A. BERGERON

A unified approach to control problems in discrete event processes

Informatique théorique et applications, tome 27, nº 6 (1993), p. 555-573

<http://www.numdam.org/item?id=ITA_1993__27_6_555_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 27, n° 6, 1993, p. 555 à 573)

A UNIFIED APPROACH TO CONTROL PROBLEMS IN DISCRETE EVENT PROCESSES (*)

by A. BERGERON (1)

Communicated by A. ARNOLD

Abstract. – *Control theory of discrete event processes has been developped in recent years by Ramadge and Wonham. Their framework is based on formal language theory and control theoretic ideas such as* controllability *and* observability. *In this paper, we focus on the automata modeling the processes and the specifications, rather than on the sets of possible or acceptable sequences. By adopting this slightly different point of view, we show that the main results of control theory can be obtained in a simple, uniform way.*

Centralized control problem, observation problems and distributed control problems are stated as particular cases of what we call design problems. *With the help of two basic constructions on automata, we show that necessary and sufficient conditions for the existence of solutions are all particular cases of a general lemma on the solutions of design problems. All the results are effective, in the sense that they yield immediate algorithms for the design of controllers.*

Résumé. – *Un processus discret est un système qui change abruptement d'état selon l'occurrence de suites d'événements. La théorie du contrôle de tels processus a été développée dans les dernières années par Ramadge et Wonham. Fondée à la fois sur la théorie des langages formels et sur des idées provenant de la théorie du contrôle, cette théorie explore l'existence et la construction de solutions à des problèmes de contrôle centralisé – ou réparti – dans un contexte d'observation partielle.*

Dans cet article, nous adoptons un point de vue légèrement différent en centrant l'attention sur les automates qui modélisent les processus et leur spécification. Ce nouveau point de vue permet d'obtenir l'ensemble des résultats de la théorie d'une manière simple et uniforme. Nous exposons d'abord un cadre général qui permet d'exprimer les problèmes de contrôle ou d'observation avec un formalisme identique. Nous montrons ensuite que deux constructions de base sur les automates permettent d'obtenir des conditions nécessaires et suffisantes pour la solution de ces problèmes. Tous les résultats obtenus sont constructifs, au sens où les énoncés fournissent immédiatement des algorithmes pour la construction de contrôleurs ou d'observateurs lorsqu'ils existent.

This work was partially supported by grants from BNR Ltd, FCAR of Québec, and NSERC of Canada.

(*) Received June 1992, accepted April 1993.

(1) Département de mathématiques et d'informatique, Université du Québec à Montréal, C. P. 8888 Succ 'A', Montréal, Canada, H3C 3P8.

1. INTRODUCTION

A *discrete event process* is a system that changes state according to the occurrence of a sequence of events. The focus is on the order and the nature of the events affecting the system rather than their time of occurrence or their duration. The central problems can be described as:

1) Given a discrete process and a specification of its acceptable behaviors, is it possible to construct a new process, called a *controller*, that will implement the specification by prohibiting or allowing certain events?

2) How is the solution affected if the controller does not have complete information about the occurrence of events in the original process?

3) When is it possible to distribute the controlling task among several controllers?

Various solutions to these design problems have been found in the recent years within the framework developed by Ramadge and Wonham. Necessary and sufficient conditions for the existence of solutions are expressed in terms of formal language theory and control theoretic ideas such as *controllability* and *observability* [3; 4], [6], [5]. Controllability and observability are defined, in these works, as properties of the acceptable sequences of events (in the specification) with respect to possible sequences of events (in the original process).

In this paper, we focus on the automata modeling the processes and the specifications rather than on the sets of possible or acceptable sequences. By adopting this slightly different point of view, we will show that the main results of control theory can be obtained in a simple, uniform way. This approach also provides an interesting bridge to the theory of transition systems and concurrent processes developed around the Arnold-Nivat model [1], [2].

Section 2 reviews the basic tools used in modelling discrete processes with deterministic automata. In Section 3, we present a general setting for design problems as solutions of equations of the form $\mathbf{Z}=\mathbf{D}\times\mathbf{X}$ where $\mathbf{Z}$ is a specification of a process $\mathbf{D}$ and $\mathbf{X}$ belongs to a restricted class of automata. We apply these tools in Section 4 and 5 to obtain constructive solutions of control and observation problems. Section 6 discusses control under partial observation and distributed control.

2. MODELING DISCRETE EVENT PROCESS WITH AUTOMATA

Let Σ be a finite set whose elements are called *events*. The set of all finite sequences of events is denoted by Σ^*, and the empty sequence is denoted by λ. The concatenation of two sequence x and y is denoted by xy.

Let $\mathbf{S}$ be a set whose elements are called *states*. An *automaton* $\mathbf{V}$ on the set Σ of events is given by an arbitrary partial function – multiplicatively denoted by a dot "$\cdot$" – called a *transition function*:

$$\cdot : \mathbf{S} \times \Sigma \to \mathbf{S}$$

which, when defined, associates to a pair $(\mathbf{s}, \sigma)$ the state $\mathbf{s} \cdot \sigma$. Among the states $\mathbf{S}$, we distinguish an *initial state* $\mathbf{i}$ and of a subset $\mathbf{M} \subseteq \mathbf{S}$ of *marked states.*

It will be useful to consider the extension of a transition function to the set Σ^* of all sequences of events. Every transition function can be uniquely extended to a partial function:

$$\cdot : \mathbf{S} \times \Sigma^* \to \mathbf{S}$$

that has the following properties:

(i) $\mathbf{s} \cdot \lambda = \mathbf{s}$.

(ii) $(\mathbf{s} \cdot x) \cdot \sigma = \mathbf{s} \cdot (x\sigma)$ whenever the left hand side is defined.

The language $L(\mathbf{V})$ *recognized* by the automaton $\mathbf{V}$ is the subset of Σ^* defined by:

$$L(\mathbf{V}) = \{ x \mid \mathbf{i} \cdot x \in \mathbf{M} \}.$$

Finally, we will denote by $\bar{\mathbf{V}}$ the automaton obtained by marking all the states of an automaton $\mathbf{V}$. We have:

$$L(\bar{\mathbf{V}}) = \{ x \mid \mathbf{i} \cdot x \text{ is defined} \}.$$

An automaton is often described by a *directed graph* whose vertices are labeled by the states $\mathbf{S}$ and whose arrows are labeled by the symbols σ of Σ. For each defined transition $\mathbf{s} \cdot \sigma = \mathbf{s}'$, we have the corresponding arrow:

$$\mathbf{s} \xrightarrow{\sigma} \mathbf{s}'$$

If x is a sequence of events, then $x \in L(\bar{\mathbf{V}})$ or x is defined in $\mathbf{V}$ if there is a path labeled by x beginning at the initial state $\mathbf{i}$. If the state reached by the path is marked, then $x \in L(\mathbf{V})$.

DEFINITION 2.1: *Non-blocking automata.*

(i) If L is a subset of Σ^* then $Pref_L = \{ x \in \Sigma^* \mid xy \in L \}$ is the set of *prefixes* of sequences in L. Since y can be the empty sequence, we always have $L \subseteq Pref_L$.

(ii) An automaton $\mathbf{V}$ is said to be *non-blocking* if

$$L(\bar{\mathbf{V}}) = Pref_{L(\mathbf{V})}. \quad \blacksquare$$

Consider, for example, the following automaton on $\Sigma = \{ a, b \}$:

$$\begin{tikzcd}
\mathbf{i} \arrow[r, "a"] & \mathbf{s} \arrow[r, "b"] \arrow[d, "a"'] & \mathbf{t} \\
& \mathbf{r} &
\end{tikzcd}$$

where only $\mathbf{r}$ is marked. This automaton is blocking since the sequence ab belongs to $L(\bar{\mathbf{V}})$ but is not a prefix of any sequence in $L(\mathbf{V})$.

In the following sections, we will be mainly concerned with the design and analysis of special classes of automata. For that purpose, we need ways to compare them, and the usual equivalences defined in automata theory (mostly based on the language recognized by an automaton) are not powerful enough. In design problems, we will need the non-blocking property to be preserved under equivalence.

DEFINITION 2.2: *Partial order between automata.*

The relation $\mathbf{V} \leqq \mathbf{W}$ holds whenever

(i) $L(\mathbf{V}) \subseteq L(\mathbf{W})$ and

(ii) $L(\bar{\mathbf{V}}) \subseteq L(\bar{\mathbf{W}})$.

When $\mathbf{V} \leqq \mathbf{W}$ and $\mathbf{W} \leqq \mathbf{V}$, we simply write $\mathbf{V} = \mathbf{W}$. $\blacksquare$

An immediate consequence of this definition is:

PROPOSITION 2.3: *If* $\mathbf{V} = \mathbf{W}$, *then* $\mathbf{V}$ *is non-blocking if and only if* $\mathbf{W}$ *is non-blocking.* $\blacksquare$

Given two automata $\mathbf{V}$ and $\mathbf{W}$, with states $\mathbf{S_V}$ and $\mathbf{S_W}$, initial states $\mathbf{i_V}$ and $\mathbf{i_w}$, and marked states $\mathbf{M_V}$ and $\mathbf{M_W}$. The *product* $\mathbf{V} \times \mathbf{W}$ has states $\mathbf{S} = \mathbf{S_V} \times \mathbf{S_W}$ with the transition function:

$$\cdot : \mathbf{S} \times \Sigma^* \to \mathbf{S} \text{ where } (\mathbf{v}, \mathbf{w}) \cdot x = (\mathbf{v} \cdot x, \mathbf{w} \cdot x) \text{ whenever } \textit{both} \text{ are defined.}$$

The initial state of the product is $\mathbf{i} = (\mathbf{i_V}, \mathbf{i_W})$ and the marked states

$$\mathbf{M} = \{ (\mathbf{v}, \mathbf{w}) \mid \mathbf{v} \in \mathbf{M_v} \text{ and } \mathbf{w} \in \mathbf{M_w} \}.$$

We can think of the two processes as simultaneously functioning: they synchronize on the occurrence of every event. Of course, in many examples, one of the process will 'generate' the events and the other will 'react' to them, but the modelling is the same in both cases. The following theorem expresses a basic relation between products and inequalities:

THEOREM 2.4: $\mathbf{V} \leqq \mathbf{W}$ *if and only if* $\mathbf{V} = \mathbf{W} \times \mathbf{V}$.

Proof: If $\mathbf{V} \leqslant \mathbf{W}$ then,

$$L(\mathbf{V}) = L(\mathbf{V}) \cap L(\mathbf{W}) = L(\mathbf{V} \times \mathbf{W})$$

since a state is marked in the product if and only if it is marked by both automata. We also have

$$L(\bar{\mathbf{V}}) = L(\bar{\mathbf{V}}) \cap L(\bar{\mathbf{W}}) = L(\bar{\mathbf{V}} \times \bar{\mathbf{W}}) = L(\overline{\mathbf{V} \times \mathbf{W}})$$

since those last automata are identical. On the other hand, if $\mathbf{V} = \mathbf{W} \times \mathbf{V}$, we have easily

$$L(\mathbf{V}) = L(\mathbf{V}) \cap L(\mathbf{W}) \subseteq L(\mathbf{W}) \text{ and}$$
$$L(\bar{\mathbf{V}}) = L(\bar{\mathbf{V}}) \cap L(\bar{\mathbf{W}}) \subseteq L(\bar{\mathbf{W}}). \quad \blacksquare$$

Further properties of these operations are easily verified:

PROPOSITION 2.5: *Let* $\mathbf{Y}$, $\mathbf{Z}$, $\mathbf{V}$ *and* $\mathbf{W}$ *be automata. Then*

(i) *If* $\mathbf{Y} \leqq \mathbf{V}$ *and* $\mathbf{Z} \leqq \mathbf{W}$ *then* $\mathbf{Y} \times \mathbf{Z} \leqq \mathbf{V} \times \mathbf{W}$.

(ii) $\mathbf{Z} \leqq \mathbf{V} \times \mathbf{W}$ *if and only if* $\mathbf{Z} \leqq \mathbf{V}$ *and* $\mathbf{Z} \leqq \mathbf{W}$. $\blacksquare$

3. A GENERAL FRAME FOR DESIGN PROBLEMS

Let **D** be an 'existing' discrete process and suppose we want to restrict, observe, or modify the behavior of this process. Such design problems can be solved by constructing one or more new processes that will function simultaneously with **D**, and that will be able to prevent, authorize, observe or report the occurrence of events in **D**.

In order to state the problem formally, we first suppose that the process **D** is modeled by an automaton such that any undefined transition for a pair $(\mathbf{s}, \sigma)$ corresponds to the fact that the event σ is *impossible* in state $\mathbf{s}$. All the *possible* sequences of events are taken into account in the model. If the behavior of **D** is to be restricted or observed, we need a specification of what are the acceptable behaviors or what are the properties to observe.

Such a specification will again be modeled by an automaton $\mathbf{Z}$, and we assume that $\mathbf{Z} \leqq \mathbf{D}$. This restriction is natural since if there was a sequence of events defined in $\mathbf{Z}$ but not in $\mathbf{D}$, it would correspond to an impossible sequence by our assumption on $\mathbf{D}$.

Finally, in order to solve design problems, we want to construct automata that will be able to perform specialized tasks. For a given task (such as controlling, observing, reporting, etc.) the identification of these specialized automata will correspond to a particular class $\mathscr{A}$ of automata.

In this framework, we can state the *general design problem* as:

Let $\mathscr{A}$ be a class of automata. Given a process $\mathbf{D}$, and a specification $\mathbf{Z}$ such that $\mathbf{Z} \leqslant \mathbf{D}$, does there exist an automaton $\mathbf{X} \in \mathscr{A}$ such that

$$\mathbf{Z} = \mathbf{D} \times \mathbf{X}?$$

If the class $\mathscr{A}$ is unrestricted, Theorem 2.4 gives immediately the trivial solution $\mathbf{X} = \mathbf{Z}$. By restricting the solutions to have certain additional properties on their states or transition functions, we will solve many variants of control and analysis problems. These solutions will be constructive in the sense that the existence theorems will give explicit construction for $\mathbf{X}$, and are based on the following result:

LEMMA 3.1: *Let $\mathscr{A}$ be a class of automata and suppose that there exists a mapping $\mathbf{A}$ which associates to every automata $\mathbf{Z}$ an automaton $\mathbf{A_Z} \in \mathscr{A}$, such that $\mathbf{Z} \leqq \mathbf{A_Z}$ and for each $\mathbf{X} \in \mathscr{A}$,*

$$\mathbf{Z} \leqq \mathbf{X} \Rightarrow \mathbf{A_Z} \leqq \mathbf{X}.$$

Then, if $\mathbf{Z} \leqq \mathbf{D}$, there exists an automaton $\mathbf{X} \in \mathscr{A}$ such that:

$$\mathbf{Z} = \mathbf{D} \times \mathbf{X} \quad \textit{if and only if } \mathbf{Z} = \mathbf{D} \times \mathbf{A_Z}.$$

Proof: The *if* part is immediate since we assumed that $\mathbf{A_Z} \in \mathscr{A}$. On the converse, the fact that $\mathbf{Z} \leqq \mathbf{A_Z}$ and $\mathbf{Z} \leqq \mathbf{D}$, gives us:

$$\mathbf{Z} \leqq \mathbf{D} \times \mathbf{A_Z}.$$

Suppose now that there exists an automaton $\mathbf{X} \in \mathscr{A}$ such that $\mathbf{Z} = \mathbf{D} \times \mathbf{X}$. Then $\mathbf{Z} \leqq \mathbf{X}$ so, we have $\mathbf{A_Z} \leqq \mathbf{X}$. Multiplying by $\mathbf{D}$ on both sides we get:

$$\mathbf{D} \times \mathbf{A_Z} \leqq \mathbf{D} \times \mathbf{X} = \mathbf{Z}. \quad \blacksquare$$

Thus, if a suitable mapping (or construction) $\mathbf{A}$ is provided for the class $\mathscr{A}$, Lemma 3.1 gives elegant necessary and sufficient conditions for the design problem with respect to the class $\mathscr{A}$. In the next sections, we will discuss two basic constructions corresponding respectively to the control and observation problems.

4. THE CENTRALIZED CONTROL PROBLEM

Suppose that $\mathbf{D}$ is a process on the set of events Σ. Among those events, some of them may be *controllable*, that is there exists physical means to prevent their occurrence. For example, a communication process can suspend message sending until it receives a given signal. Given such a subset Σ_c of controllable events (and its complement the *uncontrollable* events Σ_{uc}), the centralized control problem can be stated informally as designing an automaton which changes state according to the sequence of events generated by $\mathbf{D}$, and prevent or authorize controllable events according to its state.

If an automaton is to perform such a controlling task, its only 'impossible' events must be exactly those it prevented, thus we have the following definition:

DEFINITION 4.1: *Control Automata*

An automaton $\mathbf{C}$ is a *control automaton* with respect to Σ_c if, for all states $\mathbf{s}$,

$$\sigma \in \Sigma_{uc} \text{ implies } \mathbf{s} \cdot \sigma \text{ is defined}$$

The class of all control automata (with respect to Σ_c) will be noted $\mathscr{A}_c$.

If $\mathbf{C}$ is a control automaton, we can define the following control strategy for $\mathbf{C}$: in state $\mathbf{s}$, prevent the occurrence of any event such that $\mathbf{s} \cdot \sigma$ is undefined, and allow all others. It is clear, from the definition of control automata, that such a strategy is always realizable.

The *centralized control problem* can be formulated as:

Let $\mathscr{A}_c$ be the class of control automata with respect to Σ_c. Given a process $\mathbf{D}$, and a specification $\mathbf{Z}$ such that $\mathbf{Z} \leq \mathbf{D}$, does there exist an automaton $\mathbf{X} \in \mathscr{A}_c$ such that

$$\mathbf{Z} = \mathbf{D} \times \mathbf{X}?$$

Consider now the following construction $\mathbf{C_Z}$ on an automaton $\mathbf{Z}$: we add a new unmarked state $\mathbf{u}$ to $\mathbf{Z}$ and whenever $\mathbf{s} \cdot \sigma$ is undefined – including the new state $\mathbf{u}$ – and σ is uncontrollable, we set $\mathbf{s} \cdot \sigma = \mathbf{u}$ in $\mathbf{C_Z}$. With this construction we have:

THEOREM 4.2: *The mapping* $\mathbf{C_Z}$ *has the following properties:*

(i) $\mathbf{C_Z} \in \mathscr{A}_c$.

(ii) $\mathbf{Z} \leqq \mathbf{C_Z}$.

(iii) *For each* $\mathbf{X} \in \mathscr{A}_c$, $\mathbf{Z} \leqq \mathbf{X} \Rightarrow \mathbf{C_Z} \leqq \mathbf{X}$.

Proof: (i) This is immediate from the construction of $\mathbf{C_Z}$.

(ii) Every path in $\mathbf{Z}$ is also a path in $\mathbf{C_Z}$.

(iii) Suppose that $\mathbf{Z} \leqq \mathbf{X}$, we first have $L(\mathbf{C_Z}) = L(\mathbf{Z}) \subseteq L$ $(\mathbf{X})$ since any path in $\mathbf{C_Z}$ which goes in a marked state does not go through the new state $\mathbf{u}$ – there are no outgoing arrows from $\mathbf{u}$.

Now, let $x \in L(\bar{\mathbf{C}}_z)$, if $\mathbf{i} \cdot x \neq \mathbf{u}$ then $x \in L(\bar{\mathbf{Z}}) \subseteq L(\bar{\mathbf{X}})$. Otherwise, if $\mathbf{i} \cdot x = \mathbf{u}$, x can be decomposed as $x = x' \sigma y$ such that $x' \in L(\bar{\mathbf{Z}})$ and σy is a sequence of uncontrollable events. But $x' \in L(\bar{\mathbf{X}})$, so $x' \sigma y \in L(\bar{\mathbf{X}})$ since $\mathbf{X}$ is a control automaton. ■

COROLLARY 4.3: *The centralized control problem has a solution if and only if* $\mathbf{Z} = \mathbf{D} \times \mathbf{C_Z}$. ■

Notes: Other sets of necessary and sufficient conditions for the existence of a solution to the centralized control problem appear in [3]. These conditions are based on the notion of *controllable* languages. A language K is *controllable* with respect to L to if

$$x \in Pref_K \text{ and } x\sigma \in L \text{ with } \sigma \text{ uncontrollable implies } x\sigma \in Pref_K$$

and their results are established by considering the controllability of $L(\mathbf{Z})$ with respect to $L(\bar{\mathbf{D}})$.

5. THE OBSERVATION PROBLEM

Suppose that $\mathbf{D}$ is a process on the set of events Σ. Among those events, the occurrence of some of them may be *unobservable*. For example, a communication process may not be able to tell if a given message has been received. Given a specification $\mathbf{Z}$ for the process and such a subset Σ_{uo} of unobservable events (and its complement the *observable* events Σ_o), the observation problem is whether or not certain properties of a system can be determined without knowledge about the occurrence of unobservable events.

In order to perform such an observation task, an automaton should have the same 'reactions' whether or not an unobservable event occurred. The simplest observation automata are the ones that loop on all (defined) unobservable events:

DEFINITION 5.1: *Observation Automata*

An automaton **O** is an *observation automaton* with respect to Σ_o if, for all states **s**,

$$\mathbf{s} \cdot \sigma \text{ is defined and } \sigma \in \Sigma_{uo} \text{ implies } \mathbf{s} \cdot \sigma = \mathbf{s}.$$

The class of all observation automata (with respect to Σ_o) will be noted $\mathscr{A}_o$.

The *observation problem* can be formulated as:

Let $\mathscr{A}_o$ be the class of observation automata with respect to Σ_o. Given a process **D**, *and a specification* **Z** *such that* $\mathbf{Z} \leqq \mathbf{D}$, *does there exist an automaton* $\mathbf{X} \in \mathscr{A}_o$ *such that*

$$\mathbf{Z} = \mathbf{D} \times \mathbf{X}?$$

This problem will be solved in the same fashion as the control problem. We want to construct the smallest observation automata $\mathbf{O_Z}$ that contains a given automata **Z**. The idea behind the construction is the following: suppose an observable event occurs in **Z** changing its state to **s**, in order to ignore unobservable events we have to assume that **Z** could be either in state **s** or in any other state reachable from **s** by a sequence of unobservable events. For example, the initial state of $\mathbf{O_Z}$ will be the *set* of all states reachable from the initial state of **Z** with sequences of unobservable events.

More formally, let **S** be the set of states of **Z**, with initial state **i** and marked states **M**. Let Σ_{uo}^* be the set of unobservable sequences. The states of the automaton $\mathbf{O_Z}$ are non-empty subsets $\mathscr{P}^+(\mathbf{S})$ of **S**. The initial state is:

$$\mathbf{I} = \{\mathbf{i} \cdot u \mid \mathbf{i} \cdot u \text{ is defined and } u \in \Sigma_{uo}^*\}$$

and the marked states of $\mathbf{O_Z}$ are all subsets that contain at least one marked state of **Z**. The transition function is defined as:

$$\circ : \mathscr{P}^+(\mathbf{S}) \times \Sigma \to \mathscr{P}^+(\mathbf{S})$$

Let $\mathbf{T} \subseteq \mathbf{S}$, if $\mathbf{s} \cdot \sigma$ is defined in **Z** for at least one **s** in **T** then $\mathbf{T} \circ \sigma$ is defined and

(i) If $\sigma \in \Sigma_o, \mathbf{T} \circ \sigma = \{\mathbf{s} \cdot \sigma u \mid \mathbf{s} \in \mathbf{T}, \mathbf{s} \cdot \sigma u \text{ is defined and } u \in \Sigma_{uo}^*\}$

(ii) If $\sigma \in \Sigma_{uo}$, $\mathbf{T} \circ \sigma = \mathbf{T}$

otherwise $\mathbf{T} \circ \sigma$ is undefined.

In order to prove that the construction $\mathbf{O_Z}$ satisfies the hypothesis of Lemma 3.1, we need the following results on the relations between $\mathbf{Z}$ and $\mathbf{O_Z}$:

LEMMA 5.2: *If* $\mathbf{i}.x$ *is defined in* $\mathbf{Z}$, *then* $\mathbf{I}\circ x$ *is defined in* $\mathbf{O_Z}$ *and* $\mathbf{i}\cdot x\in\mathbf{I}\circ x$.

Proof: The proof is an induction on the length of x. If $x=\lambda$, then $\mathbf{i}\cdot x=\mathbf{i}$ and $\mathbf{I}\circ x=\mathbf{I}$ are always defined and we have, by construction, that $\mathbf{i}\in\mathbf{I}$. Now, let $x=y\sigma$ be defined in $\mathbf{Z}$, then y is defined in $\mathbf{Z}$ and we can suppose that $\mathbf{I}\circ y$ is defined and $\mathbf{i}.y\in\mathbf{I}\circ y$. Since $(\mathbf{i}\cdot y)\cdot\sigma$ is defined we have immediately $(\mathbf{I}\circ y)\circ\sigma$ is defined.

If σ is observable we have immediately that $(\mathbf{i}\cdot y)\cdot\sigma\in(\mathbf{I}\circ y)\circ\sigma$. On the other hand, if σ is unobservable then $(\mathbf{I}\circ y)\circ\sigma=\mathbf{I}\circ y$ so we have to prove that $(\mathbf{i}\cdot y)\cdot\sigma\in\mathbf{I}\circ y$. But if an accessible state of $\mathbf{O_Z}$ contains a given state $\mathbf{s}$ it contains, by construction, all the states reachable from $\mathbf{s}$ by unobservable events, so if $\mathbf{i}\cdot y\in\mathbf{I}\circ y$ we have $(\mathbf{i}\cdot y)\cdot\sigma\in\mathbf{I}\circ y$. ■

The second lemma expresses a deeper property: each sequence defined in $\mathbf{O_Z}$ looks like some sequence in $\mathbf{Z}$ in the following sense. Consider the projection

$$\pi:\Sigma^*\to\Sigma^*$$

that erases unobservable events, then:

LEMMA 5.3: *If* $\mathbf{I}\circ x$ *is defined in* $\mathbf{O_Z}$ *and* $\mathbf{t}\in\mathbf{I}\circ x$, *there exists a sequence* x' *defined in* $\mathbf{Z}$ *such that* $\mathbf{i}\cdot x'=\mathbf{t}$ *and* $\pi(x)=\pi(x')$.

Proof: The proof is an induction on the number of observable events of x.

Suppose that $\pi(x)=\lambda$, that is $x\in\Sigma_{uo}^*$, then $\mathbf{I}\circ x=\mathbf{I}=\{\mathbf{i}\cdot u\,|\,\mathbf{i}\cdot u$ is defined and $u\in\Sigma_{uo}^*\}$ thus if $\mathbf{t}\in\mathbf{I}\circ x$, there exists a sequence $u'\in\Sigma_{uo}^*$ defined in $\mathbf{Z}$ such that $\mathbf{t}=\mathbf{i}\cdot u'$ and $\pi(u')=\lambda=\pi(x)$.

Suppose now that $x=y\sigma z$ with σ being the last observable event of x, that is $\pi(x)=\pi(y)\sigma$. Let $\mathbf{t}$ be a state in

$$\mathbf{I}\circ x=\mathbf{I}\circ y\sigma=(\mathbf{I}\circ y)\circ\sigma=\{\mathbf{s}\cdot\sigma u\,|\,\mathbf{s}\in\mathbf{I}\circ y,\ \mathbf{s}\cdot\sigma u \text{ is defined and } u\in\Sigma_{uo}^*\}$$

so there exists an $\mathbf{s}$ in $\mathbf{I}\circ y$ and a u' in Σ_{uo}^* such that $\mathbf{s}\cdot\sigma u'=\mathbf{t}$.
By the induction hypothesis, there is a y' defined in $\mathbf{Z}$ such that $\mathbf{i}\cdot y'=\mathbf{s}$ and $\pi(y)=\pi(y')$. But the fact that $\mathbf{i}\cdot y'=\mathbf{s}$, and $\mathbf{s}\cdot\sigma u'$ is defined, implies that $\mathbf{i}\cdot y'\sigma u'=\mathbf{t}$ is defined in $\mathbf{Z}$, and $\pi(y'\sigma u')=\pi(y')\sigma=\pi(y)\sigma=\pi(x)$. ■

We can now prove:

THEOREM 5.4: *The mapping* $\mathbf{O_Z}$ *has the following properties:*

(i) $\mathbf{O_Z} \in \mathscr{A}_o$.

(ii) $\mathbf{Z} \leqq \mathbf{O_Z}$.

(iii) *For each* $\mathbf{X} \in \mathscr{A}_o$, $\mathbf{Z} \leqq \mathbf{X} \Rightarrow \mathbf{O_Z} \leqq \mathbf{X}$.

Proof: (i) is immediate by construction, since $\mathbf{O_Z}$ loops on all unobservable events.

(ii) If $x \in L(\bar{\mathbf{Z}})$ then $\mathbf{i}.x$ is defined in $\mathbf{Z}$, so $\mathbf{I} \circ x$ is defined in $\mathbf{O_Z}$ by Lemma 5.2. If $x \in L(\mathbf{Z})$ then the state $\mathbf{i} \cdot x$ is marked, and since $\mathbf{i} \cdot x \in \mathbf{I} \circ x$, the state $\mathbf{I} \circ x$ will be marked.

(iii) Suppose that $\mathbf{Z} \leqq \mathbf{X}$ with $\mathbf{X} \in \mathscr{A}_o$. First we will show that if x is defined in $\mathbf{O_Z}$ then $\pi(x)$ is defined in $\mathbf{X}$. If x is defined in $\mathbf{O_Z}$ then, by Lemma 5.3, there exists an x' defined in $\mathbf{Z}$ such that $\pi(x') = \pi(x)$. But since $\mathbf{Z} \leqq \mathbf{X}$, x' is in $\mathbf{X}$. Since $\mathbf{X}$ loops on every unobservable event, $\pi(x')$ can be traced in $\mathbf{X}$ by following the same path as x' and skipping the unobservable loops.

We will prove that $L(\bar{\mathbf{O}}_z) \subseteq L(\bar{\mathbf{X}})$ by induction on the number of unobservable events of sequences in $L(\bar{\mathbf{O}}_z)$. If $x \in L(\bar{\mathbf{O}}_z)$, and x has no unobservable events, then $\pi(x) = x$ and, by the preceding remark, we have $x \in L(\bar{\mathbf{X}})$.

Suppose now that $x = y \sigma z$ with σ being the last unobservable event of x. We will show that both $y\sigma$ and yz are defined in $\mathbf{X}$. Since $\mathbf{X}$ loops on every unobservable σ, we will be able to conclude that $y \sigma z$ is defined in $\mathbf{X}$.

If $x = y \sigma z$ is defined in $\mathbf{O_Z}$ with σ being the last unobservable event of x, then $\pi(x) = \pi(y) z$ is defined in $\mathbf{X}$. By the induction hypothesis, y also is defined in $\mathbf{X}$, and since $\pi(y)$ and y reach the same state in $\mathbf{X}$, if $\pi(y) z$ is defined, then yz is also defined.

On the other hand, if $x = y \sigma z$ is defined in $\mathbf{O_Z}$, then $y\sigma$ is also defined, and $\mathbf{s} \cdot \sigma$ is defined in $\mathbf{Z}$ for at least one $\mathbf{s}$ in $\mathbf{I} \circ y$. By Lemma 5.3, there exists a y' defined in $\mathbf{Z}$ such that $\mathbf{i} \cdot y' = \mathbf{s}$ and $\pi(y') = \pi(y)$. But since $\mathbf{s} \cdot \sigma$ is defined in $\mathbf{Z}$, and $\mathbf{i} \cdot y' = \mathbf{s}$, we have that $\mathbf{i} \cdot y' \sigma$ is defined in $\mathbf{Z}$, implying that $y' \sigma$ is defined in $\mathbf{X}$. By the induction hypothesis, y is defined in $\mathbf{X}$, and since $\pi(y') = \pi(y)$, y and y' reach the same state in $\mathbf{X}$. Thus, if $y' \sigma$ is defined so is $y \sigma$.

In order to prove that $L(\mathbf{O_Z}) \subseteq L(\mathbf{X})$, we note that if $x \in L(\mathbf{O_Z})$ then x is in $L(\bar{\mathbf{X}})$ by the preceding paragraphs, and we only have to prove that x reaches a marked state of $\mathbf{X}$. If $x \in L(\mathbf{O_Z})$ then $\mathbf{I} \circ x$ is marked in $\mathbf{O_Z}$, so there is a marked state $\mathbf{t}$ in $\mathbf{Z}$ such that $\mathbf{t} \in \mathbf{I} \circ x$. By Lemma 5.3, there exists a sequence x' defined in $\mathbf{Z}$ such that $\mathbf{i} \cdot x' = \mathbf{t}$ and $\pi(x) = \pi(x')$. But x' and x are both defined in $\mathbf{X}$ and reach the same state since $\pi(x)=\pi(x')$. Since x' reaches a marked state in $\mathbf{Z}$, it reaches a marked state in $\mathbf{X}$. ■

COROLLARY 5.5: *The observation problem has a solution if and only if* $\mathbf{Z}=\mathbf{D}\times\mathbf{O_Z}$. ■

Notes: In the framework of the theory developed by Ramadge and Wonham, the observation problem is treated through the notion of *observability* [6], and Corollary 5.5 can be used as an algorithm to determine if $L(\mathbf{Z})$ is an observable language with respect to $L(\bar{\mathbf{D}})$.

Finally, we have the following propositions relating the constructions $\mathbf{O_Z}$ and $\mathbf{C_Z}$. The first one says that the 'observation' construction preserves the property of being a control automaton:

PROPOSITION 5.6: *Let* $\mathscr{A}_c$ *be the class of control automata with respect to* Σ_c. *If* $\mathbf{C}\in\mathscr{A}_c$ *and* $\mathbf{O_C}$ *is the minimal observation automaton with respect to* Σ_o *that contains* $\mathbf{C}$, *then* $\mathbf{O_C}\in\mathscr{A}_c$.

Proof: Let $\mathbf{T}$ be a state of $\mathbf{O_C}$ and σ an uncontrollable event. Recall that $\mathbf{T}$ is a non-empty subset of the states of $\mathbf{C}$ and $\mathbf{T}\circ\sigma$ is defined if $\mathbf{s}\cdot\sigma$ is defined in $\mathbf{C}$ for at least one $\mathbf{s}$ in $\mathbf{T}$. If σ is uncontrollable, the fact that $\mathbf{C}$ is a control automaton implies that $\mathbf{s}\cdot\sigma$ is always defined. ■

Applying the constructions in the reverse order does not always yield an observation automaton. We must take care of the possibility that some unobservable and uncontrollable events are not defined:

PROPOSITION 5.7: *Let* $\mathscr{A}_o$ *the class of control automata with respect to* Σ_o. *If* $\mathbf{O}\in\mathscr{A}_o$ *and* $\mathbf{C_O}$ *is the minimal control automaton with respect to* Σ_c *that contains* $\mathbf{C}$, *then* $\mathbf{C_O}\in\mathscr{A}_o$ *if and only if the transition function of* $\mathbf{O}$ *is defined for all* $\sigma\in\Sigma_{uc}\cap\Sigma_{uo}$.

Proof: If the transition function of $\mathbf{O}$ is defined for all $\sigma\in\Sigma_{uc}\cap\Sigma_{uo}$, then we do not add any unobservable transition in the construction of $\mathbf{C_O}$. So if $\mathbf{O}$ did loop on all unobservable events, $\mathbf{C_O}$ has the same property. Conversely, if there is a state $\mathbf{s}$ in $\mathbf{O}$ such that $\mathbf{s}\cdot\sigma$ is not defined with $\sigma\in\Sigma_{uc}\cap\Sigma_{uo}$, then in constructing $\mathbf{C_O}$ we will add a new transition $\mathbf{s}\cdot\sigma=\mathbf{u}$ and the resulting automaton will not be in $\mathscr{A}_o$. ■

6. CONTROL UNDER PARTIAL OBSERVATION AND DISTRIBUTED CONTROL

6.1 The Control Problem with Partial Observation

In this section, we want to characterize solutions to the problem of constructing a controller that can function under partial observation. Here we are given two subsets of Σ : Σ_c are the controllable events, and Σ_o are the observable events. We want to construct controllers that prevent or authorize events in Σ_c while getting only the information in Σ_o.

The *control problem with partial observation* is the following:

Let $\mathscr{A}_{oc} = \mathscr{A}_o \cap \mathscr{A}_c$ *be the class of observation automata with respect to* Σ_o *and control automata with respect to* Σ_c. *Given a process* $\mathbf{D}$, *and a specification* $\mathbf{Z}$ *such that* $\mathbf{Z} \leqq \mathbf{D}$, *does there exist an automaton* $\mathbf{X} \in \mathscr{A}_{oc}$ *such that*

$$\mathbf{Z} = \mathbf{D} \times \mathbf{X}?$$

We will solve this problem by combining the results of Theorem 4.2 on the control problem, and Theorem 5.4 on the observation problem. We define, for each automaton $\mathbf{Z}$, the automaton

$$\mathbf{OC_Z} = \mathbf{O_{C_Z}}.$$

THEOREM 6.1: *The mapping* $\mathbf{OC_Z}$ *has the following properties*:

(i) $\mathbf{OC_Z} \in \mathscr{A}_{oc}$.

(ii) $\mathbf{Z} \leqq \mathbf{OC_Z}$.

(iii) *For each* $\mathbf{X} \in \mathscr{A}_{oc}$, $\mathbf{Z} \leqq \mathbf{X} \Rightarrow \mathbf{OC_Z} \leqq \mathbf{X}$.

Proof: (i) follows immediately from Proposition 5.6.

(ii) Holds, since $\mathbf{Z} \leqq \mathbf{C_Z}$ by Theorem 4.2, and $\mathbf{C_Z} \leqq \mathbf{O_{C_Z}}$ by Theorem 5.4.

(iii) Let $\mathbf{X} \in \mathscr{A}_{oc}$. Then if $\mathbf{Z} \leqq \mathbf{X}$, we have $\mathbf{C_Z} \leqq \mathbf{X}$ by Theorem 4.2, since $\mathbf{X} \in \mathscr{A}_c$. So $\mathbf{O_{C_Z}} \leqq \mathbf{X}$ by Theorem 5.4, since $\mathbf{X} \in \mathscr{A}_o$. Thus $\mathbf{OC_Z} \leqq \mathbf{X}$. ■

COROLLARY 6.2: *The control problem with partial observation has a solution if and only if*

$$\mathbf{Z} = \mathbf{D} \times \mathbf{OC_Z}. \quad ■$$

6.2 Distributed Control

We now have the tools to solve the problem of distributing the control among several controllers that can control or observe different sets of events. The problem, called the distributed control problem with global specification, can be stated as:

Let $\mathscr{A}_{oc_i}$ be a finite family of classes of observation and control automata with respect to Σ_{o_i} and Σ_{c_i}. Given a process $\mathbf{D}$, *and a specification* $\mathbf{Z}$ *such that* $\mathbf{Z} \leqq \mathbf{D}$, *do there exist automata* $\mathbf{X}_i \in \mathscr{A}_{oc_i}$ *such that*

$$\mathbf{Z} = \mathbf{D} \times \Pi\, \mathbf{X}_i?$$

To solve this problem, we simply apply to $\mathbf{Z}$ the construction of the preceding section for each class $\mathscr{A}_{oc_i}$, yielding a family of mappings $\mathbf{OC}_{\mathbf{Z}_i}$.

THEOREM 6.3: *The distributed control problem with global specification has a solution if and only if*

$$\mathbf{Z} = \mathbf{D} \times \Pi\, \mathbf{OC}_{\mathbf{Z}_i}.$$

Proof: Since $\mathbf{Z} \leqq \mathbf{D}$, and $\mathbf{Z} \leqq \mathbf{OC}_{\mathbf{Z}_i}$ for each i, we have that

$$\mathbf{Z} \leqq \mathbf{D} \times \Pi\, \mathbf{OC}_{\mathbf{Z}_i}.$$

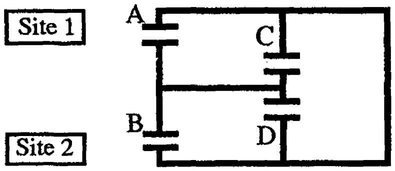
On the other hand, if a solution exists, we have that $\mathbf{Z} \leqq \mathbf{X}_i$ for each i, thus $\mathbf{OC}_{\mathbf{Z}_i} \leqq \mathbf{X}_i$. Multiplying those together, along with $\mathbf{D}$, we get

$$\mathbf{D} \times \Pi\, \mathbf{OC}_{\mathbf{Z}_i} \leqq \mathbf{D} \times \Pi\, \mathbf{X}_i = \mathbf{Z}. \quad \blacksquare$$

Notes: Necessary and sufficient conditions for the solution of the distributed control problem with global specification appear in [5], where they are expressed in terms of language properties.

7. AN EXAMPLE

Suppose three rooms are connected with doors as indicated in following diagram:



Each door, labelled A, B, C or D can allow objects to move in either direction. Observable and/or controllable events will be, for example, of the form A_{in} or C_{out} meaning that an object just passed through a door in a given direction (fixing arbitrarily the "in" direction as left to right). We are to construct two controllers, one at Site 1 and one at Site 2, that should prevent two objects from being in the same room simultaneously. The controller at Site 1 can observe events at doors A, B and C, and the controller at Site 2 events at doors A, B and D. Six events are controllable, i.e. all except A_{out} and B_{out}. Is it possible to distribute these controllable events among the two controllers such that they can enforce the constraint, while allowing maximum freedom to objects?

The following automaton represents the specification **Z**. Each state displays explicitly the position of objects and we can take the "empty" state as initial. (For clarity, we labeled only the "in" arrows since all events are reversible.)

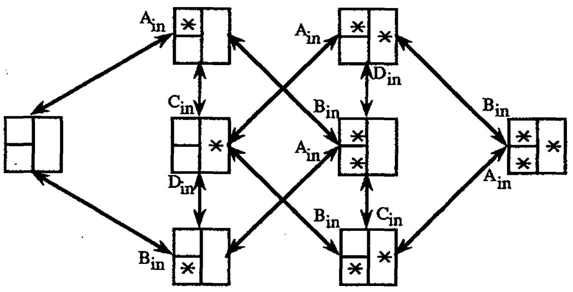


Figure 1. – The specification Z.

We can easily construct the observation automata $\mathbf{O}_{\mathbf{Z}_1}$ and $\mathbf{O}_{\mathbf{Z}_2}$ (Figure 2) giving the partial view of the specification **Z** from Site 1 and Site 2 respectively.

If these two automata function together, it is quite easy to convince oneself that the product has only eight accessible states and there is a natural correspondence with the eight original states of the specification. Indeed, states of the product can be thought of as non-empty intersections of states of each of the observers.

If the number of objects is unbounded, the whole process has an infinite number of states. But since we are interested only in enforcing the specification, we can model the whole process **D** by adding deadlock states (the "alarm" states) to the specification, and directing to them possible events that violate the constraint. In Figure 3, the round states represent these states.

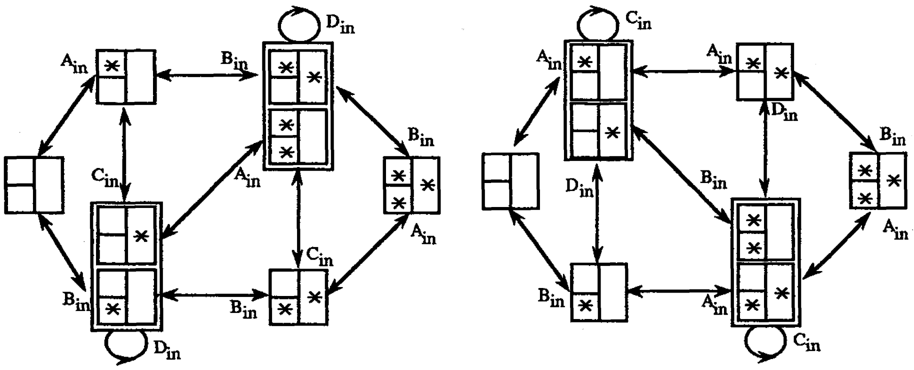


Figure 2. — Observation automata O_{Z_1} for Site 1 and O_{Z_2} for Site 2.

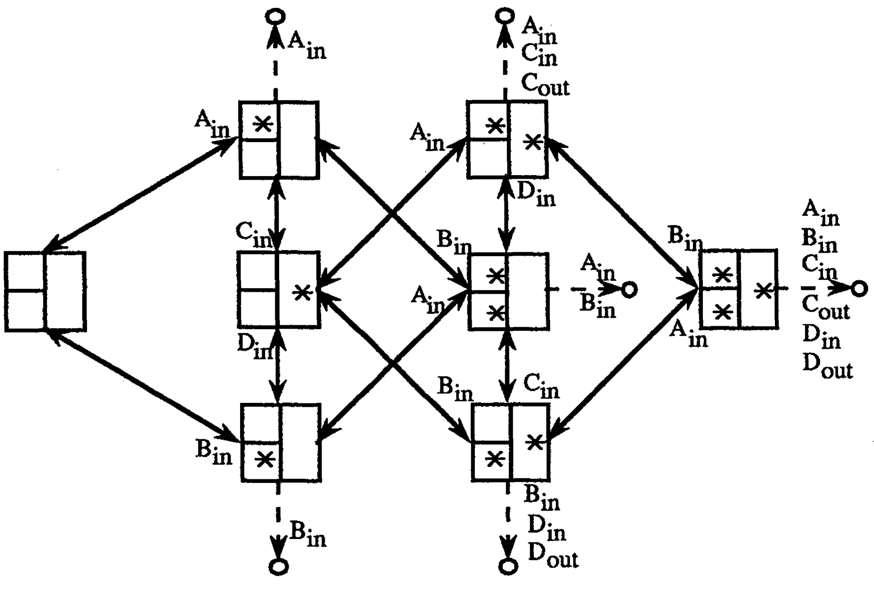


Figure 3. — The process D.

As for control strategies, suppose first that we choose the obvious distribution of controllable events:

Site 1 : A_{in}, C_{in}, C_{out}
Site 2 : B_{in}, D_{in}, D_{out}

Controllable events distribution (1)

Figure 4 shows parts of the automata $\mathbf{OC}_{\mathbf{Z}_1}$ and $\mathbf{OC}_{\mathbf{Z}_2}$ that result from this choice (with the state **u** resulting from the 'control' construction). These fragments shows explicitly that the sequence $A_{in}\,B_{in}\,C_{in}\,D_{in}$ is defined in both automata, and since it is also defined in **D**, it will be defined in the product $\mathbf{D}\times\mathbf{OC}_{\mathbf{Z}_1}\times\mathbf{OC}_{\mathbf{Z}_2}$. But the sequence $A_{in}\,B_{in}\,C_{in}\,D_{in}$ is not in the specification **Z**, so there is no solution with this choice of strategy.

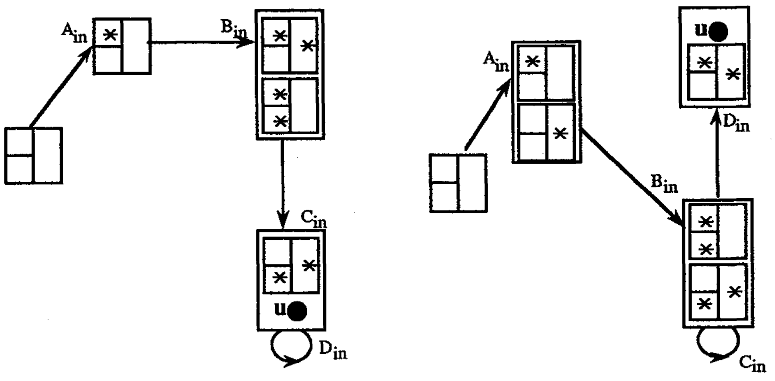


Figure 4. – **Partial automata $\mathbf{OC}_{\mathbf{Z}_1}$ and $\mathbf{OC}_{\mathbf{Z}_2}$ for distribution** (1).

An alternate strategy is the following:

Site 1 : A_{in}, C_{out}, D_{in}
Site 2 : B_{in}, D_{out}, C_{in}

Controllable events distribution (2)

This distribution, curiously, assigns control of events that are not observable from the site. For example, Site 1 will control event D_{in}, which is unobservable from the site. For this choice, a solution exists. It can be obtained simply from the two observers of Figure 2, by disabling, in a given state, all the events that are undefined but controllable from the site. We obtain the two controllers of Figure 5.

In order to prove that this is a solution, we have to show that the product of these two automata with the process **D** is contained in the specification **Z**. It suffice to verify that all events that could lead to an alarm state are disabled by one of the controllers. There are just 6 states to check. For

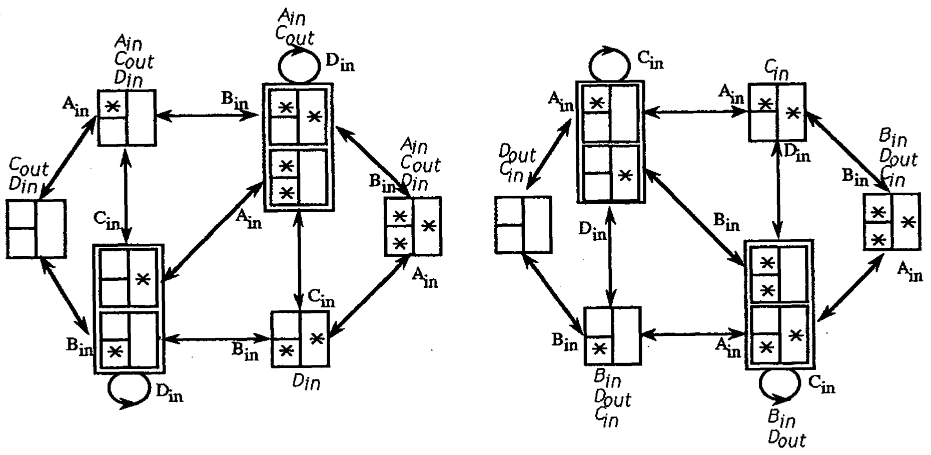
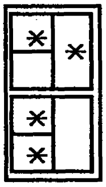
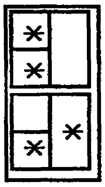
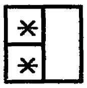


Figure 5. – Control strategy for Site 1 and Site 2 (disabled events in italics).

example, if the two controllers are in states:

 and  then the process is in state 

The events A_{in} and B_{in} must be disabled (as one can seen in Figure 3). But the first controller disables events A_{in} and C_{out}, and the second disables events B_{in} and D_{out}. The verification is just as elementary for the 5 other states that could lead to an alarm state.

8. CONCLUDING REMARKS

In this paper, we obtained new sets of necessary and sufficient conditions for the solution of problems in the theory of control of discrete event systems. Lemma 3.1 provides a unifying setting for design problems, which we applied successfully to classical control and observation problems. It also suggests that other classes of automata could be considered in order to solve other kinds of design problems.

Further work includes investigation of the *range* problem. In [5] the following problem is solved. With the notations of section 6.2, given $\mathbf{Z} \leq \mathbf{Z}' \leq \mathbf{D}$ where $\mathbf{Z}$ and $\mathbf{Z}'$ are respectively a minimal and a maximal specification, does there exists controllers $\mathbf{X}_i \in \mathscr{A}_{oc_i}$ such that

$$\mathbf{Z} \leqq \overline{\mathbf{D} \times \Pi\, \mathbf{X}_i} \leqq \mathbf{Z}'?$$

A necessary and sufficient condition for the existence of a solution is easily seen to be

$$\overline{\mathbf{D} \times \Pi\, \mathbf{OC}_{z_i}} \leqq \mathbf{Z}'.$$

Furthermore, if a solution exists, then the product $\mathbf{D} \times \Pi\, \mathbf{X}_i$ where $\mathbf{X}_i = \overline{\mathbf{OC}_{z_i}}$ is always non-blocking (simply because anything defined in this product must be marked by $\mathbf{Z}'$, thus by $\mathbf{D}$).

A more difficult problem consists in asking for the existence of controllers such that

$$\mathbf{Z} \leqq \mathbf{D} \times \Pi\, \mathbf{X}_i \leqq \mathbf{Z}'?$$

Again, a necessary and sufficient condition is that $\mathbf{D} \times \Pi\, \mathbf{OC}_{z_i} \leqq \mathbf{Z}'$, but there is no simple way to ensure that the product is non-blocking.

We also intend to explore complexity issues, especially as regards the observation construction, which can grow exponentially. These issues are crucial in the commutation and telecommunication problems around which we developed this model.

REFERENCES

1. A. Arnold, Systèmes de transitions finis et sémantique des processus communicants, *T.S.I.*, *9*,, n° 3, 1990, pp. 193-216.
2. A. Arnold and M. Nivat, *Comportements de processus*, In Colloque AFCET, Les mathématiques de l'informatique, 1982, pp. 35-68.
3. P. Ramadge and W. Wonham, Supervisory Control of a Class of Discrete Event Processes, *SIAM J. Control and Optimization*, *25*, No. 1, 1987, pp. 206-230.
4. P. Ramadge and W. Wonham, The Control of Discrete Event Systems, *Proceedings of the IEEE*, *77*, No. 1, 1989, pp. 81-98.
5. K. Rudie and W. Wonham, Think Globally, Act Locally: Decentralized Supervisory Control, *IEEE Transactions on Automatic Control*, *37*, No. 11, 1992, pp. 1692-1708.
6. F. Lin and M. Wonham, On Observability of Discrete-Event Systems, *Information Science*, *44*, 1988, pp. 173-198.