i) *The entry $R^*_{ij}(\Sigma, \Omega)$ includes a detectable feedback path.*
(ii) *The entry $R_{ij}(\Sigma, \Omega)$ includes a detectable feedback path.*

Proof: As $R_{ij}(\Sigma, \Omega)$ is a subset of $R^*_{ij}(\Sigma, \Omega)$, it is clear that (ii) implies (i). To prove the converse direction, assume that there is a detectable feedback path $S^* \subset R^*_{ij}(\Sigma, \Omega)$. For an element $\alpha \in S^*$, let $\upsilon_p(\alpha)$ be the residual uncertainty at step $p \geq 0$. Consider the pair $(s(x^i, \alpha_{|p}), \upsilon_p(\alpha))$; here, the first member is a state of Σ and the second member is a subset of Ω. Now, Σ has $n := \#X$ states. Also, by (12), the sequence $\upsilon_0(\alpha), \upsilon_1(\alpha), \upsilon_2(\alpha), \ldots$ is a monotone decreasing sequence of subsets of Ω. Therefore, for an integer $p \geq 0$, the number of different subsets that can be candidates of $\upsilon_p(\alpha)$ cannot exceed $\#\upsilon_0(\alpha)$. As $\#\upsilon_0(\alpha) \leq \#\Omega$, it follows that the number of possible distinct pairs $(s(x^i, \alpha_{|p}), \upsilon_p(\alpha))$ cannot exceed $\kappa := n(\#\Omega)$.

Consider now a string $\alpha = w|u_0u_1u_2 \ldots, u_{|\alpha|} \in R^*_{ij}(\Sigma, \Omega)$ of length $|\alpha| > \kappa$. Then, the sequence $(s(x^i, \alpha_{|k}), \upsilon_k(\alpha))$, $k = 1, 2, \ldots, |\alpha|$, must include a repeat, say $(s(x^i, \alpha_{|p}), \upsilon_p(\alpha)) = (s(x^i, \alpha_{|r}), \upsilon_r(\alpha))$, where $r > p$. Remove from the control input string of α all the terms from $p+1$ to r, i.e. replace α by $\alpha' := w|u_0u_1u_2 \cdots u_pu_{r+1} \cdots u_{|\alpha|}$ (or $\alpha' := w|u_0u_1u_2 \cdots u_p$ if $r = |\alpha|$). Then, α' still takes Σ from a stable combination with x^i to a stable combination with x^j. Furthermore, since this process preserves the state and the residual adversarial uncertainty at each step, all pairs along the shorter string remain detectable.

Now, apply this process repeatedly to each member of S^* until all repetitions are eliminated from all members; denote the resulting set by S. Then, S consists of strings of length not exceeding κ, and each member of S takes Σ from a stable combination with x^i to a stable combination with x^j. Thus, $S \subset R_{ij}(\Sigma, \Omega)$. Additionally, all pairs along each member of S remain detectable and all requirements of Definition 5.1 remain valid. Thus, S forms a detectable feedback path, and, since $S \subset R_{ij}(\Sigma, \Omega)$, our proof concludes. □

In view of Lemma 5.5, the matrix $R^*(\Sigma, \Omega)$ can be replaced in Theorem 5.3 by the matrix $R(\Sigma, \Omega)$. This assures that all computations and implementations are finite, and it leads to the following.

Corollary 5.6: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with adversarial uncertainty Ω, state set $X = \{x^1, x^2, \ldots, x^n\}$, and matrix of stable transitions $R(\Sigma, \Omega)$. Assume that Σ underwent an adversarial transition from the state x^j to the state x^i in the presence of the control input character u_0. Then, the following two statements are equivalent for all $i, j \in \{1, 2, \ldots, n\}$:*

(i) *There is a controller $C(i,j)$ that takes Σ back from a stable combination with x^i to a stable combination with x^j in fundamental mode operation.*
(ii) *The entry $R_{ij}(\Sigma, \Omega)$ includes a detectable feedback path with initial control input character u_0.*

To summarise, we have seen in this section that the presence of detectable feedback paths is the critical condition for the existence of controllers that automatically counteract adversarial interventions. The presence of detectable feedback paths is a structural feature of the matrix of stable transitions and can be validated constructively, as we discuss below.

6. Counteracting adversarial transitions

Recall that an adversarial transition is caused by a change at the adversarial input and occurs while the control input remains fixed (fundamental mode operation). It takes the affected machine from one stable combination to another; as the control input character remains fixed, it must form stable combinations with both the initial state and the terminal state of the adversarial transition. The control input string that reverses an adversarial transition starts with the control input character that was active during the adversarial transition itself.

In formal terms, consider an asynchronous input/state machine $\Sigma = (A \times B, X, f)$ with adversarial uncertainty Ω, stable recursion function s and state set $X = \{x^1, x^2, \ldots, x^n\}$. In an adversarial transition, the adversarial input character switches from, say, w to w', while the control input character remains fixed at, say, u. Thus, the set $U_{ij}(\Sigma)$ of all control input characters that can be active during an adversarial transition from a stable combination with x^j to a stable combination with x^i is given by

$$U_{ij}(\Sigma) := \{u \in A : x^j = s(x^j, u, w) \quad \text{and} \quad x^i = s(x^j, u, w') \text{ for some } w, w' \in \Omega\}. \tag{16}$$

Example 6.1: For the machine Σ of Example 3.2, an examination of Figure 2 yields

$$U_{21}(\Sigma) = U_{31}(\Sigma) = U_{32}(\Sigma) = U_{23}(\Sigma) = \{a\}.$$

We introduce the following $n \times n$ matrix.

Definition 6.2: Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with state set $X = \{x^1, x^2, \ldots, x^n\}$, adversarial uncertainty Ω, and matrix of stable transitions $R(\Sigma, \Omega)$, and let $U_{ij}(\Sigma)$ be the set of control input characters given by (16). Then, the *control skeleton matrix* $K^c(\Sigma, \Omega)$ of Σ is an $n \times n$ matrix of zeros and ones with the entries

$$K^c_{ij}(\Sigma, \Omega) = \begin{cases} 1 & \text{if } R_{ij}(\Sigma, \Omega) \text{ includes a detectable feedback path with initial character } u \text{ for every } u \in U_{ij}(\Sigma), \\ 0 & \text{else,} \end{cases} \tag{17}$$

$i, j = 1, 2, \ldots, n$.

In view of Corollary 5.6, the control skeleton matrix characterises all pairs of states among which adversarial transitions can always be counteracted by an automatic controller. This proves the following statement, which is one of the main results of the article. (Below, inequalities between numerical matrices are interpreted entry-by-entry, and a superscript ^{T} indicates transpose.)

Theorem 6.3: *Let $\Sigma = (A \times B, X, f)$ be an adversarially detectable asynchronous input/state machine with state set $X = \{x^1, x^2, \ldots, x^n\}$, adversarial uncertainty Ω, adversarial skeleton matrix $K^a(\Sigma, \Omega)$, and control skeleton matrix $K^c(\Sigma, \Omega)$. Then, the following two statements are equivalent*:

(i) *There is an automatic controller that counteracts every adversarial transition of Σ in fundamental mode operation.*
(ii) $(K^a(\Sigma, \Omega))^T \le K^c(\Sigma, \Omega)$.

In view of (17), the construction of the control skeleton matrix requires finding all pairs of states that can be connected by detectable feedback paths. The following algorithm describes a process for determining whether there is a detectable feedback path between two given states. The algorithm also derives such a detectable feedback path, whenever one exists. Following the process described in the proof of Theorem 5.3, the derived feedback path can be used to construct a controller that counteracts the corresponding adversarial transition.

Algorithm 1: Let $\Sigma = (A \times B, X, f)$ be an asynchronous input/state machine with state set $X = \{x^1, x^2, \ldots, x^n\}$, stable recursion function s, adversarial uncertainty Ω and matrix of stable transitions $R(\Sigma, \Omega)$. For a pair of integers $i, j \in \{1, 2, \ldots, n\}$, let $U_{ij}(\Sigma)$ be the set of control input characters (16). Given a string $\alpha \in R_{ij}(\Sigma, \Omega)$, denote by $b_1^p(\alpha)$ the string of bursts generated by the machine Σ during steps $1, 2, \ldots, p$, when driven by the string α. Then, for a member $u_0 \in U_{ij}(\Sigma)$, the following steps yield a detectable feedback path $S_{u_0} \subset R_{ij}(\Sigma, \Omega)$ with the initial control input character u_0, if one exists.

Step 0: Let S be the set of all members $w | u \in R_{ij}(\Sigma, \Omega)$ with initial control input character u_0. Denote by $S(u_0 u_1 \cdots u_t)$ the set of all members of S whose control input string has the prefix $u_0 u_1 \cdots u_t$. Also, let $S(u_0 u_1 \cdots u_t; b_1^t)$ be the set of all members of $S(u_0 u_1 \cdots u_t)$ for which the machine Σ generates the burst string b_1^t. Denote by $B_1^t(S(u_0 u_1 \cdots u_t))$ the set of all burst strings that Σ generates when driven by members of $S(u_0 u_1 \cdots u_t)$; here, $B_1^t(S(u_0 u_1 \cdots u_t)) := \varnothing$ when $t = 0$. Finally, let $|S|$ be the length of the longest member of S, where $|S| := 0$ when $S = \varnothing$. (Note that, according to Definition 5.4, we have $|S| \le n(\#\Omega)$.)

Step 1: If S is the empty set, then go to Step 8; otherwise, set $q := 0$ and $S' := S$.

Step 2: Set $B := B_1^q(S'(u_0 u_1 \cdots u_q))$.

Step 3: Select a burst string $b_1^q \in B$ and a string $\alpha \in S'(u_0 u_1 \cdots u_q; b_1^q)$; use (12) to calculate the adversarial uncertainty $\upsilon_q(\alpha)$. Denote $\upsilon := \Pi^a S'(u_0 u_1 \cdots u_q; b_1^q)$.

Step 4: If $\upsilon_q(\alpha) \not\subseteq \upsilon$, then replace S by the difference set $S \setminus S'(u_0u_1 \cdots u_q; b_1^q)$ and return to Step 1.
Step 5: If $q = |S'|$ and $B = \varnothing$, then go to Step 8.
Step 6: Perform the following operations: select a character $u_{q+1} \in \Pi_{q+1}^c S'(u_0u_1 \cdots u_q; b_1^q)$; remove from S' all elements $\sigma \in S'(u_0u_1 \cdots u_q; b_1^q)$ for which $\Pi_{q+1}^c \sigma \neq u_{q+1}$; replace B by the difference set $B \setminus \{b_1^q\}$.
Step 7: If $B = \varnothing$, then replace q by $q+1$ and go to Step 2; otherwise, go to Step 3.
Step 8: Set $S_{u_0} := S'$ and terminate the algorithm.

An examination of the flow of Algorithm 1 shows that the following is true.

Proposition 6.4: *Let $\Sigma = (A \times B, X, f)$ be an asynchronous machine with state set $X = \{x^1, x^2, \ldots, x^n\}$. For a pair of integers $i, j \in \{1, 2, \ldots, n\}$, assume that Σ underwent an adversarial transition from x^j to x^i with control input character $u_0 \in U_{ij}(\Sigma)$, where $U_{ij}(\Sigma)$ is given by* (16). *Finally, let S_{u_0} be the outcome of Algorithm* 1. *Then, the following two statements are equivalent.*

(i) *There is a detectable feedback path from the state x^i to the state x^j with initial control input character u_0.*
(ii) $S_{u_0} \neq \varnothing$.

Furthermore, when not the empty set, S_{u_0} forms a detectable feedback path from x^i to x^j with initial control input character u_0.

We demonstrate now the use of Algorithm 1.

Example 6.5: Assume that the machine Σ of Example 3.2 has experienced an adversarial transition from the state x^2 to the state x^3. We search for a detectable feedback path back from x^3 to x^2. By Example 6.1, we have $U_{32}(\Sigma) = \{a\}$, so the only possible initial control input character is $u_0 = a$. Apply now Algorithm 1:

Step 0: An examination of the transition diagram of Σ (Figure 2) yields $S = \{\beta|ac, \beta|aba\}$.
Step 1: Clearly, $S \neq \varnothing$, so we set $q := 0$ and $S' = S$.
Step 2: As $q = 0$, we obtain $B = B_1^0(S'(a)) = \varnothing$.
Step 3: Since $B = \varnothing$, the only burst in B is $b_1^0 = \varnothing$, and we have $S'(a; b_1^0) = S'(a; \varnothing) = S'$. Select the string $\alpha := \beta|ac \in S'$; from (12), we obtain $\upsilon_0(\alpha) = \beta$. Note that $\upsilon = \Pi^a S'(a; \varnothing) = \{\beta\}$.
Step 4: Since $\upsilon_0(\alpha) \subset \upsilon$, we advance to Step 5.
Step 5: Since $q = 0 \neq |S'| = 3$, we advance to Step 6.
Step 6: As $\Pi_1^c S'(a; \varnothing) = \{c, b\}$, we can select the next character of the control input string $u_1 := c$. Further, since $\Pi_1^c \beta|aba = b \neq c$, we remove $\beta|aba$ from S', so that $S' = \{\beta|ac\}$. Note that we still have $B = \varnothing$.
Step 7: Since $B = \varnothing$, set $q = 1$ and return to Step 2.

Repeating the algorithm steps as necessary leads to the result $S_{u_0} = \{\beta|ac\}$. Since $S_{u_0} \neq \varnothing$, the latter forms a detectable feedback path from x^3 to x^2 according to Proposition 6.4.

Repeated use of Algorithm 1 allows us to build the control skeleton matrix $K^c(\Sigma, \Omega)$ of the machine Σ. Then, we can use Theorem 6.3 to determine whether or not all possible adversarial transitions of the machine Σ can be counteracted by an automatic controller. This resolves the problem of automatically counteracting the effects of adversarial interventions on asynchronous input/state machines. We conclude our discussion with an example of controller construction.

Example 6.6: Consider again the machine Σ of Example 3.2. By Example 6.5, we have $K_{32}^c(\Sigma, \Omega) = 1$. Similarly, applying Algorithm 1 to the remaining state transitions, we obtain the control skeleton matrix

$$K^c(\Sigma, \Omega) = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

The adversarial skeleton matrix $K^a(\Sigma, \Omega)$ of the machine Σ was calculated in Example 3.10. Comparing the two matrices, we obtain $(K^a(\Sigma, \Omega))^T \leq K^c(\Sigma, \Omega)$. Hence, by Theorem 6.3, there is a controller that automatically counteracts every adversarial transition that might occur in Σ.

Controller construction follows the process described in the proof of Theorem 5.3. As an example of this process, we construct $C(2, 1)$, a controller that automatically counteracts adversarial transitions from x^1 to x^2. In view of Figure 1, the controller $C(2, 1)$ is an asynchronous machine with two inputs – one receives state bursts of the machine Σ and the other receives the external input character v. Thus, the input alphabet of $C(2, 1)$ is $X^* \times A$. As $C(2, 1)$ controls the machine Σ, its output alphabet is the input alphabet A of Σ, and we can write $C(2, 1) = (X^* \times A, A, \Xi, \xi^0, \phi, \eta)$, where Ξ is the state set, ξ^0 is the initial state, $\phi : \Xi \times X^* \times A \to \Xi$ is the recursion function, and $\eta : \Xi \times X^* \times A \to A$ is the output function.

From Figure 2, it follows that an adversarial transition from the state x^1 to the state x^2 can occur only in the presence of the control input character $u_0 = a$. The external input character v of the closed loop machine of Figure 1 is then a during the adversarial transition, since the controller's only purpose here is to counteract adversarial transitions. For the same reason, a is the initial control input character produced by $C(2, 1)$.

We turn now to the construction of the recursion function and of the output function of $C(2, 1)$.

Initially, $C(2,1)$ is at the state ξ^0. The controller remains in its initial state until it detects a stable combination with the pair (x^1, a), so we set

$$\phi(\xi^0, x, u) := \xi^0 \text{ whenever } (x,u) \neq (x^1, a).$$

While at the initial state ξ^0, the controller is transparent and applies to Σ the input character it receives:

$$\eta(\xi^0, \beta, u) := u \text{ for all } \beta \in X^* \text{ and all } u \in A.$$

Next, $C(2,1)$ moves to the state ξ_1 upon detection of a stable combination with the pair (x^1, a), in preparation for a potential adversarial transition of Σ from x^1 to x^2. To implement this move, set

$$\phi(\xi^0, x^1, a) := \xi_1;$$

at this point, $C(2,1)$ continues to apply the input character a to Σ, so we set

$$\eta(\xi_1, x^1, a) := a.$$

In the course of an adversarial transition from x^1 to x^2, the machine Σ generates the burst x^3x^2 (see Figure 2; the adversarial input switches from α to γ). Upon the detection of this burst, $C(2,1)$ moves to the state ξ_2. To this end, set the recursion function to

$$\phi(\xi_1, x^3x^2, a) := \xi_2.$$

Now, use the detectable feedback path $\{\gamma|ab\}$ of Example 5.2 to undo the adversarial transition and return the machine Σ to the state x^1. As $C(2,1)$ already applies the initial control input character a, it only remains to apply the control input character b to Σ to complete implementing this detectable feedback path. To this end, set

$$\eta(\xi_2, \beta, a) := b \text{ for all } \beta \in X^*.$$

An examination of Figure 2 shows that this control input character returns Σ to the state x^1 with the burst x^1. At this point, the controller $C(2,1)$ has completed counteracting the adversarial transition; we can leave it in its last state:

$$\phi(\xi_2, \beta, a) := \xi_2 \text{ for all } \beta \in X^*.$$

This completes the construction of the controller $C(2,1)$. Similarly, controllers can be constructed to automatically counteract each possible adversarial transition of the machine Σ. These controllers can then be combined into a single controller that automatically counteracts any adversarial transitions of Σ (see Yang and Hammer (2008a) for a description of the process of combining controllers). □

7. Conclusion

To summarise, we have provided necessary and sufficient conditions for the existence of state feedback controllers that automatically counteract the effects of adversarial interventions on the operation of asynchronous sequential machines. These conditions are presented in terms of an inequality between the entries of two numerical matrices of zeros and ones, matrices that are derived directly from data about the machine that must be protected. Whenever such controllers exist, an algorithm for their design was also put forward.

In general terms, the existence of controllers that counteract adversarial interventions depends on certain reachability and detectability properties of the controlled machine. The ability of a controller to detect adversarial interventions is enhanced in this article by the use of bursts. This enhanced ability of detection strengthens the capabilities of controllers to automatically counteract adversarial interventions.

Acknowledgments

The research of Jung-Min Yang was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology under grant number KRF-2008-521-D00264.

References

Barrett, G., and Lafortune, S. (1998), 'Bisimulation, the Supervisory Control Problem, and Strong Model Matching for Finite State Machines', *Discrete Event Dynamic Systems: Theory and Application*, 8, 377–429.

Dibenedetto, M.D., Saldanha, A., and Sangiovanni-Vincentelli, A., (1994), 'Model Matching for Finite State Machines', in *Proceedings of the IEEE Conference on Decision and Control*, Lake Buena Vista, FL, 3, 3117–3124.

Geng, X.J., and Hammer, J. (2004), 'Asynchronous Sequential Machines: Input/Output Control', *Proceedings of the 12th Mediterranean Conference on Control and Automation*, Kusadasi, Turkey, June 2004.

Geng, X.J., and Hammer, J. (2005), 'Input/Output Control of Asynchronous Sequential Machines', *IEEE Transactions on Automatic Control*, 50, 1956–1970.

Hammer, J. (1994), 'On Some Control Problems in Molecular Biology', *Proceedings of the IEEE Conference on Decision and Control*, Lake Buena Vista, FL, pp. 4098–4103, December 1994.

Hammer, J. (1995), 'On the Modeling and Control of Biological Signal Chains', *Proceedings of the IEEE Conference on Decision and Control*, New Orleans, LA, pp. 3747–3752, December 1995.

Hammer, J. (1996a), 'On the Corrective Control of Sequential Machines', *International Journal of Control*, 65, 249–276.

Hammer, J. (1996b), 'On the Control of Incompletely Described Sequential Machines', *International Journal of Control*, 63, 1005–1028.

Hammer, J. (1997), 'On the Control of Sequential Machines with Disturbances', *International Journal of Control*, 67, 307–331.

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

Murphy, T.E., Geng, X.J., and Hammer, J. (2002), 'Controlling Races in Asynchronous Sequential Machines', *Proceedings of the IFAC World Congress*, Barcelona, July 2002.

Murphy, T.E., Geng, X.J., and Hammer, J. (2003), 'On the Control of Asynchronous Machines with Races', *IEEE Transactions on Automatic Control*, 48, 1073–1081.

Peng, J., and Hammer, J. (2008), 'Bursts and Output Feedback Control of Non-deterministic Asynchronous Sequential Machines' (in preparation).

Peng, J., and Hammer, J. (2010), 'Input/Output Control of Asynchronous Sequential Machines with Races', *International Journal of Control*, 83, 125–144.

Ramadge, P.J.G., and Wonham, W.M. (1987), 'Supervisory Control of a Class of Discrete Event Processes', *SIAM Journal of Control and Optimization*, 25, 206–230.

Thistle, J.G., and Wonham, W.M. (1994), 'Control of Infinite Behavior of Finite Automata', *SIAM Journal on Control and Optimization*, 32, 1075–1097.

Venkatraman, N., and Hammer, J. (2006a), 'Stable Realizations of Asynchronous Sequential Machines with Infinite Cycles', *Proceedings of the 2006 Asian Control Conference*, Bali, Indonesia, 2006.

Venkatraman, N., and Hammer, J. (2006b), 'Controllers for Asynchronous Machines with Infinite Cycles', *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems*, Kyoto, Japan, 2006.

Venkatraman, N., and Hammer, J. (2006c), 'On the Control of Asynchronous Sequential Machines with Infinite Cycles', *International Journal of Control*, 79, 764–785.

Yang, J.-M., and Hammer, J. (2008a), 'State Feedback Control of Asynchronous Sequential Machines with Adversarial Inputs', *International Journal of Control*, 81, 1910–1929.

Yang, J.-M., and Hammer, J. (2008b), 'Counteracting the Effects of Adversarial Inputs on Asynchronous Sequential Machines', *Proceedings of the IFAC World Congress*, Seoul, Korea, pp. 1432–1437, July 2008.