INT. J. CONTROL, 1996, VOL. 63, NO. 6, 1005–1028

On the control of incompletely described sequential machines

JACOB HAMMER†

This paper deals with the control of a sequential machine whose model is not precisely known. The objective is to design a controller that assigns to the machine a prescribed steady-state behaviour. The results include necessary and sufficient conditions for the existence of a controller, as well as algorithms for its design. This work is motivated by potential applications in biology and in the correction of malfunctioning computer systems.

1. Introduction

Often in applications there arises a need to control a sequential machine whose model is not completely known. The lack of knowledge about the model may originate either from a genuine lack of data about the exact characteristics of the machine, or from the desire to control an entire class of somewhat differing machines by a single controller design. Examples of applications where both situations are eminently relevant include sequential machines that model processes in molecular biology; and sequential machines that model malfunctioning complex computer systems. The present discussion is mainly motivated by potential applications in molecular biology (see Hammer 1993 and the references cited there for more details), but the mathematical formalism applies equally well to other areas.

Consider a sequential machine Σ whose model is only partly known. The main subject of the present paper is the development of controllers that automatically steer Σ toward a prescribed steady-state course, despite the uncertainty about the model of Σ.

The information available about the machine Σ is given in the form of a family M of potential models; the real model of Σ is one of the members of M. Clearly, the larger the class M, the larger is the uncertainty about the exact characteristics of Σ. The class M may represent a true uncertainty about the exact model of Σ; Alternatively, it may contain a class of sequential machines all of whose members need to be controlled by a single controller design. The two interpretations lead to identical mathematical formulations.

An important concern is to reduce the amount of data that needs to be collected about the sequential machine Σ. The data about Σ can be divided into two broad categories: *a priori* data, which determine the class M of potential models of Σ, and are collected prior to the initiation of the control process; and real-time data, which consist of data the controller requires during its operation in feedback form, and are collected by the controller during its operation.

It is particularly important to reduce the real-time data requirements, so as to avoid complex measurements in real time. This is especially critical for applications in molecular biology, where one must strive to reduce the number of chemical tests

Received 20 November 1994. Revised 18 February 1995. Communicated by Professor A. Isidori.

† Center for Mathematical System Theory, Department of Electrical Engineering, University of Florida, Gainseville, FL 32611-6130, U.S.A.

required in real time. Furthermore, one must entirely avoid lengthy chemical tests that cannot be completed within the time allotted to one controller step. There is, of course, an interplay between the *a priori* data given about Σ and the amount of real-time data that need to be collected. Generally speaking, a smaller class M (i.e. more accurate *a priori* data) reduces the requirements for real-time data.

In more specific terms, consider a sequential machine Σ that can be described by a recursive model of the form

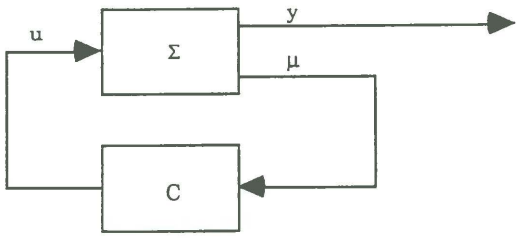
$$\begin{aligned} \sigma_{k+1} &= \phi(\sigma_k, u_k), \\ y_k &= h_0(\sigma_k, u_k), \quad k = 0, 1, 2, \ldots \end{aligned} \tag{1.1}$$

Here, ϕ and h_0 are functions; σ_k is the state of the machine at the step k; u_k is the input value at that step, and y_k is the output value. The initial condition σ_0 is given. The class M of potential models of Σ consists then simply of a family $\{\phi_i, h_{io}, \sigma_{i0}\}_{i=1,2,\ldots}$ of possible functions and initial conditions.

The output value y_k of Σ signifies the 'outcome' of the machine's operation, and is not necessarily available to the controller for feedback use. The real-time data needed by the controller is created through a *monitoring function* h_m that generates a *monitoring sequence* μ_k given by

$$\mu_k = h_m(\sigma_k, u_k), \quad k = 0, 1, 2, \ldots \tag{1.2}$$

The function h_m indicates measurements that need to be performed in real-time to generate a feedback signal for the controller. It is designed together with the controller, so as to create all the feedback data necessary for the closed loop



(1.3)

Here, the controller C uses feedback information derived from the monitoring sequence μ to generate an input sequence u that steers Σ toward a prescribed steady-state course. The controller C automatically generates the input sequence that steers Σ to the desired course, and there is no need for an external reference signal for the loop. Such a controller C is called an *autonomous controller*.

The monitoring function h_m indicates the real-time data that need to be collected about Σ. It is of great importance to design h_m so as to reduce the complexity measurements done in real time. This is particularly critical for applications in molecular biology, where the real-time detection of certain compounds may be impractical, if not impossible. The nature of the monitoring function h_m determines, to a large extent, whether a given design is practical or not.

In general, it is not possible to characterize mathematically a monitoring function that calls for a 'minimal' measurement complexity. The issues involved in determining whether a particular measurement is complex or not are, in many cases, rather qualitative. Consider, for instance, the case of molecular biology. Here, the presence of certain compounds may be relatively easy to determine, whereas the presence of

other compounds may be difficult, or even impractical, to establish. The difficulty of establishing the presence of a compound depends not only on the compound itself, but also on the presence or absence of other compounds within the monitored medium. The complexity of obtaining the real time data is not necessarily determined by the number of measurements that are required, but rather by the nature of each individual measurement.

In view of these considerations, it seems preferable to characterize all possible pairs (h_m, C) of monitoring functions and controllers that facilitate the achievement of the control objective. The designer can then select the most convenient pair. In many cases, the most important aspect is the selection of the monitoring function h_m, since, to a large extent, it determines the complexity of implementation. In our framework, h_m can be selected prior to the selection of the controller C, which is selected later based on the choice of h_m (see §4.2). Of course, all this is under the assumption that the control objective is achievable, and an appropriate pair (h_m, C) exists. Necessary and sufficient conditions for the latter are provided in §3 and 4 of the paper, together with the relevant construction algorithms.

The controllers derived in the paper comprise all possible controllers, and include controllers that exhibit 'adaptive' or 'learning' characteristics. The approach taken in the paper treats all possible controllers uniformly and under the same framework.

Finally, we comment that the present paper deals with deterministic control. We do not assign probabilities to the various potential models of the sequential machine Σ, and the controller is required to steer the system to the desired goal in all cases.

The basic formalism used in the paper is taken from Hammer (1993). The relevance of sequential machine models to various biological phenomena is considered by Rashevsky (1948), Sugita (1963), von Neumann (1966), Lindenmayer (1968), Kauffman (1969), Rosenberg and Salomaa (1975), IEEE (1974), Alberts *et al.* (1989), the reference listed in these works, and many others. Background on topics related to automata theory and discrete-event systems can be gained from Ginsburg (1962, 1966), Eilenberg (1974), Hoare (1976), Milner (1980), Arnold and Nivat (1980), Ramadge and Wonham (1987), the references cited in these works, and many other excellent sources.

The paper is organized as follows. Section 2 contains a review and refinement of those parts of the framework developed by Hammer (1993) that are relevant to the present discussion. Section 3 contains necessary and sufficient conditions for the existence of monitoring function-controller pairs that steer a machine Σ having a family M of potential models toward a prescribed goal. The paper concludes with §4, which contains search-based algorithms for the selection of monitoring function-controller combinations that achieve the control objective for Σ.

2. Basic notions and properties

2.1. *Interpreters and controllers*

Let A be a non-empty alphabet, and let A^* be the set of all words over A. A *sentence* over A is any (possibly empty) collection of words from A^*. Not all words in a sentence must be distinct; multiple copies of words are allowed. The set of all sentences over the alphabet A is denoted by S_A. The *cardinality* $\#s$ of a sentence $s \in S_A$ is the total number of words in s, counting each word according to its multiplicity. Multiple copies of a word in a sentence are needed for potential applications in molecular biology

(see Hammer, 1993 for a discussion of this point). When combining two sentences $s_1, s_2 \in S_A$ into a union, all copies of similar words need to be preserved. This is accomplished by the use of the disjoint union $s_1 \dot{\bigcup} s_2$, which includes all copies of all words contained in s_1 or in s_2.

Let $S(S_A)$ be the set of all sequences of sentences $s_0, s_1, s_2, \ldots$, where $s_i \in S_A$ for all integers $i \geqslant 0$. For a sequence $s \in S(S_A)$, we denote by s_i the ith element of the sequence, $i = 0, 1, \ldots$. The index i serves here as a step counter; a step may or may not be linked to a specific time duration. It is convenient to use the notation s_i^j for the list of sentences $s_i, s_{i+1}, \ldots, s_j$ when $j \geqslant i$; when $j < i$, we define $s_i^j := \varnothing$, the empty set. The disjoint union $s \dot{\bigcup} u$ of two sequences of sentences $s, u \in S(S_A)$ is a sequence $v \in S(S_A)$ given by $v_k := s_k \dot{\bigcup} u_k$, $k = 0, 1, 2, \ldots$.

The paper deals with the control of interpreters, which form a class of generalized sequential machines, and are defined as follows. Let $D \subset S(S_A)$ be a subset. An *interpreter* is a map $\Sigma : D \to S(S_A) \times S(S_A)$; it accepts input sequences $u \in D$, and generates pairs of output sequences $(y, \mu) \in S(S_A) \times S(S_A)$. The sequence y is called the *output sequence* of Σ, and μ is called the *monitored sequence* of Σ. The output sequence y describes the interpreter's end products. The monitored sequence μ describes quantities generated by Σ that are being measured at each step, and are available for feedback use in the closed loop control configuration (1.3). We shall write $y := \Sigma_o u$ and $\mu := \Sigma_m u$, so that $\Sigma = (\Sigma_0, \Sigma_m)$. For notational brevity, we set $SS_A := S(S_A) \times S(S_A)$.

An interpreter $\Sigma : D \to SS_A$ is *causal* (respectively, *strictly casual*) if, for every pair of input sequences $u, v \in D$ and for every integer $k \geqslant 0$, the equality $u_0^k = v_0^k$ implies that $(\Sigma u)_0^k = (\Sigma v)_0^k$ (respectively, $(\Sigma u)_0^{k+1} = (\Sigma v)_0^{k+1}$). Of course, only causal interpreters appear in real-time applications.

In order to control a causal interpreter $\Sigma : D \to SS_A$, we combine it with a controller $C : \operatorname{Im} \Sigma_m \to D$, using the scheme

$$\begin{aligned} (y, \mu) &= \Sigma u \\ u &= C\mu \end{aligned} \tag{2.1.1}$$

depicted in (1.3). The controller C is called an *autonomous controller* since it performs all necessary control operations on its own, and the closed loop system requires no prompting by an external reference signal. We shall denote the input/output map induced by the closed loop configuration of (2.1.1) by Σ_c. The system Σ_c has no input sequence, and is therefore called an *autonomous interpreter* $\Sigma_c : \varnothing \to SS_A$.

The controller C must, of course, be causal. In order to simplify our discussion, we shall impose on C the stronger requirement of strict causality. When C is strictly causal, the input sequence u of Σ is uniquely determined by (2.1.1), and the closed loop system (1.3) is well posed, as indicated by the following (see Hammer 1993).

Lemma 2.1.2: *Let $\Sigma : D \to SS_A$ be a causal interpreter. Then, for every strictly causal autonomous controller $C : \operatorname{Im} \Sigma_m \to D$, equation* (2.1.1) *uniquely determines the input sequence u of the interpreter Σ.*

The requirement that the controller C be strictly causal is imposed for convenience only. It can be replaced by plain causality combined with the requirement that u be uniquely determined by (2.1.1). However, from an implementation perspective, the requirement that C be strictly causal is more comfortable; it creates a one-step delay between the procurement of the feedback data and the initiation of its effect on the controller output. This allows more time for measurement and processing, a fact that

may be critical for some applications in molecular biology. The response of a controller that is not strictly causal may need to be adjusted 'instantaneously' at a step, based on data collected during that same step.

As is always the case when dealing with asynchronous processes, the controller C needs to be synchronized with the interpreter Σ it controls, so that the outputs of C appear at the corresponding steps of Σ. This is mainly an implementation issue, and is not elaborated upon in the present discussion.

2.2. *Recursive models of interpreters*

The interpreters we consider are described by recursive models of the following form (Hammer 1993)

$$\left.\begin{aligned}(s_{k+1}, x_{k+1}) &= f[(s_k \dot{\bigcup} u_k), x_k] \\ y_k &= h_o(s_k \dot{\bigcup} u_k), \\ \mu_k &= h_m(s_k \dot{\bigcup} u_k), \quad k = 0, 1, 2, \ldots\end{aligned}\right\} \tag{2.2.1}$$

Here, $f: S_A \times X \to S_A \times X: (s, x) \mapsto (f_s(s, x), f_x(s, x))$ is called the *recursion function*, $h_o: S_A \to S_A$ is the *output function*, and $h_m: S_A \to \Delta$ is the *monitoring function*. The sets X and Δ are finite and non-empty. The recursion is started from a given initial condition $\sigma_0 := (s_0, x_0) \in S_A \times X$, and induces an interpreter $\Sigma: D \to SS_A$. An interpreter $\Sigma: D \to SS_A$ that permits a representation of the form (2.2.1) is called a *recursive interpreter*.

In our present framework, the monitoring function h_m is not given; it is designed together and in harmony with the controller, to supply the feedback data needed for the controller's operation. Hence, we shall regard Σ as a map $\Sigma: D \to S(S_A)$, and the monitoring function will be supplied in the design process. The set Δ is a set within which the measurement results produced by the monitoring function h_m are interpreted; an element of Δ indicates a (code of a) possible outcome of a measurement.

The set X consists of the *states* of the interpreter Σ, and the pair (s_k, x_k) constitutes the *status* of the interpreter Σ at the step k. The sequence $s_0, s_1, \ldots$ is called the *internal sequence* of the interpreter, and the sequence $s \dot{\bigcup} u$ is called the *medium sequence* of Σ; the *medium value* at the step k is then simply $s_k \dot{\bigcup} u_k$. The intuitive motivation for this terminology originates from potential applications in molecular biology, and is discussed in detail by Hammer (1993). We shall denote by $\Pi_s: S_A \times X \to S_A: (s, x) \mapsto s$ the standard projection onto the internal value.

A set of sequences $D \subset S(S_A)$ is a *uniform set* if there is a set of sentences $V(D) \subset S_A$ such that D is the set of all sequences of elements of $V(D)$. When the latter holds, we call $V(D)$ the *value set* of D (see Hammer 1993 for details).

2.3. *Families of interpreters*

Our present discussion is concerned with the case where an accurate description of the recursive interpreter Σ, which needs to be controlled, is not available. Instead, we are given a family $M = \{(f_1, h_{1o}, \sigma_{10}), (f_2, h_{2o}, \sigma_{20}), \ldots, (f_q, h_{qo}, \sigma_{q0})\}$ of potential models of the interpreter Σ, each one with its own initial condition. Here, $f_1, \ldots, f_q$ are potential recursion functions; $h_{1o}, \ldots, h_{qo}$ are potential output functions; and $\sigma_{10}, \ldots, \sigma_{q0}$ are potential initial conditions. Without loss of generality, we assume that the functions $f_1, \ldots, f_q$ all share a common domain and a common codomain, and similarly for the

functions $h_{1o}, \ldots, h_{qo}$. The monitoring function for Σ is to be computed as part of the controller design. Of course, the set M may include several copies of the same recursion function and output function pairs, with each pair having its own distinct initial status. In other words, we do not require exact knowledge of the initial status of any potential model of Σ.

In each case, only one of the potential models of Σ is present; we refer to this model of Σ as the *active model*. The identity of the active model is not known in advance, except for the information that it is a member of the family M. The controller has to be designed so as to be capable of dealing with the indeterminacy about the active model, whenever possible. Designing such controllers is, in brief, the main subject of the paper.

2.4. *Tail sets*

Let M be a family of potential models of the interpreter Σ. Our objective is to develop techniques for the design of autonomous controllers that assign a prescribed steady-state response to Σ, despite the indeterminacy about its model. To state this objective in more precise terms, we need some terminology and notation. We start with a discussion of what constitutes a 'steady-state response'.

Let $k \geqslant 0$ be an integer, and let $s := \{s_k, s_{k+1}, s_{k+2}, \ldots\}$, where $s_i \in S_A, i \geqslant k$, be a sequence of sentences. A *tail* of the sequence s is any subsequence $\{s_j, s_{j+1}, \ldots\}$, where $j \geqslant k$ is an integer. A *tail set* is a set that consists of tails of sequences. A tail set is *complete* if it contains all the tails of each one of its elements. For example, the complete tail set $T(s)$ of a sequence $s \in S(S_A)$ is given by

$$T(s) := \bigcup_{k \geqslant 0} s_k^{\infty}$$

For a set of sequences $S \subset S(S_A)$, we denote by

$$T(S) := \bigcup_{s \in S} \bigcup_{k \geqslant 0} s_k^{\infty}$$

the complete tail set induced by the set S. For the autonomous interpreter Σ_c: $\varnothing \to S(S_A)$ that describes the configuration (1.3), we denote by $T(\Sigma_c)$ the complete tail set of its (single) output sequence.

Given two tail sets T_1 and T_2, we define the intersection $T_1 \cap T_2$ as the set of all sequences $z := z_0, z_1, \ldots$ that satisfy the following property: there is a pair of integers t, $\tau \geqslant 0$ such that the sequence $v_t := z_0, v_{t+1} := z_1, v_{t+2} := z_2, \ldots$ belongs to T_1, and the sequence $w_\tau := z_0, w_{\tau+1} := z_1, w_{\tau+2} := z_2, \ldots$ belongs to T_2. In other words, shifted versions of the sequence z are found in T_1 and in T_2.

Finally, given a list of elements $\theta_1, \ldots, \theta_r \in S_A$, we denote by $\theta_1^{r*} = (\theta_1, \ldots, \theta_r)^*$ the periodic sequence $\theta_1, \ldots, \theta_r, \theta_1, \ldots, \theta_r, \ldots, \theta_1, \ldots, \theta_r, \ldots$. The list $\theta_1, \ldots, \theta_r$ then becomes a cycle of the periodic sequence θ_1^{r*}. By $T(\theta_1^{r*})$ we denote the complete tail set generated by all (cyclic permutation) sequences $(\theta_1, \ldots, \theta_r)^*, (\theta_2, \ldots, \theta_r, \theta_1)^*, \ldots, (\theta_r, \theta_1, \ldots, \theta_{r-1})^*$. A *periodic tail set* is a union of a finite number of complete tail sets of periodic sequences.

2.5. *Statement of the problem*

Consider again the interpreter Σ having the family

$$M = \{(f_1, h_{1o}, \sigma_{1o}), \ldots, (f_q, h_{qo}, \sigma_{qo})\}$$

of potential models. With each potential model of Σ, we associate a periodic tail set $T_i, i = 1, \ldots, q$, that describes the prescribed steady-state behaviour of Σ, in case model number i is the active model. We shall refer to T_i as the *target tail set* of the potential model i. It will be convenient to denote by

$$T := T_1 \times T_2 \times \cdots \times T_q$$

the cross product of all target sets, and to refer to it as the target tail set of the interpreter Σ.

Recall that, at the outset, a monitoring function is not provided for the interpreter Σ. The monitoring function is to be determined as part of the design process, in conjunction with a controller. All potential models of Σ share one and the same monitoring function h_m, referred to as the 'monitoring function of Σ'. For each potential model of Σ, we shall denote by Σ_i the interpreter generated by the model $(f_i, h_{io}, \sigma_{io})$. The main question discussed in this paper can then be phrased as follows.

2.5.1. *Autonomous control of the interpreter Σ having the family M of potential models.* Find a monitoring function h_m and a strictly causal autonomous controller C such that the closed loop system Σ_{ic} satisfies $T(\Sigma_{ic}) \cap T_i \neq \varnothing$ for all $i = 1, \ldots, q$.

In this way, the control objective is achieved without regard as to which potential model of Σ is active. If a monitoring function h_m and controller C satisfying the above exist, we say that C *steers* Σ to the target tail set T.

As discussed by Hammer (1993) the fact that the tail sets $T_1, \ldots, T_q$ consist of a finite number of periodic sequences guarantees that the controller C, whenever it exists, is finite dimensional (and hence implementable).

Finally, we comment that due to the strict causality of the controller C, the output value generated by C for step number zero must be the same for all potential models of Σ, and cannot depend on the monitored sequence.

2.5.2. *Basic assumption.* In this paper, we consider only recursive interpreters $\Sigma: D \to S(S_A)$ that satisfy the following requirements: (i) the domain D is a uniform set with a finite value set; (ii) all potential recursion functions of Σ have a finite image; and (iii) the target tail set of Σ consists of a finite number of periodic sequences. Interpreters that satisfy these conditions are called *bounded interpreters*.

3. The existence of monitoring functions and controllers

In the present section we derive necessary and sufficient conditions for the existence of monitoring function-controller combinations that steer a recursive bounded interpreter Σ to a prescribed target tail set. The interpreter Σ has a family M of potential models. We start with a few preliminary issues.

3.1. *Target sets in status space*

Let $M = \{(f_1, h_{1o}, \sigma_{10}), \ldots, (f_q, h_{qo}, \sigma_{q0})\}$ be the family of potential models of the bounded interpreter $\Sigma: D \to S(S_A)$. Recall that with each potential model $(f_i, h_{io}, \sigma_{i0})$, there is associated a periodic target tail set $T_i, i = 1, \ldots, q$. We now translate the information contained in the target tail sets $T_1, \ldots, T_q$ into quantities in status space. For this purpose it will be convenient to regard the functions f_i and h_{io} as functions over the domain $S_A \times V(D) \times X$ (rather than $S_A \times X$ or S_A only), and we shall write $f_i(s_k, x_k, u_k)$ rather than $f_i[(s_k \bigcup u_k, x_k]$, and $h_{io}(s_k, u_k, x_k)$ rather than $h_{io}(s_k \bigcup u_k)$.

We have then $h_{io}: S_A \times V(D) \times X \to S_A$, and we denote by $P(S_A)$ the set of all subsets of S_A. Let $h_{io}^{-1}: P(S_A) \to S_A \times V(D) \times X$ be the inverse-set function induced by the output function h_{io}; i.e. for every subset $B \subset S_A$, the set $h_{io}^{-1}(B)$ consists of all points $a = (s, u, x) \in S_A \times V(D) \times X$ for which $h_{io}\, a = h_{io}(s \dot{\bigcup} u) \in B$. Given a set of lists $S \subset (S_A)^k$, where k is a positive integer, we denote by $h_{io}^{-1}(S)$ the set of all lists $(b_1, \ldots, b_k) \in (S_A \times V(D) \times X)^k$ for which $(h_{io}(b_1), \ldots, h_{io}(b_k)) \in S$.

We now construct the sets of sequences

$$\Theta_i := h_{io}^{-1}[T_i], \quad i = 1, \ldots, q \tag{3.1.1}$$

The set Θ_i is called the *internal target set* of the model $(f_i, h_{io}, \sigma_{i0})$, and its members are sequences of elements of $S_A \times V(D) \times X$. Note that even though T_i consists of periodic sequences only, the internal target set Θ_i will contain non-periodic sequences (in addition to periodic ones) whenever the output function h_{io} is not injective.

Let $\Pi_u: S_A \times V(D) \times X \to V(D): (s, u, x) \mapsto u$ be the standard projection onto the input value. For a sequence $\theta = (s_k, u_k, x_k), (s_{k+1}, u_{k+1}, x_{k+1}), \ldots$, we denote by $\Pi_u \theta$ the sequence $u_k, u_{k+1}, \ldots$.

Given two internal target sets Θ_i and Θ_j, we denote by $(\Theta_i, \Theta_j)_u$ the set of all pairs of sequences $(\theta_1, \theta_2) \in \Theta_i \times \Theta_j$ for which $\Pi_u \theta_1 = \Pi_u \theta_2$. In other words, $(\Theta_i, \Theta_j)_u$ consists of all pairs of internal target sequences having the same input sequence. An element $(\theta_1, \theta_2) \in (\Theta_i \times \Theta_j)_u$ is *periodic* if θ_1 and θ_2 are periodic sequences. The fact that the target tail sets $T_1, \ldots, T_q$ consist of periodic sequences leads to the following.

Lemma 3.1.1: *The set $(\Theta_i, \Theta_j)_u$ contains a periodic element whenever it is not empty*

Proof: Assume that $(\Theta_i, \Theta_j)_u$ is not empty, and let (θ_i, θ_j) be an element of $(\Theta_i, \Theta_j)_u$. Then, θ_i and θ_j must start at the same step, say k; and $\theta_i \in h_{io}^{-1}[T_i]$ and $\theta_j \in h_{io}^{-1}[T_j]$, or $h_{io}(\theta_i) \in T_i$ and $h_{jo}(\theta_j) \in T_j$. Since T_i and T_j consist of periodic sequences, it follows that $h_{io}(\theta_i)$ and $h_{jo}(\theta_j)$ are periodic sequences. Let m, n be the periods of the sequences $h_{io}(\theta_i)$ and $h_{jo}(\theta_j)$, respectively, and let $t \geqslant 0$ be a least common multiple of m and n. Then, clearly, $h_{io}((\theta_i]_k^{k+t-1})^*) = h_{io}(\theta_i)$ and $h_{jo}((\theta_j]_k^{k+t-1})^*) = h_{jo}(\theta_j)$, and the membership $(\theta_j, \theta_j) \in (\Theta_i, \Theta_j)_u$ implies that also $((\theta_i]_0^{t-1})^*, (\theta_j]_0^{t-1})^*) \in (\Theta_i, \Theta_j)_u$. Thus, $(\Theta_i, \Theta_j)_u$ contains the periodic element $((\theta_i]_0^{t-1})^*, (\theta_j]_0^{t-1})^*)$, and the proof concludes. □

In general, given r internal target sets $\Theta_{i(1)}, \ldots, \Theta_{i(r)}$, $r \in \{1, \ldots, q\}$, we denote by $(\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$ the set of all lists of sequences $(\theta_1, \ldots, \theta_r) \in \Theta_{i(1)} \times \cdots \times \Theta_{i(r)}$ for which $\Pi_u \theta_1 = \Pi_u \theta_2 = \cdots = \Pi_u \theta_r$, i.e. the set of all r-tuples of sequences that share a common input sequence; for $r = 1$, set $(\Theta_{i(1)})_u := \Theta_{i(1)}$. We call $(\Theta_i(1), \ldots, \Theta_{i(r)})_u$ the *joint target tail* of the class $c := \{(f_{i(1)}, h_{i(1)o}, \sigma_{i(1)0}), \ldots, (f_{i(r)}, h_{i(r)o}, \sigma_{i(r)0})\}$ of potential models, and denote it by $\Theta_u(c)$.

As before, an element $(\theta_1, \ldots, \theta_r) \in (\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$ is *periodic* if the sequences $\theta_1, \ldots, \theta_r$ are all periodic. The proof of Lemma 3.1.1 can be readily adapted to yield the following.

Corollary 3.1.1: *The set $(\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$ contains a periodic element whenever it is not empty.*

Consider again the family $M = \{(f_1, h_{1o}, \sigma_{10}), \ldots, (f_q, h_{qo}, \sigma_{q0})\}$ of potential models of the bounded interpreter Σ. For an integer $r \in \{1, \ldots, q\}$, we denote by N_r the class of all subsets $\{(f_{i(1)}, h_{i(1)o}, \sigma_{i(1)0}), \ldots, (f_{i(r)}, h_{i(r)o}, \sigma_{i(r)0})\} \subset M$ of r models for which $(\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u \neq \varnothing$. We set $N_r := \varnothing$ if $(\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u = \varnothing$. In intuitive terms, each element of

N_r consists of r models that share a common input sequence along certain paths within their internal target sets.

A subset of models $\omega \in N_r$ is *maximal* if there is no member $\omega' \in N_j$ with $j > r$ for which $\omega \subset \omega'$, i.e. if there is no other model that shares a common input sequence with all the models contained in ω. Let M_r be the class of all maximal elements of $N_r, r = 1, \ldots, q$, and set $M_r := \varnothing$ if N_r contains no maximal elements. Finally, define the *target compatibility class* $\mathscr{M}$ of M by

$$\mathscr{M} := \bigcup_{i=1,\ldots,q} M_i$$

Let $\phi = \{(f_{i(1)}, h_{i(i)0}, \sigma_{i(1)0}), \ldots, (f_{i(r)}, h_{i(r)0}, \sigma_{i(r)0})\} \in \mathscr{M}$ be a family of models, and consider an element $\theta := (\theta_1, \ldots, \theta_r) \in (\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$. Note that each $\theta_i, i = 1, \ldots, r$, is a sequence $\theta_i = (s_k(i), x_k(i), u_k(i))$, $(s_{k+1}(i), x_{k+1}(i), u_{k+1}(i))$, $(s_{k+2}(i), x_{k+2}(i), u_{k+2}(i)), \ldots$ that starts at a step $k \geqslant 0$, the latter being common to all $\theta_i, i = 1, \ldots, r$. Let

$$(s(\theta_i), x(\theta_i), u(\theta)) := (s_k(i), x_k(i), u_k(i))$$

be the first element of the sequence θ_i, where we used the fact that

$$(\theta_1, \ldots, \theta_r) \in (\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$$

implies that $u(\theta_1) = u(\theta_2) = \cdots = u(\theta_r) := u(\theta)$. Now, construct the set of vectors

$$T(\phi) := \bigcup_{\theta \in (\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u} (s(\theta_1), x(\theta_1), s(\theta_2), x(\theta_2), \ldots, s(\theta_r), x(\theta_r), u(\theta)) \qquad (3.1.2)$$

and set $T(\theta) := \varnothing$ for $\phi \notin \mathscr{M}$. The set $T(\phi)$ is called the *point target* set of the family ϕ, and it is a subset of $(S_A \times X)^r \times V(D)$. It consists of the first elements of all sequences in $(\Theta_{i(1)}, \ldots, \Theta_{i(r)})_u$. The significance of the point target set $T(\phi)$ stems from the fact that once the members of the family ϕ are brought to a point belonging to $T(\phi)$, they can all be kept within their respective target sets by applying the same common input sequence. The point target set $T(\phi)$ serves the important function of 'translating' a target set of tail sequences into a target set of points. Reaching a point within the point target set $T(\phi)$ then becomes the basic control objective, if it has been determined that the active model of Σ is a member of the family ϕ (and $T(\phi) \neq \varnothing$).

3.2. *Jointly reachable and jointly controllable sets*

Jointly reachable sets, defined below, play a critical role in our discussion. These sets describe the points in status space that can be reached by the family M of potential models of Σ, when all are driven by a common input sequence. The definition is through a recursive procedure.

Let then $M = \{(f_1, h_{10}, \sigma_{10}), \ldots, (f_q, h_{q0}, \sigma_{q0})\}$ be the family of potential models of the bounded interpreter Σ. Denote by $\sigma_0 := (\sigma_{10}, \ldots, \sigma_{q0})$ the *initial status vector* of the family M. For a point $\rho := (\rho_1, \ldots, \rho_q, u) \in (S_A \times X)^q \times V(D)$, let $(f_1, \ldots, f_q)\rho$ be the point $(f_1(\rho_1, u), \ldots, f_q(\rho_q, u)) \in (S_A \times X)^q$, i.e. the result of applying the recursion function vector to the point ρ.

Now, let $\omega \in (S_A \times X)^q$ be a fixed point. We construct recursively a sequence of subsets $R_0(M, \omega), R_1(M, \omega), \ldots$ of the space $(S_A \times X)^q \times V(D)$ as follows.

(i) $R_0(M, \omega) := \omega \times V(D)$, i.e. the set of all vectors of the form $(\omega, u), u \in V(D)$.

(ii) Assume that $R_j(M, \omega)$ has been constructed for some integer $j \geqslant 0$. The set $R_{j+1}(M, \omega)$ is then given by

$$R_{j+1}(M, \omega) := \{\bigcup_{\rho \in R_j(M, \omega)} (f_1, \ldots, f_q)\rho\} \times V(D) \qquad (3.2.1)$$

(iii) The *jointly reachable set* $R(M, \omega)$ of the family M at the point ω is defined by

$$R(M, \omega) := \bigcup_{j \geq 0} R_j(M, \omega) \tag{3.2.2}$$

As the name indicates, the jointly reachable set $R(M, \omega)$ consists of all points in the set $(S_A \times X)^q \times V(D)$ that can be reached by applying common input sequences to all potential models of Σ, starting from the status vector ω. When ω is the initial status vector σ_0 of the family M, we simply write $R(M) := R(M, \sigma)$, and refer to $R(M)$ as the *jointly reachable set* of the family M. It consists of all points that can be reached from the initial status by applying identical input sequences to all potential models of Σ.

Given a subfamily $\phi = \{(f_{i(1)}, h_{i(1)0}, \sigma_{i(1)0}), \ldots, (f_{i(r)}, h_{i(r)0}, \sigma_{i(r)0})\} \subset M$ of potential models of Σ, we define the standard projection

$$\Pi_\phi : (S_A \times X)^q \times V(D) \to (S_A \times X)^r : (\rho_1, \rho_2, \ldots, \rho_q, u) \mapsto (\rho_{i(1)}, \rho_{i(2)}, \ldots, \rho_{i(r)})$$

where $r = \#\phi$.

Once the jointly reachable set $R(M, \omega)$ has been computed for the family M at a point ω, we can directly obtain from it the jointly reachable set $R(\phi, \omega)$ for any subfamily $\phi \subset M$ at the point ω through the projection

$$R(\phi, \omega) = [\Pi_\phi R(M, \omega)] \times V(D)$$

We set $R(\phi) := [\Pi_\phi R(M)] \times V(D)$, i.e. the jointly reachable set of the subfamily ϕ from the initial status.

We discuss now some elementary aspects of the control problem for a bounded interpreter Σ having the family M of potential models. Clearly, the simplest control law would be to apply a single pre-determined input sequence, without taking any real-time measurements of the response. A controller that operates on this principle is called an *open loop controller*.

Open loop controllers are rather limited in their applicability, since, when the family M of potential models of Σ is relatively large, it is unlikely that the same input sequence would suit all potential models, and drive each one of them to its target tail set. Still, the open loop control scheme is not to be ignored, as it has no rivals for simplicity. Considering that the jointly reachable set $R(M)$ characterizes the set of all points that can be reached by applying a common input sequence to all members of M, we obtain the following necessary and sufficient condition for open loop control.

Proposition 3.2.1: *Let $\Sigma : D \to S(S_A)$ be a bounded interpreter having the family M of potential models, let $T(M)$ be the point target set of M, and let $R(M)$ be the jointly reachable set. Then, Σ can be steered to its target by an open loop controller if and only if $R(M) \cap T(M) \neq \varnothing$.*

Proof: Recall that $M = \{(f_1, h_{1o}, \sigma_{10}), \ldots, (f_q, h_{qo}, \sigma_{q0})\}$, and let $\Sigma_i : D \to S(S_A)$ be the interpreter induced by the pair (f_i, h_{io}) with the initial condition σ_{i0}. Recall that T_i is the target tail set of Σ_i, $i = 1, \ldots, q$. Now, open loop control means that there is a sequence $u \in D$ for which $T(\Sigma_i u) \cap T_i \neq \varnothing$ for all $i = 1, \ldots, q$. By definition of the sets $R(M)$ and $T(M)$, the latter is equivalent to the existence of a point $r \in R(M)$ that belongs to the target set $T(M)$. □

Whether or not an interpreter Σ with the family M of potential models is amenable to open loop control depends, to some extent, on the size of the family M. When M consists of just one single model, i.e. when Σ is accurately known, open loop control is possible in every case in which Σ can be steered to its target tail set. In other words,

when Σ is known, open loop control is possible whenever control is at all possible. However, as the number of models within the family M increases, it becomes less likely that an open loop controller will be able to achieve the control objective, since, in general, different models require different input sequences to reach their targets.

In preparation for our discussion of the general control problem, the next two statements indicate that the jointly reachable set is computable.

Lemma 3.2.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of q potential models. Then, for every point $\omega \in (S_A \times X)^q$, there is an integer $k \geqslant 1$ such that $R_k(M,\omega) \subset \cup_{i=0,\ldots,k-1} R_i(M,\omega)$.*

Proof: By contradiction, assume there is no integer $k \geqslant 1$ for which

$$R_k(M,\omega) \subset \cup_{i=0,\ldots,k-1} R_i(M,\omega).$$

Then, the jointly reachable set $R(M,\omega) = \bigcup_{j\geqslant 0} R_j(M,\omega)$ must contain an infinite number of distinct points. However, since Σ is a bounded interpreter, the number of points in $R(M,\omega)$ cannot exceed $(\#S_A)^q(\#X)^q(\#V(D))$, a contradiction. Hence, the Lemma is valid. □

Based on Lemma 3.2.1, the jointly reachable set can be computed as follows.

Proposition 3.2.2: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of q potential models. Let $k \geqslant 1$ be an integer for which*

$$R_k(M,\omega) \subset \bigcup_{i=0,\ldots,k-1} R_i(M,\omega).$$

Then, $R(M,\omega) = \bigcup_{i=0,\ldots,k-1} R_i(M,w)$.

Proof: Let k be as described in Proposition 3.2.2. We show by induction that $R_j(M,\omega) \subset \cup_{i=0,\ldots,k-1} R_i(M,\omega)$ for all $j \geqslant k$. The case $j = k$ follows by the definition of k. In preparation for an induction, assume that $R_n(M,\omega) \subset \cup_{i=0,\ldots,k-1} R_i(M,\omega)$ for some $n \geqslant k$, and consider the case $j = n+1$. Then, by definition of $R_{n+1}(M,\omega)$, we have

$$\begin{aligned} R_{n+1}(M,\omega) &= \{\textstyle\bigcup_{\rho\in R_n(M,\omega)} (f_1,\ldots,f_1)\rho\} \times V(D) \\ &\subset \{\textstyle\bigcup_{\rho\in \cup_{i=0,\ldots,k-1} R_i(M,\omega)} (f_1,\ldots,f_q)\rho\} \times V(D) \\ &= \textstyle\bigcup_{i=1,\ldots,k} R_i(M,\omega) \\ &\subset \textstyle\bigcup_{i=0,\ldots,k-1} R_i(M,\omega) \end{aligned}$$

where the step before last follows from the definition of the sets $R_i(M,\omega)$, $i = 1,\ldots,k$, and the last step is implied by the relation $R_k(M,\omega) \subset \bigcup_{i=0,\ldots,k-1} R_i(M,\omega)$. This concludes our proof. □

The combination of Lemma 3.2.1 and Proposition 3.2.2 yields the Corollary 3.2.1.

Corollary 3.2.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of q potential models. Then, the jointly reachable set $R(M,\omega)$ can be computed in a finite number of steps, for any point $\omega \in (S_A \times X)^q$.*

We induce now a partial order on the jointly reachable set $R(M)$. For two points $r_1, r_2 \in R(M)$, we say that r_1 is a *predecessor* of r_2 (written $r_1 < r_2$) if $r_2 \in R(M, r_1)$. In other words, if there is an input list that jointly leads the entire family M of potential models from the point r_1 to the point r_2. Equivalently, r_2 is a *successor* of r_1 when $r_1 < r_2$. Note that a point may simultaneously be a successor and a predecessor of

another point, as is the case, for instance, along a periodic sequence. Accordingly, the jointly reachable set is not well ordered by this relation.

Let $\phi \subset M$ be a subfamily of potential models. The set of all predecessors of a point $r \in R(\phi)$ is denoted by $C(\phi, r)$, and is called the *jointly controllable set* to r of the family ϕ. The jointly controllable set $C(\phi, r)$ can be computed through the solution of a system of linear algebraic equations, as described by Eilenberg (1974, Chapter 7, §6).

For a subset $S \subset (S_A \times X)^{\#\phi} \times V(D)$, we use the notation

$$R(\phi, S) := \bigcup_{\rho \in S \cap R(\phi)} R(\phi, \rho)$$
$$C(\phi, S) := \bigcup_{\rho \in S \cap R(\phi)} C(\phi, \rho)$$

where $C(\phi, S) := \varnothing$ and $R(\phi, S) := \varnothing$ whenever $S \cap R(\phi) = \varnothing$. Note that, by definition, $C(\phi, S) \subset R(\phi)$. In the case $S = T(\phi)$, the point target set of the family ϕ, we call $C(\phi, T(\phi))$ the *jointly controllable set* of the family ϕ, and denote it by $C(\phi)$.

Some insight into the significance of the jointly controllable set is provided by the following statement, which is equivalent to Proposition 3.2.1.

Proposition 3.2.3: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of potential models with the initial status vector σ_0. Then, Σ can be steered to its target tail set by an open loop controller if and only if $\sigma_0 \in \Pi_M C(M)$, where $C(M)$ is the jointly controllable set of the family M.*

3.3. *Feedback control*

Apart from the limited number of instances where open loop control is possible, a feedback controller is required to steer an interpreter with multiple potential models to its target. The general form of a feedback controller is depicted in (1.3). Clearly, the advantage of a feedback controller originates from the real-time data it collects about the interpreter response through the monitoring function h_m. The controller can use these data in an attempt to identify the active model of the interpreter, and to adjust its own characteristics accordingly. In this way, feedback control can handle a broader family of potential models, thus reducing the amount of *a priori* data that needs to be collected.

Let $\Sigma: D \to S(S_A)$ be a bounded interpreter, having the family M of potential models and the target tail set T. Our next objective is to derive necessary and sufficient conditions for the existence of monitoring function-controller combinations that steer Σ to its target tail set. We start with some terminology related to the partition of sets into disjoint subsets.

Let $\phi \subset M$ be a subset of potential models of Σ. As usual, a *partition* p of ϕ is a family $p = \{c_1, \ldots, c_k\}$ of disjoint subclasses of ϕ whose union is ϕ; i.e. $c_i \cap c_j = \varnothing$ for all $i \neq j \in \{1, \ldots, k\}$; and $\bigcup_{i=1,\ldots,k} c_i = \phi$.

A partition P of a partition $p = \{c_1, \ldots, c_k\}$ of ϕ is a set of partitions $P = \{p_1, \ldots, p_k\}$, where p_i is a partition of the class c_i, $i = 1, \ldots, k$. The combined partition Pp consists of the classes $Pp = \{\{p_1\}, \{p_2\}, \ldots, \{p_k\}\}$, which create a partition of ϕ into disjoint sets that are 'smaller' than those induced by the original partition p.

Given two partitions p and q of the same set ϕ, we say that the partition q is *finer* than the partition p (written $p \leqslant q$) if there is a partition P of the partition p such that $q = Pp$. Equivalently, when $p \leqslant q$, we say that p is *coarser* than q.

Let A, B be two partitions of the same set ϕ. The *meet* $A \wedge B$ of A and B is the coarsest partition of ϕ that is finer than A and finer than B.

A *partition chain* $\mathscr{P}(\phi)$ of ϕ is simply an ordered list of partitions $p_0 \leqslant p_1 \ldots \leqslant p_m$ of ϕ, with $p_0 := \phi$ being the identity partition.

Let $\mathscr{P}(\phi) = \{p_0 \leqslant p_1 \leqslant \cdots \leqslant p_m\}$ be a partition chain, and let $c \in p_i$ be a member of the partition p_i. We denote by $p_{i+1}(c)$ the partition of the class c induced by the partition p_{i+1}; i.e. letting $p_{i+1} = \{c_{i+1,1}, \ldots, c_{i+1,k}\}$, the partition $p_{i+1}(c)$ consists of all non-empty intersections $c \cap c_{i+1,j}, j = 1, \ldots, k$. A *path* of partition chain $\phi \leqslant p_1 \leqslant \cdots \leqslant p_m$ of ϕ is an ordered list $\{\phi, c_1, c_2, \ldots, c_m\}$ of subsets of ϕ, where $c_i \in p_i(c_{i-1}), i = 1, \ldots, m$, and $c_0 := \varnothing$.

Consider next the family $M = \{(f_1, h_{1o}, \sigma_{1o}), \ldots, (f_q, h_{qo}, \sigma_{qo})\}$ of potential models of Σ. Let $c = \{(f_{i(1)}, h_{i(1)o}, \sigma_{i(1)o}), \ldots, (f_{i(r)}, h_{i(r)o}, \sigma_{i(r)o})\}$, $1 \leqslant r \leqslant q$, be a subset of M. For a point $\rho = (\rho_1, \ldots, \rho_q, u) \in R(M)$, we denote

$$\Gamma(c)\rho := \bigcup_{j=i,\ldots,r} \{(\Pi_s \rho_{i(j)}] \dot{\bigcup} u\} \tag{3.3.1}$$

i.e. the set of all medium values corresponding to the members of c at the point ρ.

Now, let $\phi \subset M$ be a subfamily of potential models of Σ, let $P = \{c_1, \ldots, c_m\}$ be a partition of ϕ, and let $h: S_A \to \Delta$ be a function. Then, we say that h is *compatible* with the partition P of ϕ at the point $\rho \in R(M)$ if the following holds for all $i, j = 1, \ldots, m$

$$h[\Gamma(c_i)\rho] \cap h[\Gamma(c_j)\rho] = \varnothing \quad \text{whenever} \quad i \neq j \tag{3.3.2}$$

i.e. the function h assumes a distinct set of values over each one of the classes $c_1, \ldots, c_m$.

The following statement provides a necessary and sufficient condition for the existence of a controller that steers the bounded interpreter Σ to its target tail set. It is one of the main results of the present paper, and we shall discuss its intuitive meaning immediately.

Theorem 3.3.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of potential models, and let T be the target tail set of Σ. Denote by $T(c)$ the point target set of a subfamily $c \subset M$. Then,* (i) *and* (ii) *are equivalent.*

(i) *There is a strictly causal autonomous controller C and a monitoring function $h_m: S_A \to \Delta$ that steer Σ to its target tail set.*

(ii) *There is a partition chain $\mathscr{P}(M) = \{M \leqslant p_1 \leqslant \cdots \leqslant p_m\}$ of M, every path $\{c_0, c_1, \ldots, c_m\}$ of which satisfies the following.*

(ii*a*) *There are points $\rho_1 < \cdots < \rho_m$ of $R(M)$ and a function $h: S_A \to \Delta$ such that h is compatible with the partition $p_i(c_{i-1})$ at the point ρ_i, $i = 1, \ldots, m$, and*

(ii*b*) *$T(c_m) \cap \Pi(c_m) R(M, \rho_m) \neq \varnothing$ when $m \geqslant 1$, or*

(ii*c*) *$T(M) \cap \Pi(M) R(M) \neq \varnothing$ when $m = 0$.*

In qualitative terms, Theorem 3.3.1 indicates that the controller acts through a hierarchical identification scheme, using the function $h: S_A \to \Delta$ as the monitoring function. At each step, the controller attempts to narrow the set of possible models of Σ by checking the values submitted by the monitoring function through the feedback channel. The controller starts by providing Σ with an input list that takes the family M from the initial condition to the point $\rho_1 \in R(M)$. When the point ρ_1 is reached, condition (ii*a*) means that the controller can identify the class $c_1 \in p_1(M)$ to which the active model must belong. The controller then supplies a continuation of the input list that leads from ρ_1 to ρ_2. At this point, condition (ii*a*) shows that the controller can

identify the class $c_2 \in p_2(c_1)$ to which the active model must belong. And so on, up to the point ρ_m at which the class $c_m \subset M$, to which the active model must belong, is identified. In view of (ii*b*), more detailed identification of the active model is not necessary, since all members of c_m can be kept within their target tail sets by a common input sequence.

An algorithm for the selection of a monitoring function is given in §4.2 below, where we characterize the class of all possible monitoring functions. Meanwhile, we turn to the proof of Theorem 3.3.1, which depends on the following rather technical statement.

Lemma 3.3.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family $M = \{(f_1, h_{1o}, \sigma_{10}), \dots, (f_q, h_{qo}, \sigma_{q0})\}$ of potential models, and let T be the target tail set of Σ. Assume there are a monitoring function $h_m: S_A \to \Delta$ and a strictly causal controller C that steer Σ to its target. Let $u(i)$ be the input sequence of Σ generated by the controller C when the active model is $(f_i, h_{io}, \sigma_{i0})$. For each integer $k \geqslant 0$, let P_k be the partition of the family M induced by the relation $u(i)]_0^k = u(i')]_0^k$, $i, i' \in \{1, \dots, q\}$. Then, the following hold.*

(i) *The partitions $\{P_k\}$ satisfy $M = P_0 \leqslant P_1 \leqslant P_2 \leqslant \cdots$.*

(ii) *There is an integer $\alpha \geqslant 0$ beyond which $P_k = P_\alpha$ for all $k \geqslant \alpha$.*

(iii) *There is a chain of partitions $\mathscr{P} = \{M \leqslant p_1 \leqslant \cdots \leqslant p_r\}$ of M and an ordered list of points $\rho_1 < \cdots < \rho_r$ of $R(M)$ such that:*

(iii*a*) $p_r \geqslant P_\alpha$.

(iii*b*) *For every path $\{c_0, c_1, \dots, c_r\}$ of $\mathscr{P}$, the function h_m is compatible with the partition $p_i(c_{i-1})$ at the point ρ_i, $i = 1, \dots, r$.*

(iii*c*) $r \leqslant \max\{\alpha - 1, 0\}$.

(iii*d*) *For every class $c = \{(f_{i(1)}, h_{i(1)o}, \sigma_{i(1)0}), \dots, (f_{i(d)}, h_{i(d)o}, \sigma_{i(d)0})\} \in p_r$, the corresponding input lists satisfy $u(i(1))]_{\alpha+1}^\infty = u(i(2))]_{\alpha+1}^\infty = \cdots = u(id))]_{\alpha+1}^\infty$.*

Proof of Lemma 3.3.1: Part (i) of the Lemma is a direct consequence of the definition of the sequence $\{P_k\}$, whereas (ii) follows from the fact that the family M is finite. Furthermore, since $P_k = P_\alpha$ for all $k \geqslant \alpha$, and since $p_r \geqslant P_\alpha$, (iii*d*) holds.

To prove the remaining parts of the Lemma, let $\mu(i)$ be the monitored sequence generated by the monitoring function h_m when the model $(f_i, h_{io}, \sigma_{i0})$ is active and receiving the input sequence $u(i)$. then, $\mu(i)$ serves as the input sequence of the strictly causal controller C. Denote $\tau := \max\{\alpha - 1, 0\}$. For each integer $k \in \{0, \dots, \tau\}$, let π_k be the partition of M induced by the equivalence relation $\mu_k(i) = \mu_k(i')$, $i, i' \in \{0, \dots, q\}$. Applying the meet operation on the list $\{\pi_k\}_{k=0}^\tau$, we create the partitions

$$\Pi_j := \wedge_{k=0,\dots,j} \pi_k, \quad j = 0, \dots, \tau$$

A slight reflection shows that Π_j is the partition of M created by the equivalence relation $\mu(i)]_0^j = \mu(i')]_0^j$, $i, i' \in \{1, \dots, q\}$, and $M \leqslant \Pi_0 \leqslant \cdots \leqslant \Pi_\tau$. But then, since the controller C is strictly causal and generates the sequence $u(i)$ in response to the sequence $\mu(i)$, we must have

$$\Pi_{j-1} \geqslant P_j \tag{3.3.3}$$

for all $j \geqslant 1$. We distinguish now between two cases.

When $\alpha = 0$, all sequences $u(1), \dots, u(q)$ generated by the controller C are equal (i.e. it is an open loop controller), and $P_\alpha = M$. We then set $\mathscr{P} := \{M\}$, and (iii*a*), (iii*c*) are valid.

When $\alpha \geqslant 1$, let $(s_k(i), x_k(i))$ be the status of the model $(f_i, h_{i0}, \sigma_{i0})$ at the step $k \geqslant 0$, when driven by its input sequence $u(i)$. Using (3.3.2), we conclude that for every class $c_k = \{(f_{i(1)}, h_{i(1)0}, \sigma_{i(1)0}), \dots, (f_{i(a)}, h_{i(a)0}, \sigma_{i(a)0})\} \in \Pi_k$, the corresponding input sequences satisfy $u(i(1))]_0^k = u(i(2))]_0^k = \cdots = u(i(a))]_0^k$. Denote

$$u_k(c_k) := u_k(i(1)) = u_k(i(2)) = \cdots = u(i(a)),$$

and set

$$\chi(c_k) := ((s_k(1), x_k(1)), (s_k(2), x_k(2)), \dots, (s_k(q), x_k(q)), u_k(c_k))$$

$k = 0, \dots, \tau$. Then, since $\mu_k(i) = h_m(s_k(i) \dot{\bigcup} u_k(i))$, it follows by the construction of the partitions $\{\pi_k\}_{k=0}^{\tau}$ and $\{\Pi_k\}_{k=0}^{\tau}$ that h_m is compatible with any path $\{c_0, \dots, c_\tau\}$ of $\{M \leqslant \Pi_1 \leqslant \cdots \leqslant \Pi_\tau\}$ at the points $\chi(c_0) < \chi(c_1) < \chi(c_2) < \cdots < \chi(c_\tau) \in R(M)$.

Finally, set $\mathscr{P} := \{M \leqslant \Pi_0 \leqslant \cdots \leqslant \Pi_\tau\}$. Then, (iii*a*) and (iii*b*) hold, with $\rho_i := \chi(c_i)$, $i = 0, \dots, \tau$. □

Proof of Theorem 3.3.1: Assume first that there are a monitoring function $h_m : S_A \to \Delta$ and a strictly causal controller C that steer Σ to its target tail set T. Lemma 3.3.1 then applies, and part (ii) of the Theorem follows directly from the Lemma.

The converse direction of the proof is given by the Controller Design Algorithm 3.3.1 below. □

We now provide an algorithm for the design of controllers that steer an interpreter having multiple potential models to its target tail set. A qualitative explanation of the way the algorithm operates was provided earlier, subsequent to the statement of Theorem 3.3.1.

Algorithm 3.3.1 — The controller design algorithm:

Assume that Part (ii) of Theorem 3.3.1 is valid, and use the notation of the theorem. A strictly causal controller C that steers Σ to its target tail set is then obtained as follows.

The case $m = 0$

Use open loop control (see Proposition 3.2.1). Assign to the controller C a single output sequence that steers the family M from its initial status to a point of $T(M)$. Then, continue the controller sequence to infinity with an appropriate sequence given by the input sequence of a periodic element of $(\Theta_1, \dots, \Theta_q)_u$.

The case $m \geqslant 1$

Use the function $h : S_A \to \Delta$ as the monitoring function (the derivation of monitoring functions is discussed in Corollary 3.3.1 and in §4.2 below). The controller is then designed through the following iterative procedure.

Step 0. At the initial step, it is (only) known that the active model of Σ belongs to the family M. Set $k_0 := 0$.

Step 1. Assume the controller C has reached a step $k_i \geqslant 0$, where $i \geqslant 0$ is an integer, and that the following conditions hold.

(1*a*) Along the way, the controller has identified a descending list

$$M = c_0 \supset c_1 \supset c_2 \supset \cdots \supset c_{i-1}$$

of subsets of M to which the active model belongs, where $c_j \in p_j(c_{j-1})$, $j = 1, \dots, i-1$.

(1 *b*) Let $u_0^{k_i}$ be an input list that steers the entire family M of potential models of Σ to the point ρ_i, while passing through the points $\sigma_0, \rho_1, \ldots, \rho_{i-1}$ along the way. Assign this input list as the output list of the controller C up to the step k_i. (Such an input list exists since $\sigma_0 < \rho_1 < \cdots < \rho_m$).

Step 2. Denote

$$\rho_i = ((s_i(1), x_i(1)), (s_i(2), x_i(2)), \ldots, (s_i(q), x_i(q)), u_{k_i})$$

and let $\{(f_{j(1)}, h_{j(1)0}, \sigma_{j(1)0}), \ldots, (f_{j(d)}, h_{j(d)0}, \sigma_{j(d)0})\}$ be the class c_{i-1}. Set the controller C to determine the class $c_i \in p_i(c_{i-1})$ to which the active model belongs, by checking the monitored values

$$h[s_{k_i}(j(1)) \dot{\bigcup} u_{k_i}], \ldots, h[s_{k_i}(j(d)) \dot{\bigcup} u_{k_i}]$$

corresponding to the members of c_{i-1}. This is possible, since the monitoring function h is compatible with the partition $p_i(c_{i-1})$ at the point ρ_i (Theorem 3.3.1) (ii *a*)), and since by (1 *a*) the active model is known to belong to the class c_{i-1}.

Step 3. Iterate Steps 1 and 2 for $i = 1, \ldots, m$.

Step 4. After the iteration for $i = m$, we obtain a class $c_m \subset M$ that satisfies condition (ii *b*) of Theorem 3.3.1, and to which the active model must belong. By Corollary 3.1.1, there is then a periodic input sequence $u_{k_m}^{\infty}$ in the joint target tail of the class c_m that steers all members of c_m to their respective target tail sets. Assign then $u_{k_m}^{\infty}$ as the tail of the sequence generated by the controller C. This completes the controller design.

We comment that the controller constructed in the algorithm is strictly causal, since at each step k_i, $i = 1, \ldots, m$, the output values of the controller for the steps $k_i + 1, k_i + 2, \ldots, k_{i+1}$ are determined from the monitored values for the steps $0, \ldots, k_i$.

It is interesting to note that the controller C, which steers Σ to its target tail set, performs control and identification processes that are interrelated. The controller generates an initial input sequence that steers Σ to the point ρ_1. At this point, the controller can identify a class c_1 that contains the active model. Then, C steers the interpreter to the point ρ_2, to make possible the identification of the class c_2. This process of successive control and identification steps is continued up to the point where the active model is identified to within the class c_m. More accurate identification of the active model is then not necessary, since all members of c_m can be driven to their respective target tail sets by the same input list. An algorithm that yields the points $\rho_1, \ldots, \rho_m \in R(M)$ is described in §4 below.

We conclude this section with a preliminary discussion of the selection of the function h of Theorem 3.3.1 (ii), which, as we have seen in Algorithm 3.3.1, serves as the monitoring function. Elementary algebraic considerations show that compatibility with any function $h: S_A \to \Delta$ implies compatibility with the identity function $S_A \to S_A$. Thus, if there is a function h that satisfies Part (ii) of Theorem 3.3.1, then the identity function $S_A \to S_A$ will satisfy the same conditions, with the same partition chain $\mathscr{P}(M)$ and the same points $\rho_1, \ldots, \rho_m$.

The identity function $S_A \to S_A$ provides the most detailed feedback information possible, as it detects and identifies at each step every element of the medium. This is, of course, done at the expense of the highest measurement burden. Nevertheless, the identity function provides a concrete example of a monitoring function. We emphasize

that, in many cases, it is possible to use a monitoring function that requires fewer measurements than required by the identity function. The set of all possible monitoring functions is derived in §4 below. We summarize the discussion of the last two paragraphs in the following corollary.

Corollary 3.3.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter, having the family M of potential models, and the target tail set T. Then, there is a function $h: S_A \to \Delta$ that satisfies Part* (ii) *of Theorem* 3.3.1 *if and only if the identity function $S_A \to S_A$ satisfies the same.*

Finally, we comment that an analogue of Theorem 3.3.1 can be stated in terms of the jointly controllable set $C(M)$, rather than in terms of the jointly reachable set $R(M)$. This would be similar to the connection between Propositions 3.2.1 and 3.2.3, and is omitted here.

4. Computation and the monitoring function

The present section concentrates on the development of computational algorithms. These algorithms determine whether or not a control objective can be achieved; and when the control objective is achievable, they help characterize all possible monitoring functions. Appropriate controllers are then obtained via the controller design algorithm 3.3.1. We start with some basics.

4.1. *Common divisors and the search algorithm*

Consider a sequence $u \in S(S_A)$ and a (finite or infinite) non-empty list

$$\lambda = (\lambda_0, \ldots, \lambda_n) \in (S_A)^{n+1}.$$

We say that λ is a *left divisor* of u of $u_0^n = \lambda$. The *length* $|\lambda|$ of the divisor λ is simply the number of elements in its list, i.e. $|\lambda| = n+1$ in the present case.

Given two sequences $u, v \in S(S_A)$, we say that λ is a *common left divisor* of u and v if $u_0^n = v_0^n = \lambda$. More generally, λ is a common left divisor of a set $S \subset S(S_A)$ whenever λ is a left divisor of every element of S.

A list λ is a *longest common left divisor* of two sequences $u, v \in S(S_A)$ if λ is a common left divisor of u and v, and if the length of λ is not less than the length of any common left divisor of u and v. A longest common left divisor for a subset $S \subset S(S_A)$ is defined analogously.

It is easy to see that any two sequences $u, v \in S(S_A)$ that have a common left divisor, have a unique longest common left divisor. The longest common left divisor of u and v is of infinite length if and only if $u = v$.

Using common left divisors, we induce an equivalence relation L on the set $S(S_A)$ by writing uLv whenever the two sequences $u, v \in S(S_A)$ have a common left divisor.

Next, let $S(1), S(2), \ldots, S(q) \subset S(S_A)$ be a family of non-empty subsets of sequences. A *comb* of the family $\{S(i)\}_{i=1}^q$ is any set $\chi \subset S(S_A)$ of sequences that contains exactly one sequence from each one of the sets $S(1), \ldots, S(q)$.

Consider now a bounded interpreter $\Sigma: D \to S(S_A)$ having the family

$$M = \{(f_1, h_{1o}, \sigma_{10}), \ldots, (f_q, h_{qo}, \sigma_{q0})\}$$

of potential models, and let T_i be the target tail set associated with the model $(f_i, h_{io}, \sigma_{i0})$, $i = 1, \ldots, q$. As before, $\Sigma_i: D \to S(S_A)$ is the interpreter induced by

the model $(f_i, h_{io}, \sigma_{i0})$, and $T(\Sigma_i u)$ is the complete tail set of the output sequence $\Sigma_i u$. Let $I(\Sigma_i) \subset D$ be the set of all ultimately periodic input sequences $u \in D$ for which $T(\Sigma_i u) \cap T_i \neq \varnothing$. In other words, $I(\Sigma_i)$ consists of all ultimately periodic input sequences that steer Σ_i to its target tail set. We call $I(\Sigma_i)$ the set of *successful input sequences* of the potential model Σ_i of Σ.

Clearly, if any of the potential models $\Sigma_1, \ldots, \Sigma_q$ of Σ has an empty set $I(\Sigma_i)$ of successful input sequences, then it is impossible to steer Σ to its target tail set. Therefore, we shall assume throughout that all potential models have non-empty successful input sets.

The computation of the set $I(\Sigma_i)$ can be performed in status space, using the notions introduced in §3.1. First, find the set of all input lists that steer the recursion function f_i to its own target point set $T(\{f_i, h_{io}, \sigma_{i0}\})$. This can be accomplished through the solution of a system of linear algebraic equations, as described by Eilenberg (1974, Chapter 7, §6). Then, augment these input lists with their periodic counterparts in the target tail Θ_i of the model $(f_i, h_{io}, \sigma_{i0})$.

A preliminary indication of the relevance of the notion of a comb is provided by the following statement, which follows directly from the definitions.

Proposition 4.1.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of q potential models and the target tail set T. Assume there are a monitoring function h_m and a strictly causal autonomous controller C that steer Σ to its target tail set. Let $u(i) \in D$ be the sequence generated by the controller C when model number i of M is active, and let $I(\Sigma_i)$ be the set of all successful input sequences of potential model number i. Then, the list $\{u(1), \ldots, u(q)\}$ forms a comb of the class $\{I(\Sigma_i)\}_{i=1}^{q}$.*

We can now state the following criterion that determines whether or not an interpreter Σ can be steered to its target tail set by an open loop controller.

Proposition 4.1.2: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family $M = \{\Sigma_1, \ldots, \Sigma_q\}$ of potential models, and the target tail set T. Let $I(\Sigma_i\}$ be the set of successful input sequences of potential model number i. Then, the following are equivalent.*

(i) *Σ can be steered to its target tail set by an open loop controller.*

(ii) *The class $\{I(\Sigma_i)\}_{i=1}^{q}$ has a comb χ whose elements have a common left divisor of infinite length.*

Proof: Let $T(M)$ be the point target set of the family M, and let $R(M)$ be its jointly reachable set. A slight reflection shows that (ii) is valid if and only if $T(M) \cap R(M) \neq \varnothing$. But then, the present proposition follows by Proposition 3.2.1. □

The criterion for open loop control described in Proposition 4.1.2 simply involves a search for a sequence that is common to all the sets $I(\Sigma_1), \ldots, I(\Sigma_q)$; i.e. for a member of the intersection $I(\Sigma_1) \cap \cdots \cap I(\Sigma_q)$. Since all involved sequences become periodic after a finite number of steps, it is seen that this criterion involves a finite search, and can be implemented on a digital computer. Of course, for cases where the criterion holds, the control objective can be achieved without the use of a monitoring function.

We turn now to a discussion of the general case of feedback control. We provide a computational algorithm that determines whether or not the control objective can be achieved by feedback control. In the notation introduced in this subsection, the algorithm qualitatively functions as follows.

First, select a comb χ of the class $\{I(\Sigma_i)\}_{i=1}^{q}$. Compare the initial segments of the sequences of χ element by element to determine whether they have a common left divisor. If there is no common left divisor, choose another comb; if none of the combs of $\{I(\Sigma_i)\}_{i=1}^{q}$ has a common left divisor, then control of Σ is impossible. This is due to the strict causality of the controller C, which implies that all sequences generated by C must have the same initial value.

If the sequences of χ have a common left divisor, let $\alpha_{1,1}$ be their longest common left divisor. Assign $\alpha_{1,1}$ as the output list of the controller C for the steps $0, 1, \ldots, |\alpha_{1,1}|-1$. This list is appropriate for all models of the family M, irrespective of which one is active. If $|\alpha_{1,1}| = \infty$, we obtain in this way an open loop controller C, completing the process. Otherwise, if $|\alpha_{1,1}| \neq \infty$, proceed as follows.

Delete the initial segment $\alpha_{1,1}$ from all sequences of χ, and denote by χ_1 the resulting family of sequences. Using the equivalence relation L on χ_1, induce a partition P_1 of the family M of potential models of Σ, by grouping into each class of P_1 all models whose corresponding sequences in χ_1 have a common left divisor. Then, all models that belong to the same class of P_1 require the same input value at the step $|\alpha_{1,1}|$; and models from different classes of P_1 require different input values at this step.

Thus, at the step $|\alpha_{1,1}|$, the input value that C must generate varies, depending on the class of P_1 to which the active model belongs. In other words, since C is strictly causal, it must be possible by the step $|\alpha_{1,1}|-1$ to determine to which class of P_1 the active model belongs.

This determination must be made based on the monitored values for the steps $0, \ldots, |\alpha_{1,1}|-1$; Clearly, the latter is possible if and only if such determination can be made based on the medium values for these steps, since the monitored values are determined by the medium values through the monitoring function.

Let $\mu_0(i), \mu_1(i), \ldots, \mu_{|\alpha_{1,1}|-1}(i)$ be the medium list for steps $0, \ldots, |\alpha_{1,1}|$, obtained when model number $i \in \{1, \ldots, q\}$ is driven by the input list $\alpha_{1,1}$ from its initial status σ_{i0}. Use the equivalence relation $\mu_0^{|\alpha_{1,1}|-1}(i) = \mu_0^{|\alpha_{1,1}|-1}(j)$, $i, j \in \{1, \ldots, q\}$, to induce a partition $P_{\mu,1}$ of the family M of potential models. Each class of $P_{\mu,1}$ consists of all potential models whose medium lists $\mu_0^{|\alpha_{1,1}|-1}(\cdot)$ are identical. A slight reflection shows then that the class of P_1 to which the active model belongs can be identified by a strictly causal controller if and only if $P_{\mu,1} \geqslant P_1$.

The complete algorithm is listed next. It iterates the process just described, one single step at a time.

Algorithm 4.1.1 — The search algorithm:

Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family

$$M = \{(f_1, h_{1o}, \sigma_{10}), \ldots, (f_q, h_{qo}, \sigma_{q0})\} = \{\Sigma_1, \ldots, \Sigma_q\}$$

of potential models, and let T be the target tail set of Σ. Recall that $R(M)$ is the jointly reachable set of the family M. Denote by $I(\Sigma_i)$ the set of successful input sequences of the model Σ_i, $i = 1, \ldots, q$, and let $\chi = \{u(1), \ldots, u(q)\}$ be a comb of the class $\{(\Sigma_i)\}_{i=1}^{q}$. The algorithm computes two partition chains of the family M.

Step 0. If the sequences of χ do not have a common left divisor, choose a different comb χ, and return to Step 0. If a comb with a common left divisor cannot be found, terminate the algorithm.

Step 1. Let $\alpha_{1,1}$ be the longest left divisor of the sequences of χ. Clearly, $\alpha_{1,1}$ is either of infinite length, or of finite length.

Step 2. If $|\alpha_{1,1}| = \infty$, set

$$\mathscr{P}(\chi) := [M]$$

i.e. the chain consisting only of the entire family M. Terminate the algorithm. An open loop controller is possible.

Step 3. If $|\alpha_{1,1}| < \infty$, set $P_0 := M$ and $P_{\mu,0} := M$. The initial condition vector $\sigma_0 =: \sigma_{0,1}$ of the family M and the comb $\chi =: \chi_0$ are associated with the partition P_0. Finally, set $\rho_{0,1} := (\sigma_0, \alpha_{1,1}]_0)$, where $\alpha_{1,1}]_0$ is the initial element of the list $\alpha_{1,1}$. Note that $\rho_{0,1} \in R(M)$, the jointly reachable set of the family M.

Step 4. Computation of the input partition.

In preparation for a recursion, assumed that a partition P_r of the family M is given for some integer $r \geqslant 0$. In this step we compute a partition P_{r+1} of M, based on properties of the sequences of the comb χ.

Initial conditions. Let $P_r = \{c_1, \ldots, c_t\}$ be the given partition, where $t := \#P_r$, and let $c_n = \{\Sigma_{i(1)}, \ldots, \Sigma_{i(n)}\}$, $n = 1, \ldots, t$. With each class $c_n \in P_r$ there is associated a vector $\sigma_{r,n} \in \Pi_M R(M)$ and a class of sequences $\chi_n = \{v(i(1)), \ldots, v(i(n))\}$. Here, $v(i(j))$ is a tail of that sequence of χ that belongs to $I(\Sigma_{i(j)})$, so that exactly one element of χ_n is associated with each model of c_n. Finally, with each class $c_n \in P_r$, there is associated a point $\rho_{r,n} \in R(M)$.

The Step. Using the equivalence relation L, create a partition of the class χ_n, and let $\xi_{n,1}, \ldots, \xi_{n,j(n)}$ be the disjoint sets of this partition. Since $\xi_{n,i} \subset \chi_n$, each element of $\xi_{n,i}$ is a tail of a successful input sequence of one of the members of c_n. Let $c_{n,i}$ be the subset of c_n that consists of all models whose input sequence tails belong to $\xi_{n,i}$, $i = 1, \ldots, j(n)$. Clearly, $\{c_{n,1}, \ldots, c_{n,j(n)}\}$ forms a partition of the class c_n.

Let P_{r+1} be the partition of M that consists of the classes

$$c_{n,i}, n = 1, \ldots, t, i = 1, \ldots, j(n).$$

We call P_{r+1} an *input partition*, since it is induced by the input sequences that belong to the comb χ.

Next, let $\alpha_{n,i}$ be the longest common left divisor of the sequences of $\xi_{n,i}$, and set

$$\beta(r+1) := \min\{|\alpha_{nm,i}|, n = 1, \ldots, t, i = 1, \ldots, j(n)\}$$

Let $\chi_{n,i}$ be the set of sequences obtained when the first element is deleted from each one of the sequences of $\xi_{n,i}, n = 1, \ldots, t$, $i = 1, \ldots, i(n)$.

Next, start the family M of models at the status vector $\sigma_{r,n}$, and apply the (common) input value $\alpha_{n,i}]_0$, the first element of $\alpha_{n,i}$, to all models. Let

$$\sigma_{r+1,n,i} \in \Pi_M R(M)$$

be the resulting status vector. Finally, each class $c_{n,i}$ of P_{r+1} is associated with the vector $\sigma_{r+1,n,i} \in \Pi_M R(M)$; with the class of sequences $\chi_{n,i}$; and with the vector $\rho_{n,i} := (\sigma_{r,n}, \alpha_{n,i}]_0) \in R(M)$.

Step 5. Computation of the medium partition.

For recursion purposes, assume that a partition $P_{\mu,r}$ of the family M of potential models is given. In this step we compute a partition $P_{\mu,r+1}$ of M.

As any vector of the jointly reachable set, the vector $\rho_{r,n} \in R(M)$ of Step 4 determines the medium values $\mu_1(r,n), \ldots, \mu_q(r,n)$ of all models in M, through the direct union of the corresponding internal and input values.

Let c_n be a class of the partition P_r, as in Step 4. Denote by $\pi(r,n)$ the partition induced on c_n by the equivalence relation $\mu_\varepsilon(r,n) = \mu_\tau(r,n)$, where ε and τ are integers representing models $\Sigma_\varepsilon, \Sigma_\tau \in c_n$. Let $\Pi(r,n) := \{\pi(r,n), \bar{c}_n\}$ be the partition of M created by $\pi(r,n)$ and the complement $\bar{c}_n$ of c_n in M. Define the partition

$$P_{\mu,r+1} := [\wedge_{n=1,\ldots,t} \Pi(r,n)] \wedge P_{\mu,r}$$

which, being created by the medium values, is called the *medium partition.*

Step 6. Verification. If $P_{\mu,r+1} \geqslant P_{r+1}$, continue; otherwise, terminate the algorithm, choose a different comb χ of $\{I(\Sigma_i)\}_{i=1}^q$, and return to Step 0.

Step 7. Termination. If $\beta(r) = \beta(r+1) = \infty$, terminate the algorithm; otherwise, return to Step 4. Termination at the present step indicates the existence of a monitoring function-controller combination that steers Σ to its target tail set.

When the algorithm terminates at Steps 2 or 7, it generates a chain of partitions

$$\mathscr{P}(\chi) := \{M \leqslant P_1 \leqslant P_2 \leqslant \cdots \leqslant P_s\} \tag{4.1.1}$$

of the family M of potential models of Σ. Furthermore, for every path

$$M \leqslant c_1 \leqslant c_2 \leqslant \cdots \leqslant c_s \text{ of } \mathscr{P}(\chi),$$

it generates a list of points

$$\omega_0 < \omega_1 < \cdots \leqslant \omega_s \tag{4.1.2}$$

that consists of the points $\rho_{r,n}$ of Step 4, that have been renamed and renumbered here for future use.

Note that, for a bounded interpreter, the search algorithm involves only a finite number of search and comparison steps, and is implementable on a digital computer. The search algorithm facilitates the computation of all critical quantities needed for the solution of the control problem for the interpreter Σ. First we show that it can be used to determine whether control is possible or not.

Theorem 4.1.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the class $M = \{\Sigma_1, \ldots, \Sigma_q\}$ of potential models and the target tail set T. Let $\{I(\Sigma_i)\}_{i=1}^q$ be the class of successful input sequences for the family M. Then, the following are equivalent.*

(i) *There are a monitoring function and a strictly causal controller that steer Σ to its target tail set.*

(ii) *The class $\{I(\Sigma_i)\}_{i=1}^q$ has a comb χ for which the Search Algorithm* 4.1.1 *terminates at Steps 2 or 7.*

Proof: Assume that the search algorithm (4.1.1) terminates at Steps 2 or 7, and let $\mathscr{P}(\chi)$ and $\omega_0 < \omega_1 < \cdots < \omega_s$ be given by (4.1.1) and (4.1.2). Then, in view of Step 6 of the algorithm and the construction of the partitions $P_{\mu,r}$ and P_r, it follows that part (ii) of Theorem 3.3.1 is valid for the partition $\mathscr{P}(M) := \mathscr{P}(\chi)$, with the list of points $\{\rho_1 < \cdots < \rho_m\} := \{\omega_0 < \omega_1 < \cdots < \omega_s\}$, and with the function h being taken as the

identity function $S_A \to S_A$. Part (i) of Theorem 4.1.1 follows directly from Theorem 3.3.1.

We turn now to the converse direction of the theorem. First, let χ be a comb for which the search algorithm terminates at Steps 0 or 6. Taking into account the definition of the partitions $P_{\mu,r}$ and P_r, this implies that the sequences of χ cannot be generated by a strictly causal controller C, while using the identity function $S_A \to S_A$ as the monitoring function.

Assume now that part (ii) of the theorem does not hold; i.e. there is no comb χ of the class $\{I(\Sigma_i)\}_{i=1}^{q}$ for which the search algorithm terminates at Steps 2 or 7. Then, for every such comb χ, the algorithm terminates at Steps 0 or 6. Combining the previous paragraph with Proposition 4.1.1, it follows then that there is no strictly causal controller C that steers the interpreter Σ to its target tail set, while using the identity function $S_A \to S_A$ as the monitoring function. In view of Corollary 3.3.1, this implies that part (i) of the theorem does not hold, and our proof concludes. □

As the first part of the proof indicates, the search algorithm, when terminating at Steps 2 or 7, provides all the data necessary for the construction of a controller that steers Σ to its target tail set, using the identity function $S_A \to S_A$ as the monitoring function. Clearly, one such set of data is obtained for every comb χ for which the search algorithm terminates at Steps 2 or 7.

To summarize, a solution to the problem of controlling the interpreter Σ is obtained by performing the search algorithm 4.1.1, and then using the data it generates to perform the controller design algorithm 3.3.1. So far however, we have concentrated mostly on the case where the monitoring function h_m is taken as the identity function $S_A \to S_A$, a choice that requires the largest number of measurements. Nevertheless, the search algorithm 4.1.1 provides the means to find all possible monitoring functions, as discussed in the next subsection. Once all possible monitoring functions are known, the simplest one can be selected, and an appropriate controller is then obtained through the controller design algorithm 3.3.1.

4.2. *Characterizing all possible monitoring functions*

We start by inducing a partial order on the set of partition chains. Let

$$P = \{M \leqslant P_1 \leqslant \cdots \leqslant P_m\}$$

and $P' = \{M \leqslant P'_1 \leqslant \cdots \leqslant P'_k\}$ be two partition chains of the same family M. The integer m is called the *length* of the partition P. We say that the chain P is *finer* than the chain P' (written $P \geqslant P'$) if $P_i \geqslant P'_i$ for all $i = 1, \ldots, \min\{m, k\}$, and if $P_m \geqslant P'_k$ when $m < k$. When P is finer than P', we also say that P' is *coarser* than P. The fact that the family M consists of a finite number of elements implies the following.

Proposition 4.2.1: *Let P be a partition chain of a finite family M, and let $r \geqslant 0$ be an integer. There is then a finite number of partition chains of M that are coarser (or finer) than P, among all partition chains of length not exceeding r.*

Let $\Sigma: D \to S(S_A)$ be a bounded interpreter having the family M of q potential models and the target tail set T. Let $\rho_0 < \rho_1 < \cdots < \rho_m \in R(M)$, be a list of points, where $R(M)$ is the jointly reachable set of the family M. In terms of individual coordinates, write

$$\rho_i := ((s_i(1), x_i(1)), \ldots, (s_i(q), x_i(q)), u_i)$$

$i = 0, \ldots, m$. The pair $(s_i(j), x_i(j))$ is then the status of model number j, and all models share the common input value u_i.

Now, let $h: S_A \to \Delta$ be a function, and, for each $i \in \{0, \ldots, m\}$, let P_i be the partition of M induced by the equivalence relation $h(s_i(j) \dot{\bigcup} u_i) = h(s_i(j') \dot{\bigcup} u_i)$, where $j, j' \in \{1, \ldots, q\}$ correspond to model numbers in M. Define

$$p_k(h, \rho_0^m) := \wedge_{i=0,\ldots,k} P_i$$

$k = 0, \ldots, m$. In this way we obtain a partition chain

$$\mathscr{P}(h, \rho_0^m) := \{M \leqslant p_0(h, \rho_0^m) \leqslant \cdots \leqslant p_m(h, \rho_0^m)\} \tag{4.2.1}$$

which clearly depends on the function h and on the points $\rho_0 < \rho_1 < \cdots \leqslant \rho_m$.

As before, given a class $c \subset M$, we denote by $p_k(h, \rho_0^m)(c)$ the partition induced on the class c; i.e. letting $p_k(h, \rho_0^m) = \{d_1, \ldots, d_n\}$, the partition $p_k(h, \rho_0^m)(c)$ consists of all non-empty sets $c \cap d_i, i = 1, \ldots, n$. We can now characterize all possible monitoring functions.

Theorem 4.2.1: *Let $\Sigma: D \to S(S_A)$ be a bounded interpreter, having the family $M = \{\Sigma_1, \ldots, \Sigma_q\}$ of potential models and the target tail set T. Let $\{I(\Sigma_i)\}_{i=1}^q$ be the class of successful input sequences of the family M. Then, statements* (i) *and* (ii) *below are equivalent.*

(i) *The function $h: S_A \to \Delta$ can be used as a monitoring function for Σ.*

(ii) *There is a comb χ of the class $\{I(\Sigma_i)\}_{i=1}^q$ for which the search algorithm* 4.1.1 *terminates at Steps 2 or 7, and the following holds in the notation of the algorithm.*

For every path $M \leqslant c_1 \leqslant c_2 \leqslant \cdots \leqslant c_s$ of the partition chain

$$\mathscr{P}(\chi) = \{M \leqslant P_1 \leqslant \cdots \leqslant P_s\},$$

the corresponding list of points $\omega_0 < \omega_1 < \cdots < \omega_s \in R(M)$ satisfies

$$p_k(h, \omega_0^k)(c_k) \geqslant P_{k+1}(c_k), \; k = 0, \ldots, s-1.$$

Proof: Let $\chi = \{u(1), \ldots, u(q)\}$ be the comb of part (ii) of Theorem 4.2.1, and denote by $\mu(\Sigma_i, h)$ the monitoring sequence generated when Σ_i is the active model and h is the monitoring function. In view of Theorem 4.1.1, a slight reflection shows that the present condition (ii) is equivalent to the following statement. There exists a strictly causal map $C: S(\Delta) \to S(S_A)$ that performs the assignment $\mu(\Sigma_i, h) \mapsto u(i)$, $i = 1, \ldots, q$. Clearly, the strictly causal map C is the same as a strictly causal controller C. Thus, condition (ii) is equivalent to the suitability of h as a monitoring function. □

Condition (ii) of Theorem 4.2.1 simply amounts to replacing the identity monitoring function by the function h in the search algorithm 4.1.1. This replacement affects only Step 6 of the algorithm, where the condition $P_{\mu, r+1} \geqslant P_{r+1}$ is replaced by the similar condition

$$p_r(h, \sigma_0^r) P_r \geqslant P_{r+1} \tag{4.2.2}$$

A function $h: S_A \to \Delta$ can be used as a monitoring function if and only if the search algorithm so modified terminates at Steps 2 or 7.

Thus, the class of all possible monitoring functions can be obtained while executing the (modified) search algorithm (4.1.1). A convenient monitoring function is then selected based on the real-time measurements it requires. Once the monitoring function has been selected, the controller design algorithm (3.3.1) yields an appropriate controller.

References

ALBERTS, B., BRAY, D., LEWIS, J., RAFF, M., ROBERTS, K., and WATSON, J. D., 1989, Molecular Biology of the Cell, second edition (New York: Garland).

ARNOLD, A., and NIVAT, M., 1980, Controlling behaviors of systems: some basic concepts and some applications. In *Mathematical Foundations of Computer Science, 1980: Proceedings of the 9th Symposium*, Rydzyna, Poland, 1–5 September 1980, edited by P. Dembinski. Lecture Notes in Computer Science, Vol. 88 (Berlin, Germany: Springer-Verlag).

EILENBERG, S., 1974, *Automata, Languages and Machines*, Vol. A (New York: Academic Press).

GINSBURG, S., 1962, *An Introduction to Mathematical Machine Theory* (Reading, Massachusetts, U.S.A.: Addison-Wesley), *The Mathematical Theory of Context Free Languages* (New York: McGraw Hill).

HAMMER, J., 1993, On corrective control of sequential machines. To be published.

HOARE, C. A. R., 1976, *Communicating Sequential Processes* (Englewood Cliffs, New Jersey, U.S.A.: Prentice Hall).

IEEE COMPUTER SYSTEMS SOCIETY (Conference publications), 1974, *Proceedings of the 1974 Conference on Biologically Motivated Automata Theory*, McLean, Virginia, U.S.A.

KAUFFMAN, S. A., 1969, Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, **22**, 437–467.

LINDENMAYER, A., 1968, Mathematical models for cellular interactions in development, Parts I and II. *Journal of Theoretical Biology*, **18**, 280–315.

MILNER, R., 1980, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science (Berlin, Germany: Springer-Verlag).

NEUMANN, J. VON., 1966, *The Theory of Self-reproducing Automata*, edited by A. W. Burks (Urbana, Illinois, U.S.A.: University of Illinois Press).

RAMADGE, P. J., and WONHAM, W. M., 1987, Supervisory control of a class of discrete event processes. *SIAM Journal of Control and Optimization*, **25**, 206–230.

RASHEVSKY, N., 1948, *Mathematical Biophysics* (Chicago, Illinois, U.S.A.: The University of Chicago Press).

ROZENBERG, G., and SALOMAA, A., 1975, *L Systems*, Lecture Notes in Computer Science, Vol. 15 (Berlin, Germany: Springer-Verlag).

SUGITA, M., 1963, Functional analysis of chemical systems in vivo using a logical circuit equivalent. II. The idea of a molecular automaton. *Journal of Theoretical Biology*, **4**, 179–189.