Discrete Event Dynamic Systems: Theory and Applications 1, 61–92 (1991)


# On Tolerable and Desirable Behaviors in Supervisory Control of Discrete Event Systems*

STÉPHANE LAFORTUNE
*Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122*

FENG LIN
*Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202*

*Received March 26, 1990; Revised October 8, 1990*

**Abstract.** We formulate and solve a new supervisory control problem for discrete event systems. The objective is to design a logical controller—or supervisor—such that the discrete event system satisfies a given set of requirements that involve event ordering. The controller must deal with a limited amount of controllability in the form of uncontrollable events. Our problem formulation considers that the requirements for the behavior (i.e., set of traces) of the controlled system are specified in terms of a "desired" behavior and a larger "tolerated" behavior. Due to the uncontrollable events, one may wish to tolerate behavior that sometimes exceeds the ideal desired behavior if overall this results in achieving more of the desired behavior. The general solution of our problem is completely characterized. The nonblocking solution is also analyzed in detail. This solution requires the study of a new class of controllable languages. Several results are proved about this class of languages. Algorithms to compute certain languages of interest within this class are also presented.

**Key Words:** discrete event systems, supervisory control, nonblocking controllers, controllable languages

## 1. Introduction

### *1.1. Background*

The modeling of a discrete event system (DES) can be done at various complementary levels of abstraction. At the logical level, one is only concerned with the logical order of the events in the traces—the system's trajectories—and not with the time elapsed between two consecutive events. At the temporal level, time information is attached to events in the traces. At the stochastic level, the times of occurrence of the events are random variables. These levels also correspond to the usual methodology of analysis of a particular DES. In the first step, one undertakes a qualitative analysis of the system's properties at the logical level. Typically, this leads to the proposal of various logical controllers for the system in order to satisfy all the qualitative specifications that involve event ordering. Each logical controller leads to a much smaller set of traces for the closed-loop system for which one then undertakes a quantitative analysis at the temporal and stochastic levels.

*Research supported in part by the National Science Foundation under grants ECS-8707671, ECS-9057967, and ECS-9008947.

In this paper we consider the design of a logical controller—or supervisor—for a given DES in order to satisfy a given set of qualitative specifications. This is usually referred to as the problem of "supervisory control." An important issue in that context is how to deal with the possibly limited amount of controllability available to the controller. A theory for the supervisory control of DESs that specifically addresses such controllability issues has been developed over the last 10 years by several researchers, most notably P.J. Ramadge, W.M. Wonham, and F. Lin (see the survey paper by Ramadge and Wonham [1989]). We will refer to this body of work as supervisory control theory (SCT) in the remainder of this paper. A brief review of the notation and main definitions of SCT is given in the appendix.

In the paradigm of SCT, only a proper subset $\Sigma_c$ of the set of events $\Sigma$ is controllable, meaning that only the events in $\Sigma_c$ can be disabled by the controller. The set $\Sigma_u := \Sigma - \Sigma_c$ is the set of uncontrollable events. There are essentially two reasons for an event to be uncontrollable. First, the event may be inherently uncontrollable because it models an unpreventable failure of the system; standard examples are "machine breakdown" in manufacturing, "packet lost" in networks, and so forth. Second, the event may be uncontrollable because it would be impractical or undesirable to make it controllable (i.e., to allow for its disablement) in an implementation of the control system. For example, one may not wish to allow for the disablement of the bottleneck machine in a manufacturing system; the events modeling the dynamical behavior of that machine would thus be uncontrollable. In a real-time computer system, operations that correspond to the execution of a task with a "hard" time constraint should never be disabled. From a different perspective, it may be undesirable to make an event controllable because of hardware limitations or costs.

Consider a DES $G$ whose behavior is specified in terms of the two (nonempty) languages $L_m(G)$ and $L(G)$; $L(G)$ is the set of all traces that the uncontrolled system can generate, while $L_m(G)$ is the subset of marked traces. As is standard in SCT we require the following condition:

(H0) $\overline{L_m(G)} = L(G) \subseteq \Sigma^*$.

In their seminal paper, Ramadge and Wonham [1987], formulate the problem of supervisory control (SCP hereafter) in terms of two languages that are subsets of $L_m(G)$. The first language, denoted $L_{\min}$, corresponds to the minimally acceptable behavior, and the second one, denoted $L_{am}$, corresponds to the legal or admissible behavior. The goal in SCP is to synthesize a controller such that the behavior of the controlled system, characterized by the two languages $L_m(S/G)$ and $L(S/G)$, satisfies the correctness condition

$$\overline{L_{\min}} \subseteq L(S/G) \subseteq \overline{L_{am}}$$

and the "nonblocking" condition

$$\overline{L_m(S/G)} = L(S/G).$$

The correctness condition is obvious. The nonblocking condition requires that the controlled DES never allows a trace that, albeit legal, cannot be extended to any member of the set of marked admissible traces $L_{am}$. Intuitively, the controlled system should not "deadlock."

More recently, Chen and Lafortune [1991] have considered a generalized version of SCP where the nonblocking condition is relaxed. In their formulation, there is no minimally acceptable behavior, but rather an admissible language $L_{am}$ for the marked traces and an admissible language $L_a$ for all traces (marked or unmarked); $L_a = \overline{L_a} \subseteq L(G)$. It is required that $L_a \cap L_m(G) = L_{am}$, but in general $\overline{L_{am}} \subseteq L_a$. The control problem, termed *supervisory control problem with blocking* (SCPB), is to synthesize $S$ such that

$$L(S/G) \subseteq L_a.$$

The performance of $S$ is evaluated in terms of the trade-off between its *satisficing measure*, defined as the set $L_m(S/G) \cap L_{am} = L_m(S/G)$, and its *blocking measure*, defined as the set $L(S/G) - \overline{L_m(S/G)}$. The satisficing measure indicates how much of the admissible language $L_{am}$ is allowed under control, while the blocking measure indicates how often the execution blocks due to the impossibility of continuing the execution within $L_{am}$. The motivation behind the formulation of SCPB is that a nonblocking controller may be too conservative in the sense that it must prevent all uncontrollable events that lead to blocking, a strategy that may considerably constrain the behavior of the system. SCPB does not have a unique solution, but rather a set of solutions that can be compared in terms of their respective satisficing and blocking measures.

### *1.2. Our Approach*

We take a slightly different and somewhat more general approach than the work mentioned in the previous section. We assume that the qualitative specifications for the design of the logical controller are given in terms of two languages $B_1$ and $B_2$, with $B_1 \subseteq B_2$, where $B_1$ represents the "desired" behavior and $B_2$ represnts the "tolerated" behavior. We believe that in several applications, a problem formulation in terms of desired and tolerated behaviors is more appropriate than one in terms of minimally acceptable and admissible behaviors. $B_1$ represents the ideal behavior of the controlled system, the type of behavior that one would require if all the events were controllable. $B_2$ is determined from the collection of all "hard" requirements that are imposed on the controlled behavior. On the other hand, there may be "soft" requirements in $B_1$ that one would like to satisfy but could relax if they prove too restrictive due to the limited amount of controllability. This is the excess of the desired behavior that is tolerated. In this sense, one would tolerate behavior that would violate some of the soft requirements, if that would help in achieving more of the desired behavior.

For example, in a manufacturing system, there may be a set of desired maximum buffer occupancies and a larger set of tolerated maximum buffer occupancies if extra storage space is available nearby. In network protocols, it may be desired to never retransmit, but tolerated to retransmit a certain number of times. Finally, we mention that the recent work of Chen and Lafortune on SCPB [1991] is also relevant to our problem formulation if one selects $B_1 = L_{am}$ and $B_2 = L_a$.

The control problem that we consider is as follows. We are given a DES satisfying (H0). Our specifications are given in terms of the two languages $B_1$ and $B_2$. Since the desired

behavior is more naturally expressed in terms of marked traces, we assume that $B_1 \subseteq L_m(G)$. On the other hand, a prefix of a tolerable trace should also be tolerable. Hence we assume that the tolerated behavior is a closed sublanguage of $L(G)$. In summary, our assumptions are as follows:

(H1) $B_1 \subseteq L_m(G)$ and $B_1 = \overline{B_1} \cap L_m(G)$ (i.e., $B_1$ is $L_m(G)$-closed).
(H2) $B_2 \subseteq L(G)$ and $B_2 = \overline{B_2}$.
(H3) $\overline{B_1} \subseteq B_2$.

Informally, the objective is to design a controller such that the controlled system

1. Never goes beyond the tolerated behavior
2. Achieves as much as possible of the desired behavior under 1
3. Achieves 2 with the smallest possible solution.

All comparisons are with respect to (w.r.t.) set inclusion. Conditions 1 and 2 are natural, while condition 3 is to ensure that as few as possible of the soft requirements are relaxed. We call this control problem *supervisory control problem with tolerance* (SCPT). It may or may not be required that the controller be nonblocking; both cases will be studied.

A precise formulation of SCPT is given in Section 2. But one can already observe that the trade-off inherent in SCPB is absent in SCPT because condition 2 is enforced before condition 3. In contrast to SCPB, one can therefore talk of *the* solution of SCPT. It should also be pointed out that it is not required that $B_2 \cap L_m(G) = B_1$. In this sense SCPT is more general than SCPB, where as we said earlier it is assumed that $L_a \cap L_m(G) = L_{am}$.

### *1.3. Specifying Desired and Tolerated Behaviors*

$B_1$ and $B_2$ are in effect the design parameters associated with the control problem. Although their precise form is dependent on the particular problem considered, some general cases are worthy of mention.

***Blocking and Recovery.*** There are several situations in computer systems (e.g., operating systems, database systems) where deadlock detection and recovery schemes perform better than deadlock prevention schemes. (Recovery refers to the process of resolving a deadlock.) When recovery is not explicitly modeled in the uncontrolled discrete event process—and this is usually desirable for the sake of simplicty—deadlock corresponds to blocking in the framework of SCT (see, e.g., [Lafortune 1988]). Typically, there are some deadlock situations that can be viewed as "soft" in the sense that they are considered recoverable. Let $B_r$ be the (closed) set of traces corresponding to such situations. Then for a given $B_1$, one could take

$$B_2 = \overline{B_1} \cup B_r.$$

***Rare Uncontrollable Events.*** Certain uncontrollable events may have a very low probability of occurrence. If some of these events lead to undesirable, yet tolerable, behavior, then one may be willing to take a chance with them. Let $\Sigma_i \subseteq \Sigma_u$ be the set of "rare" and tolerable uncontrollable events. One could define

$$B_2 = \overline{B_1}\,\Sigma_i^* \cap L(G).$$

One may also wish to consider the more general case

$$B_2 = \overline{B_1}\,K \cap L(G),$$

where $K$ is a tolerated set of suffixes that go beyond the desired $B_1$.

***Language Enlargement.*** In the same vein, one could be more formal than above and precisely quantify the probabilities of events in given states. This is the approach first presented by Lin [1989]. This leads to the definition of an $\epsilon$-enlargement of a given language, where $\epsilon$ is the degree of tolerance for the enlargement. There are several ways to define $\epsilon$-enlargement; one possible general definition is

$$(E)_\epsilon := E \cup \{st \in L(G)\colon \text{Prob}(s|CD) < \epsilon\},$$

where $\text{Prob}(s|CD)$ is the probability that trace $s$ occurs under condition $CD$. For example, by defining $CD = (s' \in E \text{ has occurred})$, one could enlarge $E$ to include traces $st$ of the form $s'\sigma t$, with $\sigma \in \Sigma_u$, such that the first event $\sigma$ that takes the traces outside of $E$ occurs with probability less than $\epsilon$ in state $\delta(s', q_0)$. In our context, one could take

$$B_2 = (\overline{B_1})_\epsilon$$

for any appropriate definition of $\epsilon$-enlargement.

### *1.4. Organization and Contribution of Paper*

The previous sections have motivated the formulation of the new supervisory control problem SCPT. The remainder of this paper is devoted to the solution of SCPT. The general case is considered in Section 2, the nonblocking case in Section 3. While the general solution is straightforward, the nonblocking solution poses technical difficulties, since an optimal solution may not exist. Several new results are developed for the study of that case.

Nonblocking solutions of SCPT depend upon the following operation on languages:

> Given $L \subseteq M \subseteq L_m(G)$, find a minimal superlanguage of $L$ that is (i) contained in $M$, (ii) controllable, and (iii) $M$-closed.

This operation is studied in detail in Section 4. The main results that we prove about it are: (i) there is a nonempty set of minimal controllable and $M$-closed superlanguages, but,

in general, no infimal superlanguage; (ii) some minimal superlanguages need not be regular; and (iii) there may be an infinite number of regular minimal superlanguages. In addition, we present an algorithm (on languages) that generates all minimal superlanguages. An implementation of the algorithm based on finite state machines is described in Section 5. That implementation generates a subset of the regular minimal superlanguages.

Section 6 concludes the paper, while some useful definitions and technical results are collected in appendix.

*Remark 1.1.* Notation: In the development that follows, we will often be doing operations on one language that will result in a set of languages. Let $L$ be the given language and let op denote the given operation. Then the set of resulting languages will be denoted by $\mathbf{L}^{\mathrm{op}}$, while $L^{\mathrm{op}}$ will denote any element of $\mathbf{L}^{\mathrm{op}}$.

## 2. General Solution of SCPT

### 2.1. Problem Formulation

Formally, we state SCPT as follows:

***Supervisory Control Problem with Tolerance (SCPT).*** Consider languages $L_m(G)$, $L(G)$, $B_1$, and $B_2$ satisfying (H0)–(H3). Synthesize a controller $S$ such that

1. $L(S/G) \subseteq B_2$.
2. $(\forall K \subseteq L(G))K = \bar{K} = K^{\uparrow} \subseteq B_2 \Rightarrow K \cap B_1 \subseteq L(S/G) \cap B_1$.
3. $(\forall K \subseteq L(G))[(K = \bar{K} = K^{\uparrow} \subseteq B_2) \wedge (K \cap B_1 = L(S/G) \cap B_1)] \Rightarrow L(S/G) \subseteq K$.

The first condition requires that the language generated by $S/G$ be tolerable. The second condition requires that the language generated by $S/G$ contain the largest possible part of the desired behavior under the first condition. The third condition requires that the language generated by $S/G$ be smallest under the first two conditions.

In this section, we do not restrict ourselves to nonblocking solutions. Hence, the controller synthesized may block; i.e., $\overline{L_m(S/G)} \neq L(S/G)$. Nonblocking solutions will be discussed in Section 3. It turns out that general solutions are much simpler than nonblocking solutions.

### 2.2. General Solution

The solution of SCPT is unique, as shown in the following theorem.

THEOREM 2.1. The unique solution of SCPT is given by

$$L(S/G) = (B_1 \cap B_2^{\uparrow})^{\downarrow}.$$

*Proof.* We prove that the three conditions in the problem statement are satisfied.

1. $L(S/G) = (B_1 \cap B_2^{\uparrow})^{\downarrow}$
$\Rightarrow L(S/G) \subseteq (B_2^{\uparrow})^{\downarrow}$ (since the operation $(\cdot)^{\downarrow}$ is monotonic)
$= B_2^{\uparrow}$ (since $B_2^{\uparrow}$ is closed and controllable)
$\subseteq B_2$.

2. $(\forall K \subseteq L(G))K = \bar{K} = K^{\uparrow} \subseteq B_2$
$\Rightarrow K \subseteq B_2^{\uparrow}$ (since $K$ is controllable)
$\Rightarrow K \cap B_1 \subseteq B_2^{\uparrow} \cap B_1$
$\Rightarrow K \cap B_1 \subseteq (B_2^{\uparrow} \cap B_1)^{\downarrow}$
$\Rightarrow K \cap B_1 \subseteq (B_2^{\uparrow} \cap B_1)^{\downarrow} \cap B_1$
$\Rightarrow K \cap B_1 \subseteq L(S/G) \cap B_1$.

3. $(\forall K \subseteq L(G))[(K = \bar{K} = K^{\uparrow} \subseteq B_2) \wedge (K \cap B_1 = L(S/G) \cap B_1)]$
$\Rightarrow K \supseteq K \cap B_1$
$= (B_1 \cap B_2^{\uparrow})^{\downarrow} \cap B_1$ (by hypothesis)
$\supseteq B_1 \cap B_2^{\uparrow} \cap B_1$
$= B_1 \cap B_2^{\uparrow}$.

Also, $K$ is closed and controllable. Therefore, by the minimality of $(B_1 \cap B_2^{\uparrow})^{\downarrow}$,

$$L(S/G) = (B_1 \cap B_2^{\uparrow})^{\downarrow} \subseteq K.$$

Since $(B_1 \cap B_2^{\uparrow})^{\downarrow}$ uniquely exists, the solution of SCPT is unique. Q.E.D.

For a closed language $L \subseteq L(G)$, a formula to calculate $L^{\uparrow}$ is [Brandt et al. 1991]

$$L^{\uparrow} = L - [(L(G) - L)/\Sigma_u^*]\Sigma^*,$$

where / denotes the quotient operation on languages. For a language $L \subseteq L_m(G)$, a formula to calculate $L^{\downarrow}$ is [Lafortune and Chen 1990; Lin and Wonham 1988]

$$L^{\downarrow} = \bar{L}\Sigma_u^* \cap L(G).$$

Combining these two formulas, we get a formula to calculate $(B_1 \cap B_2^{\uparrow})^{\downarrow}$:

$$(B_1 \cap B_2^{\uparrow})^{\downarrow} = \overline{(B_1 \cap B_2^{\uparrow})}\Sigma_u^* \cap L(G)$$

$$= \overline{B_1 \cap (B_2 - [(L(G) - B_2)/\Sigma_u^*]\Sigma^*)\Sigma_u^*} \cap L(G). \quad (1)$$

The following example illustrates the above results.

### 2.3. *Example*

Consider a system consisting of two parallel processes, $G = G_1 \parallel G_2$, as shown in Figure 1. Assume that $\alpha_i$ is controllable and $\beta_i$ is uncontrollable. The desired behavior is that $\beta_1$ and $\beta_2$ occur alternately, beginning with $\beta_1$. Therefore, $B_1$ is generated by the generator in Figure 2. The tolerated behavior is that after an occurrence of $\beta_1$, $\beta_1$ cannot occur again until $\beta_2$ occurs at least once. Therefore $B_2$ is generated by the generator in Figure 3.

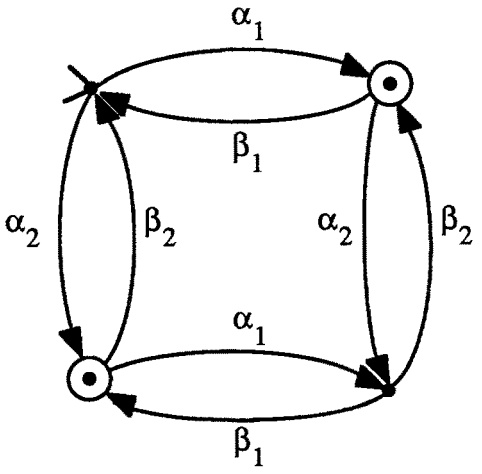


*Figure 1.*

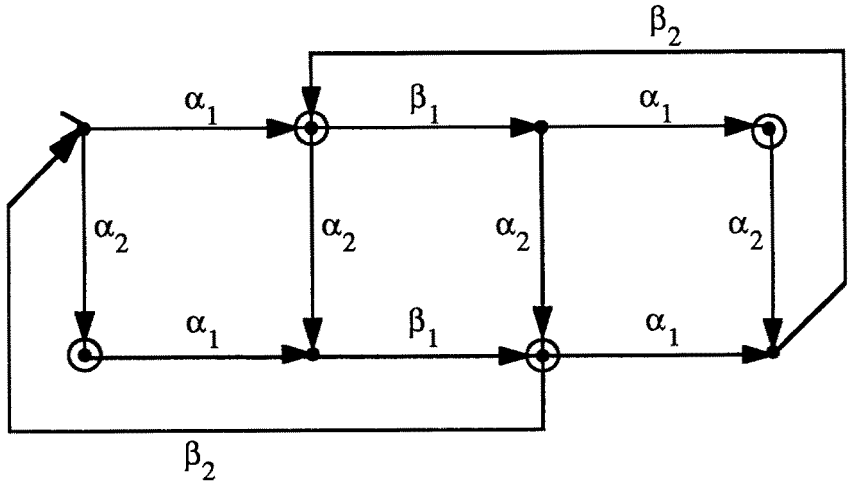


*Figure 2.*

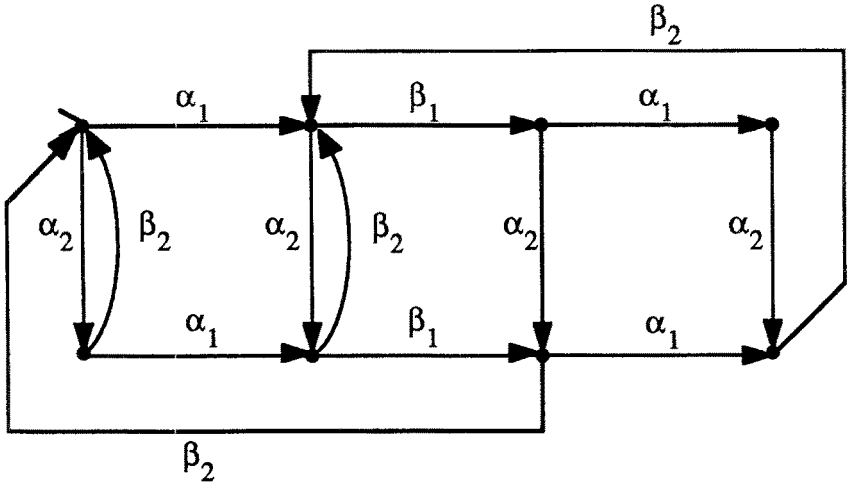


*Figure 3.* (All the states are marked.)

To find the solution of SCPT, $(B_1 \cap B_2^{\uparrow})^{\downarrow}$ is calculated; a generator of that language is shown in Figure 4. Observe that the controlled system $(B_1 \cap B_2^{\uparrow})^{\downarrow}$ may block because $\alpha_2\beta_2 \in L(S/G)$, but $\alpha_2\beta_2 \notin \overline{L(S/G) \cap L_m(G)}$. Nonblocking solutions of this problem are discussed in Section 3.3.

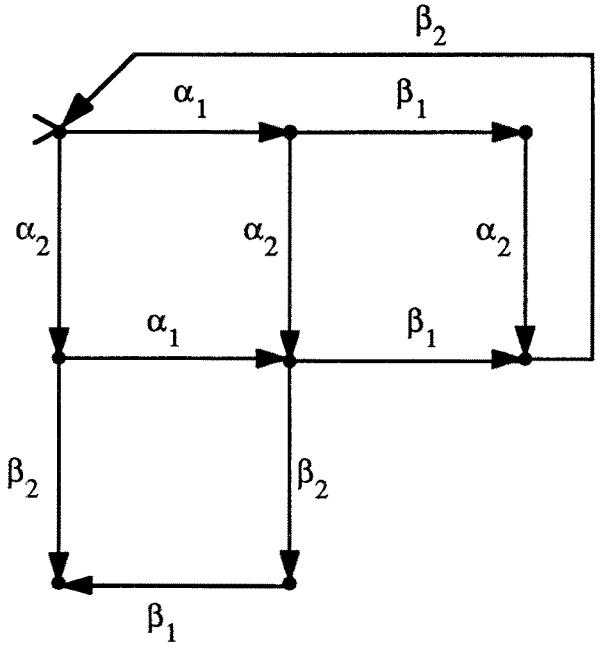


*Figure 4.* (All the states are marked.)

## 3. Nonblocking Solution of SCPT

### *3.1. Problem Formulation*

As shown in the above example, there is no guarantee that the general solution of the previous section is nonblocking. But in some applications, nonblocking solutions may be required. In this section, we discuss nonblocking solutions of SCPT. We first need to introduce a definition and a new assumption.

Let us define a language to be *livelock-free* if continuations of any trace of the language cannot remain unmarked for arbitrarily long suffixes in $\Sigma^*$. Formally, we have

DEFINITION 3.1. A language $L \subseteq \Sigma^*$ is said to be *livelock-free* if

$$(\forall s \in \bar{L})(\exists n \in \mathbb{N})(\forall t \in \Sigma^*)|t| \geq n \wedge st \in \bar{L} \Rightarrow (\exists u \in \Sigma^*)(s \leq u \leq st \wedge u \in L),$$

where $\mathbb{N}$ is the set of natural numbers, $|t|$ is the length of $t$, and $s \leq u$ denotes that $s$ is a prefix of $u$.

If a language $L$ is regular, then it will be livelock-free iff each directed cycle in the directed graph representaion of any finite-state generator of $L$ touches at least one marked state. Indeed, any $n \geq \|L\|$ will work in Definition 3.1.

As will become apparent in Section 4, in order to guarantee the existence of "interesting" nonblocking solutions of SCPT (cf. Theorem 4.2), we have to introduce further assumptions. These assumptions involve regularity and livelock-free conditions. More precisely, we add the following condition to our list of hypotheses:

(H4) $B_1$ and $L_m(G)$ are regular. $L_m(G)$ is livelock-free.

[Observe that it is not necessary for $B_1$ to be regular. Regularity of $B_1$ need only be introduced when one is interested in finite-step computations (see Section 5).]

***Supervisory Control Problem with Tolerance–Nonblocking Case (SCPT-NB).*** Consider languages $L_m(G)$, $L(G)$, $B_1$, and $B_2$ satisfying (H0)–(H4). Synthesize a nonblocking controller $S$ such that

1. $\overline{L_m(S/G)} \subseteq B_2$.
2. $(\forall K \subseteq L_m(G))[(K = \bar{K} \cap L_m(G) = K^{\uparrow}) \wedge (\bar{K} \subseteq B_2)] \Rightarrow K \cap B_1 \subseteq L_m(S/G) \cap B_1$.
3. $(\forall K \subseteq L_m(G))[(K = \bar{K} \cap L_m(G) = K^{\uparrow}) \wedge (\bar{K} \subseteq B_2) \wedge (K \cap B_1 = L_m(S/G) \cap B_1)]$
   $\Rightarrow K \not\subset L_m(S/G)$.

The first condition is to ensure that the language generated by the nonblocking controlled process $S/G$ is tolerable. The second condition requires that this language contain the largest possible part of the desired behavior under the first condition. The third condition says that in addition to satisfying the first two conditions, a solution should also be minimal with respect to set inclusion.

Let

$$B_{2m} := B_2 \cap L_m(G), \tag{2}$$

$$B_{1\max} := B_1 \cap B_{2m}^{\uparrow}. \tag{3}$$

The following properties of $B_{2m}^{\uparrow}$ and $B_{1\max}$ will be used.

LEMMA 3.1.

i) $B_{2m}^{\uparrow}$ is $L_m(G)$-closed.
ii) $B_{2m}^{\uparrow}$ is livelock-free.
iii) $B_{1\max}$ is $L_m(G)$-closed.
iv) $B_{1\max}$ is $B_{2m}^{\uparrow}$-closed.
v) $(B_{1\max})^{\downarrow} \cap L_m(G) = (B_{1\max})^{\downarrow} \cap B_{2m}^{\uparrow}$.

*Proof.* (i) $B_{2m}$ is $L_m(G)$-closed by definition and Lemma B.1. Thus $B_{2m}^{\uparrow}$ is $L_m(G)$-closed from Proposition 6.1 in [Wonham and Ramadge 1988].

(ii) The result follows from (i) and Lemma B.3.

(iii) It suffices to show that $\overline{B_{1\max}} \cap L_m(G) \subseteq B_{1\max}$. But

$$\begin{aligned}
\overline{B_{1\max}} \cap L_m(G) &= \overline{B_1 \cap B_{2m}^{\uparrow}} \cap L_m(G) \\
&\subseteq \overline{B_1} \cap \overline{B_{2m}^{\uparrow}} \cap L_m(G) \\
&= (\overline{B_1} \cap L_m(G)) \cap (\overline{B_{2m}^{\uparrow}} \cap L_m(G)) \\
&= B_1 \cap B_{2m}^{\uparrow} \\
&=: B_{1\max},
\end{aligned}$$

where the next-to-last equality follows by (H1) and (i).

(iv) Again, it suffices to show that $\overline{B_{1\max}} \cap B_{2m}^{\uparrow} \subseteq B_{1\max}$.

$$\begin{aligned}
\overline{B_1 \cap B_{2m}^{\uparrow}} \cap B_{2m}^{\uparrow} &\subseteq \overline{B_1} \cap B_{2m}^{\uparrow} \cap B_{2m}^{\uparrow} \\
&= \overline{B_1} \cap B_{2m}^{\uparrow} \\
&= \overline{B_1} \cap B_{2m}^{\uparrow} \cap L_m(G) \\
&= B_1 \cap B_{2m}^{\uparrow} \\
&=: B_{1\max}
\end{aligned}$$

(v) The result follows by observing that

$$\begin{aligned}(B_{1\max})^{\downarrow} \cap L_m(G) &= (B_1 \cap B_{2m}^{\uparrow})^{\downarrow} \cap L_m(G) \\ &= (B_1 \cap \overline{B_{2m}^{\uparrow}} \cap L_m(G))^{\downarrow} \cap L_m(G) \\ &= (B_1 \cap \overline{B_{2m}^{\uparrow}})^{\downarrow} \cap L_m(G) \\ &\subseteq \overline{B_{2m}^{\uparrow}} \cap L_m(G) \\ &= B_{2m}^{\uparrow}. \qquad \text{Q.E.D.}\end{aligned}$$

From (2) and Lemma 3.1(i), the largest tolerable nonblocking solution is $L_m(S/G) = B_{2m}^{\uparrow}$. This solution achieves $B_{1\max}$ in the desired behavior. We would like to achieve the same $B_{1\max}$ but with a smaller nonblocking solution than $L_m(S/G) = B_{2m}^{\uparrow}$. From Theorem A.1(ii), the language marked by such a solution must be controllable and $L_m(G)$-closed. In order to find a solution of SCPT-NB, we thus need to introduce the concept of controllable and $L_m(G)$-closed superlanguages of $B_{1\max}$ that are contained in $B_{2m}^{\uparrow}$. Therefore, define

$$\begin{aligned}CM(B_{1\max}, B_{2m}^{\uparrow}, L_m(G)) := \{K \subseteq \Sigma^*: (B_{1\max} \subseteq K \subseteq B_{2m}^{\uparrow}) \\ \wedge (\bar{K}\Sigma_u \cap \overline{L_m(G)} \subseteq \bar{K}) \wedge (K = \bar{K} \cap L_m(G))\}. \qquad (4)\end{aligned}$$

The class of languages $CM(B_{1\max}, B_{2m}^{\uparrow}, L_m(G))$ is not closed under intersection. As will be shown in Section 4, the infimal element of $CM(B_{1\max}, B_{2m}^{\uparrow}, L_m(G))$ may not exist, but there exists at least one minimal element. We denote the set of minimal elements of $CM(B_{1\max}, B_{2m}^{\uparrow}, L_m(G))$ by $(\mathbf{B_{1max}})^{\mathbf{CM}}$ and a minimal element by $(B_{1\max})^{CM}$. The following results are immediate from (2)–(4) and Lemma 3.1(i).

LEMMA 3.2.

i) $(B_{1\max})^{CM} \subseteq B_{2m}^{\uparrow}$.
ii) $(B_{1\max})^{CM} \cap B_1 = B_{1\max}$.

The $(\cdot)^{CM}$ operation is studied in detail in Section 4. Lemmas 3.1 and 3.2 together with the existence of $(B_{1\max})^{CM}$, a minimal controllable and $L_m(G)$-closed superlanguage of $B_{1\max}$ (to be formally established in Theorem 4.2), are sufficient for our present purposes.

### *3.2. Nonblocking Solution*

The following theorem shows that SCPT-NB can be solved, but that it does not have a unique solution in general.

THEOREM 3.1. A solution of SCPT-NB is given by

$$L_m(S/G) = (B_{1\max})^{CM}.$$

*Proof.* Since $(B_{1\max})^{CM}$ is controllable and $L_m(G)$-closed, the solution $L_m(S/G) = (B_{1\max})^{CM}$ is nonblocking. We prove that the three conditions in the problem statement are satisfied.

1. $\overline{L_m(S/G)} = \overline{(B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM}}$

$$\Rightarrow \overline{L_m(S/G)} \subseteq \overline{(B_2 \cap L_m(G))^{\uparrow}} \quad \text{(by Lemma 3.2)}$$

$$\subseteq \overline{B_2 \cap L_m(G)}$$

$$\subseteq \overline{B_2}$$

$$= B_2.$$

2. $(\forall K \subseteq L_m(G))[(K = \bar{K} \cap L_m(G) = K^{\uparrow}) \wedge (\bar{K} \subseteq B_2)]$,

$$\bar{K} \subseteq B_2$$

$$\Rightarrow \bar{K} \cap L_m(G) \subseteq B_2 \cap L_m(G)$$

$$\Rightarrow \bar{K} \cap L_m(G) \subseteq (B_2 \cap L_m(G))^{\uparrow} \quad \text{(since } \bar{K} \cap L_m(G) \text{ is controllable)}$$

$$\Rightarrow \bar{K} \cap B_1 \subseteq (B_2 \cap L_m(G))^{\uparrow} \cap B_1$$

$$\Rightarrow \bar{K} \cap B_1 \subseteq (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM}$$

$$\Rightarrow \bar{K} \cap B_1 \subseteq (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM} \cap B_1$$

$$\Rightarrow \bar{K} \cap L_m(G) \cap B_1 \subseteq (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM} \cap B_1$$

$$\Rightarrow K \cap B_1 \subseteq (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM} \cap B_1 \quad \text{(since } K \text{ is } L_m(G)\text{-closed)}$$

$$\Rightarrow K \cap B_1 \subseteq L_m(S/G) \cap B_1.$$

3. $(\forall K \subseteq L_m(G))[(K = \bar{K} \cap L_m(G) = K^{\uparrow}) \wedge (\bar{K} \subseteq B_2) \wedge (K \cap B_1 = L_m(S/G) \cap B_1)]$,

$$\bar{K} \subseteq B_2$$

$$\Rightarrow \bar{K} \cap L_m(G) \subseteq B_2 \cap L_m(G)$$

$$\Rightarrow K \subseteq B_2 \cap L_m(G) \quad \text{(since } K \text{ is } L_m(G)\text{-closed)}$$

$$\Rightarrow K \subseteq (B_2 \cap L_m(G))^{\uparrow} \quad \text{(since } K \text{ is controllable)}$$

and

$$\begin{aligned} B_1 \cap (B_2 \cap L_m(G))^{\uparrow} &= B_1 \cap (B_2 \cap L_m(G))^{\uparrow} \cap B_1 \\ &\subseteq (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM} \cap B_1 \\ &= K \cap B_1 \qquad \text{(by hypothesis)} \\ &\subseteq K. \end{aligned}$$

These results show that $K \in CM(B_{1\max}, B_{2m}^{\uparrow}, L_m(G))$ since, by hypothesis, $K$ is controllable w.r.t. $L_m(G)$ and is $L_m(G)$-closed. Therefore, by minimality of $(B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM}$,

$$K \not\subset (B_1 \cap (B_2 \cap L_m(G))^{\uparrow})^{CM} = L_m(S/G). \qquad \text{Q.E.D.}$$

In view of Theorem 3.1 and Lemma 3.2(ii), it should now be clear to the reader why the language $B_1 \cap B_{2m}^{\uparrow}$ is denoted $B_{1\max}$. If the solution of SCPT is not required to be nonblocking, then $(B_1 \cap B_2^{\uparrow})^{\downarrow} \cap B_1$ is the maximum achievable part of the desired behavior $B_1$. On the other hand, when the nonblocking condition is enforced, the maximum achievable part of the desired behavior is reduced to $B_{1\max}$. Moreover, in contrast to the unique general solution $(B_1 \cap B_2^{\uparrow})^{\downarrow}$, several incomparable solutions of SCPT-NB can achieve $B_{1\max}$. These solutions are the elements of the set $(\mathbf{B_{1max}})^{\mathbf{CM}}$. Sections 4 and 5 are devoted to the computation of these solutions.

We conclude this section with some remarks on special cases. (Remark 4.1 is also a special case of interest.)

*Remark 3.1.* If $L_m(G)$ is closed, then a sublanguage of $L_m(G)$ is $L_m(G)$-closed iff it is closed. Therefore,

$$\begin{aligned} (B_{1\max})^{CM} &= (B_{1\max})^{\downarrow} \\ &= (B_1 \cap B_2^{\uparrow})^{\downarrow} \qquad \text{[by (H2)]} \end{aligned}$$

and SCPT and SCPT-NB have the same (unique) solution.

*Remark 3.2.* Of course, if $L(S/G) = (B_1 \cap B_2^{\uparrow})^{\downarrow}$ is nonblocking, then it is the unique solution of SCPT-NB. It can be shown that a (rather strong) sufficient condition for this to be true is that the two languages $L_m(G)$ and $\overline{B_1 \cap B_2^{\uparrow}}\Sigma_u^*$ are nonconflicting.

### *3.3. Example*

Take the same $B_1$, $B_2$, and $L_m(G)$ as in Section 2.3. We will find a nonblocking solution of SCPT. The first step is to calculate $B_{1\max} = B_1 \cap (B_2 \cap L_m(G))^{\uparrow}$, which is generated by the generator in Figure 5. The next step is to calculate $(B_{1\max})^{CM}$ (an algorithm will be given in Section 5), which is generated by the generator in Figure 6.

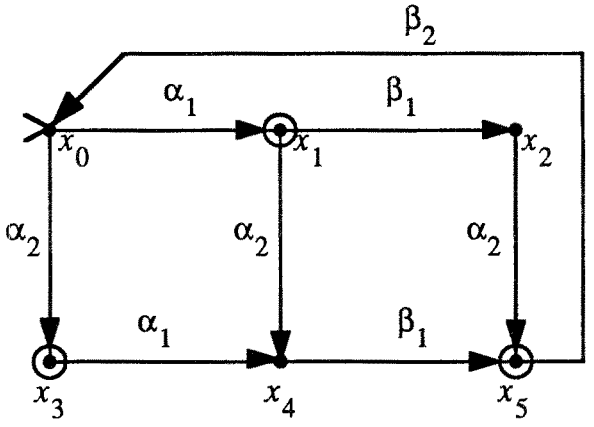


*Figure 5.*

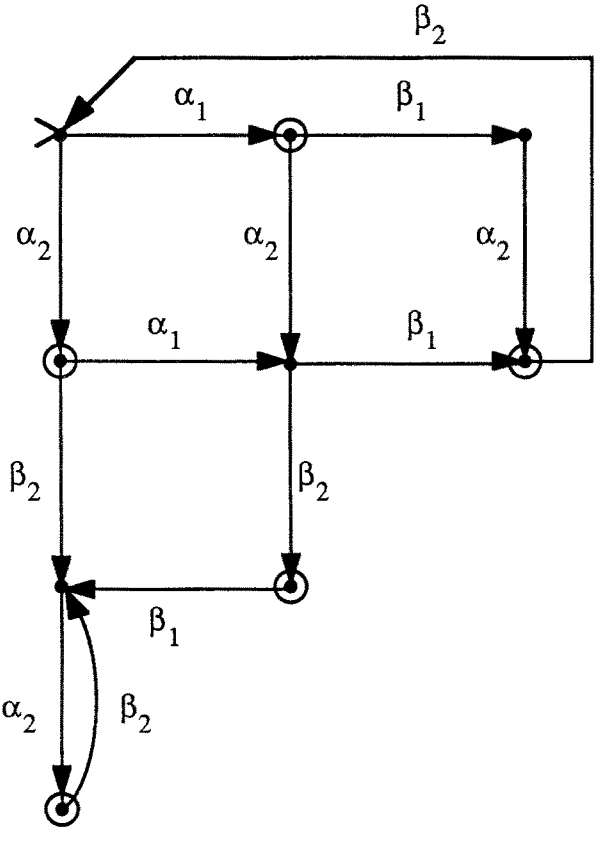


*Figure 6.*

## 4. Minimal Controllable and $L_m(G)$-Closed Superlanguages of a Given Language

### *4.1. Preliminaries*

In order to find minimal nonblocking solutions of SCPT, we must find minimal elements of the class $CM(B_{\text{lmax}}, B^{\uparrow}_{2m}, L_m(G))$. This is the class that we study in this section. For the sake of generality, we write

$$CM(B, B^{\uparrow}_{2m}, L_m(G)) := \{K \subseteq \Sigma^*: (B \subseteq K \subseteq B^{\uparrow}_{2m}) \wedge (\bar{K}\Sigma_u \cap \overline{L_m(G)} \subseteq \bar{K}) \wedge (K = \bar{K} \cap L_m(G))\} \quad (5)$$

when $B$ is a given language satisfying $B \subseteq B^{\uparrow}_{2m}$ and $B = \bar{B} \cap L_m(G)$. Observe that $B_{\text{lmax}}$ satisfies these conditions. It is convenient to define the new class of languages

$$CMR(B, B^{\uparrow}_{2m}) := \{K \subseteq \Sigma^*: (B \subseteq K \subseteq B^{\uparrow}_{2m}) \wedge (\bar{K}\Sigma_u \cap \overline{B^{\uparrow}_{2m}} \subseteq \bar{K}) \wedge (K = \bar{K} \cap B^{\uparrow}_{2m})\} \quad (6)$$

for a given language $B$ such that $B \subseteq B^{\uparrow}_{2m}$ and $B = \bar{B} \cap B^{\uparrow}_{2m}$. Once again, $B_{\text{lmax}}$ satisfies all conditions. Observe also that $B^{\uparrow}_{2m}$ is livelock-free by Lemma 3.1(ii).

THEOREM 4.1. Given $B \subseteq B^{\uparrow}_{2m}$ such that $B = \bar{B} \cap L_m(G)$ and $B = \bar{B} \cap B^{\uparrow}_{2m}$, $CM(B, B^{\uparrow}_{2m}, L_m(G)) = CMR(B, B^{\uparrow}_{2m})$.

*Proof.* (i) $CMR(B, B^{\uparrow}_{2m}) \subseteq CM(B, B^{\uparrow}_{2m}, L_m(G))$. First, observe that *CMR* and *CM* are defined within the same range $B \subseteq K \subseteq B^{\uparrow}_{2m}$. Second, if $K$ is controllable w.r.t. $B^{\uparrow}_{2m}$, then $K$ is also controllable w.r.t. $L_m(G)$ since $B^{\uparrow}_{2m}$ is controllable w.r.t. $L_m(G)$ (Lemma B.2). Third, $K = \bar{K} \cap B^{\uparrow}_{2m}$ and $B^{\uparrow}_{2m} = \overline{B^{\uparrow}_{2m}} \cap L_m(G)$ [Lemma 3.1(i)] imply that

$$K = \bar{K} \cap \overline{B^{\uparrow}_{2m}} \cap L_m(G) = \bar{K} \cap L_m(G).$$

(ii) $CM(B, B^{\uparrow}_{2m}, L_m(G)) \subseteq CMR(B, B^{\uparrow}_{2m})$. This time, $\bar{K}\Sigma_u \cap \overline{B^{\uparrow}_{2m}} \subseteq \bar{K}\Sigma_u \cap \overline{L_m(G)} \subseteq \bar{K}$, and thus $K$ is controllable w.r.t. $B^{\uparrow}_{2m}$. Similarly, $\bar{K} \cap B^{\uparrow}_{2m} \subseteq \bar{K} \cap L_m(G) = K$, and thus $K$ is also $B^{\uparrow}_{2m}$-closed. Q.E.D.

In view of Theorem 4.1 and for the sake of generality, in the remainder of this section we will consider the class

$$CMR(L, M) := \{K \subseteq \Sigma^*: (L \subseteq K \subseteq M) \wedge (\bar{K}\Sigma_u \cap \bar{M} \subseteq \bar{K}) \wedge (K = \bar{K} \cap M)\}, \quad (7)$$

where $L$ and $M$ are two given languages over $\Sigma$ satisfying the following properties: (i) $L \subseteq M$; (ii) $L = \bar{L} \cap M$; and (iii) $M$ is regular and livelock-free. Controllability will always be defined w.r.t. $M$ and w.r.t. the fixed $\Sigma_u \subseteq \Sigma$. All the results obtained will thus be directly applicable to the nonblocking solution of SCPT when $L = B_{\text{lmax}}$ and $M = B^{\uparrow}_{2m}$.

The following subclass of minimal elements of $CMR(L, M)$ plays a crucial role:

$$\mathbf{L}^{\mathbf{CM}} := \{K \in CMR(L, M): \forall K' \in CMR(L, M), K' \not\subset K\}. \tag{8}$$

THEOREM 4.2. $|\mathbf{L}^{\mathbf{CM}}| \geq 1$.

*Proof.* First, it is clear that $M \in CMR(L, M)$. We argue that $CMR(L, M)$ possesses at least one minimal element. By contradiction, suppose that $\mathbf{L}^{\mathbf{CM}}$ is empty. This means that there exists $(K_i, i > 0)$, $K_i \in CMR(L, M)$ for all $i$, such that

$$M \supset K_1 \supset K_2 \supset K_3 \supset \ldots \supseteq L.$$

In other words, an arbitrarily large number of traces can be removed from $M$. Since $M$ is regular, any finite-state generator of $M$ must therefore contain at least one cycle in its digraph representation. An arbitrarily large number of the traces that are removed must be traces that pass through one particular cycle. Let $t \in \Sigma^*$ be the subtrace corresponding to that cycle. Without loss of generality, we can write

$$K_i = K_{i-1} - \{su\},$$

$$K_{i+1} = K_i - \{stu\}.$$

But $K_i$ is $M$-closed. Thus if $su < v < stu$, then $v \notin M$. In other words, the cycle does not touch any marked states in the digraph generating $M$. This however implies that $M$ is not livelock-free and we have a contradiction.

Finally, $CMR(L, M)$ is not closed under set intersection and thus does not possess, in general, a unique infimal element. This is shown by the following example. Let $\Sigma_u = \{s\}$, $\Sigma = \{\alpha, s, t_1, t_2\}$, $M = \{\alpha, \alpha st_1, \alpha st_2\}$, and $L = \{\alpha\}$. Then $K_1 = \{\alpha, \alpha st_1\} \in CMR(L, M)$ and $K_2 = \{\alpha, \alpha st_2\} \in CMR(L, M)$. But $K_1 \cap K_2 = L$ and $L \notin CMR(L, M)$ since $L \neq L^{\uparrow}$. Consequently, $CMR(L, M)$ may possess more than one minimal element. Q.E.D.

COROLLARY 4.1. Given any $K \in CMR(L, M)$, there exists $L^{CM} \in \mathbf{L}^{\mathbf{CM}}$ such that $L^{CM} \subseteq K$.

*Proof.* By contradiction, suppose that no such $L^{CM}$ exists. Then we can construct a strictly decreasing sequence of sets $(K_i, i > 0)$, $K_i \in CMR(L, M)$ for all $i$ such that

$$M \supseteq K \supset K_1 \supset K_2 \supset \ldots \supseteq L.$$

But by the same argument as in the proof of Theorem 4.2, this contradicts the assumption that $M$ is livelock-free. Q.E.D.

*Example 4.1.* To illustrate that if $M$ is not livelock-free, then $\mathbf{L}^{\mathbf{CM}}$ could be empty, consider

$$M = \{a, abc(uc)^*f\} \quad \text{and} \quad L = \{a\}$$

with $\Sigma_u = \{b, u\}$. Then we can construct the sequence in $CMR(L, M)$:

$$K_0 = M,$$

$$K_i = M - \overline{\{abc(uc)^i f\}} \cup \{a\}.$$

For any $K \in CMR(L, M)$, there exists $i \in \mathbb{N}$ such that $K_i \subset K$, and consequently $\mathbf{L}^{\mathbf{CM}} = \emptyset$.

The following result provides a sufficient condition for $CMR(L, M)$ to possess a unique infimal element.

THEOREM 4.3. If $L^{\downarrow} \cap M$ is controllable, then $|\mathbf{L}^{\mathbf{CM}}| = 1$ and $L^{CM} = L^{\downarrow} \cap M$.

*Proof.* First, by hypothesis and by Lemma B.1, $L^{\downarrow} \cap M \in CMR(L, M)$. Next, suppose that there exists $K \in CMR(L, M)$ such that $L^{\downarrow} \cap M \not\subseteq K$. Then

$$L^{\downarrow} \cap M \not\subseteq \bar{K} \cap M \Rightarrow L^{\downarrow} \not\subseteq \bar{K} \Rightarrow L^{\downarrow} \not\subseteq K^{\downarrow} \Rightarrow L \not\subseteq K,$$

and the last statement contradicts $K \in CMR(L, M)$. Q.E.D.

*Remark 4.1.* When applied to SCPT-NB, Theorem 4.3 implies that if $\overline{(B_{1\max})}\Sigma_u^* \cap B_{2m}^{\uparrow}$ is controllable w.r.t. $B_{2m}^{\uparrow}$, then

$$(B_{1\max})^{CM} = \overline{(B_{1\max})}\Sigma_u^* \cap B_{2m}^{\uparrow},$$

and thus SCPT-NB has a unique solution. Translating back to controllability w.r.t. $L_m(G)$, Lemmas 3.1(v) and B.2 can be used to reformulate this result as follows: if $(B_{1\max})^{\downarrow} \cap L_m(G)$ is controllable, then

$$(B_{1\max})^{CM} = (B_{1\max})^{\downarrow} \cap L_m(G),$$

where $\downarrow$ is w.r.t. $L_m(G)$ this time.

In the following sections we discuss the properties of $\mathbf{L}^{\mathbf{CM}}$ and find algorithms to compute its elements. Recall that $L^{CM}$ denotes any element of $\mathbf{L}^{\mathbf{CM}}$.

### *4.2. Constructive Algorithm*

Our objective is to develop an algorithmic procedure to construct all the elements of $\mathbf{L}^{\mathbf{CM}}$. Intuitively, the task consists of extending the traces of $L$ by all possible suffixes of uncontrollable events in order to get a controllable language, and then extending again the traces in order that they all be part of $M$. The resulting language can be made $M$-closed, but the last extension may destroy controllability, and, consequently, the procedure must be reapplied. Observe that this last extension is nonunique, even if it is minimal (cf. proof of Theorem 4.2), which is the reason why the infimal controllable and $M$-closed superlanguage need not exist.

We have found that after the extension of the traces of $L$ with suffixes of uncontrollable events has been performed, it suffices to extend all those traces that are not in $M$ (and that have no continuation in the new $L$) by *one* controllable event, rather than by a longer suffix, before reapplying the procedure. This motivates the following definitions.

Let $K = \bar{K} \subseteq \bar{M}$. We wish to identify which traces of $K$ should be extended in order for the new language to be contained in $M$. These traces constitute the set $TBE(K)$ defined as

$$TBE(K) := \{s \in K - M\colon (\forall \sigma \in \Sigma)(s\sigma \notin \bar{K})\}. \tag{9}$$

Let $I$ be an index set for $TBE(K)$; i.e., $TBE(K) = \bigcup_{i \in I}\{s_i\}$. For each $s_i \in TBE(K)$, define the set of events

$$\Sigma(s_i) := \{\sigma \in \Sigma\colon s_i\sigma \in \bar{M}\}. \tag{10}$$

We wish to choose one extension $s_i\sigma$, $\sigma \in \Sigma(s_i)$, for each $s_i \in TBE(K)$ and thus create the new set $K^e$. $K^e$ is called and *e-extension* of $K$. All the possible combinations of extensions of traces in $TBE(K)$ give rise to the class of languages $\mathbf{K^e}$. Formally, we form the cartesian product

$$\Sigma^I(K) := \bigtimes_{i \in I} \Sigma(s_i)$$

and define the composition operation $\circ$ between $TBE(K)$ and an element $\sigma^I$ of $\Sigma^I(K)$ as

$$TBE(K) \circ \sigma^I := \bigcup_{i \in I} \{s_i\sigma_i\colon s_i \in TBE(K) \wedge \sigma_i = \sigma^I|_i\}, \tag{11}$$

where $\sigma^I|_i$ denotes the $i$th component of $\sigma^I$. Finally, we let

$$\mathbf{K^e} := \{K^e \subseteq \bar{M}\colon K^e = K \cup (TBE(K) \circ \sigma^I) \text{ for some } \sigma^I \in \Sigma^I(K)\}. \tag{12}$$

Clearly, $K^e = \overline{K^e}$ for all $K^e \in \mathbf{K^e}$. Observe that each $K^e$ contains a single extended trace $s\sigma$ for each trace in $TBE(K)$.

DEFINITION 4.1. An *extension sequence* $(D_i, i \geq 0)$ is a monotonically increasing sequence of closed languages such that

$$D_0 = \bar{L} \quad \text{and} \quad D_{i+1} \in (\mathbf{D_i^{\downarrow}})^{\mathbf{e}} \quad \text{for all } i > 0.$$

The *derived extension sequence* of an extension sequence $(D_i, i \geq 0)$ is the sequence $(C_i, i \geq 0)$ such that $C_i := D_i \cap M$ for all $i \geq 0$.

For all extension sequences $(D_i, i \geq 0)$, we have that $D_i \subseteq \bar{M}$ for all $i \geq 0$ (by definition of the $\downarrow$ and $e$ operators). Thus each extension sequence converges to a limit $D_\infty := \lim_{i \to \infty} D_i$. Clearly, $D_\infty = \overline{D_\infty}$. The same is true for the corresponding derived extension

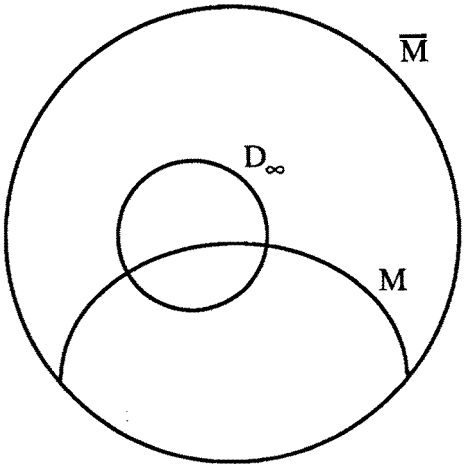


*Figure 7.*

sequence $(C_i, i \geq 0)$, whose limit is denoted $C_\infty := \lim_{i\to\infty} C_i = \lim_{i\to\infty} (D_i \cap M)$. Clearly, we have that $C_\infty = D_\infty \cap M$. We define the set of limits

$$\mathbf{C}_\infty := \{C_\infty : C_\infty = D_\infty \cap M \text{ for some extension sequence } (D_i, i \geq 0)\}. \quad (13)$$

The following result will be used in Section 4.3.

LEMMA 4.1. Consider the limit $D_\infty$ of an extension sequence $(D_i, i \geq 0)$ and its corresponding $C_\infty = D_\infty \cap M$. Then $D_\infty \subseteq \overline{C_\infty}$.

*Proof.* Since $D_\infty = \lim_{i\to\infty} D_i$, then it can be proved by contradiction that $TBE(D_\infty) = \emptyset$. Thus (i) all the traces in $D_\infty$ that have no continuation are in $D_\infty \cap M$, and, on the other hand, (ii) all the traces in $D_\infty - M$ have a continuation in $D_\infty \subseteq \bar{M}$. But such continuations cannot remain in $D_\infty - M$ for arbitrarily long suffixes because this would contradict the fact that $M$ is livelock-free. (See Figure 7.) Thus any trace in $D_\infty - M$ has a continuation in $D_\infty \cap M$. In conclusion, $D_\infty \subseteq \overline{D_\infty \cap M}$. Q.E.D.

### *4.3. Main Results*

The first important result of this section is that $\mathbf{C}_\infty = \mathbf{L}^{\mathbf{CM}}$. Three intermediate results need to be established first.

PROPOSITION 4.1. Given any $L^{CM} \in \mathbf{L}^{\mathbf{CM}}$, there exists an extension sequence $(D_i, i \geq 0)$ for which $C_\infty \subseteq L^{CM}$.

*Proof.* We will prove the result by induction on the extension sequence $(D_i, i \geq 0)$ to be constructed. First, $D_0 = \bar{L} \subseteq \overline{L^{CM}}$. Thus $C_0 = \bar{L} \cap M = L \subseteq L^{CM}$. Next, let $D_i \subseteq \overline{L^{CM}}$. This implies that

$$C_i := D_i \cap M \subseteq \overline{L^{CM}} \cap M = L^{CM}$$

by definition of $CMR(L, M)$. If we can construct $D_{i+1} \subseteq \overline{L^{CM}}$, then the proposition will be proved because the corresponding derived extension sequence $(C_i, i \geq 0)$ will satisfy $C_i \subseteq L^{CM}\ \forall i \geq 0$. But $D_i \subseteq \overline{L^{CM}}$ implies that $D_i^{\downarrow} \subseteq \overline{L^{CM}}$, since $L^{CM}$ is controllable by definition. Since $L^{CM} \subseteq M$ and $L^{CM}$ is $M$-closed, we can choose $\sigma_1^I \in \Sigma^I(D_i^{\downarrow})$ such that $TBE(D_i^{\downarrow}) \circ \sigma_1^I \subseteq \overline{L^{CM}} \subseteq \bar{M}$. This results in $D_{i+1} := D_i^{\downarrow} \cup (TBE(D_i^{\downarrow}) \circ \sigma_1^I) \subseteq \overline{L^{CM}}$. Q.E.D.

PROPOSITION 4.2. Given any $C_\infty \in \mathbf{C}_\infty$, there exists $L^{CM} \in \mathbf{L^{CM}}$ such that $L^{CM} \subseteq C_\infty$.

*Proof.* In view of Corollary 4.1, it suffices to show that the given $C_\infty \in CMR(L, M)$. Let $(D_i, i \geq 0)$ be an extension sequence that yields $C_\infty$—i.e., for which $D_\infty \cap M = C_\infty$. First, $\bar{L} \subseteq D_\infty \subseteq \bar{M}$ since $\bar{L} \subseteq D_i \subseteq \bar{M}$ for all $i \geq 0$. Thus

$$\bar{L} \cap M \subseteq D_\infty \cap M \subseteq \bar{M} \cap M \Rightarrow L \subseteq C_\infty \subseteq M.$$

Second, $C_\infty$ is $M$-closed by construction (cf. Lemma B.1). Hence, it remains to show that $C_\infty$ is controllable (w.r.t. $M$ and $\Sigma_u$). Since $D_\infty$ is the limit point of $(D_i, i \geq 0)$, the proof of its controllability is straightforward. But then $C_\infty = D_\infty \cap M$ is also controllable because

$$\begin{aligned}\overline{(D_\infty \cap M)}\Sigma_u \cap \bar{M} &\subseteq (D_\infty \cap \bar{M})\Sigma_u \cap \bar{M} \\ &\subseteq D_\infty \cap \bar{M} \\ &= D_\infty \\ &\subseteq \overline{D_\infty \cap M},\end{aligned}$$

where the last inequality follows from Lemma 4.1. Q.E.D.

PROPOSITION 4.3. For all distinct $C_\infty^1$ and $C_\infty^2$ in $\mathbf{C}_\infty$, $C_\infty^1 \not\subset C_\infty^2$.

*Proof.* Consider two distinct extension sequences $(D_i^1, i \geq 0)$ and $(D_i^2, i \geq 0)$. Definition 4.1 and (12) imply that there exists $i \geq 0$ such that $D_j^1 = D_j^2$, $0 \leq j \leq i$, but $D_{i+1}^1$ and $D_{i+1}^2$ are incomparable. More precisely, without loss of generality we can write

$$D_{i+1}^1 = (D_i^1)^{\downarrow} \cup \{s\sigma_1\} \cup K_1,$$

$$D_{i+1}^2 = (D_i^1)^{\downarrow} \cup \{s\sigma_2\} \cup K_2,$$

where $s\sigma_2 \notin K_1$ and $s\sigma_1 \notin K_2$. ($\cup$ denotes disjoint union.) Observe that $K_1$ or $K_2$ could be empty. Since the $e$-extension is done after the $\downarrow$ operation, $\sigma_1 \in \Sigma_c$ and $\sigma_2 \in \Sigma_c$. We can thus conclude that $s\sigma_1 \notin D^2_{i+k}$ and $s\sigma_2 \notin D^1_{i+k}$ $\forall k \geq 1$, since subsequent elements of the extension sequences only differ from previous ones by the addition of suffixes to certain traces (as a consequence of the $\downarrow$ and $e$-extension operations on closed languages). After step $i$, trace $s$ will no longer be a member of $TBE(D^{\downarrow}_{i+k})$, $k \geq 1$.

Consequently, $s\sigma_1 \in D^1_\infty$, $s\sigma_1 \notin D^2_\infty$ and $s\sigma_2 \in D^2_\infty$, $s\sigma_2 \notin D^1_\infty$; the two sets $D^1_\infty$ and $D^2_\infty$ are thus incomparable. By Lemma 4.1, there exist suffixes $t_1$ and $t_2$ in $\Sigma^*$ such that $s\sigma_1 t_1 \in C^1_\infty$ and $s\sigma_2 t_2 \in C^2_\infty$. On the other hand, $s\sigma_2 \notin D^1_\infty$ implies that $s\sigma_2 t \notin D^1_\infty$ for any $t \in \Sigma^*$, and thus $s\sigma_2 t \notin C^1_\infty$ either. Similarly, $s\sigma_1 t \notin C^2_\infty$ for any $t \in \Sigma^*$. We conclude from these observations that $C^1_\infty$ and $C^2{}_\infty$ are also incomparable. Q.E.D.

THEOREM 4.4. $\mathbf{C}_\infty = \mathbf{L}^{\mathbf{CM}}$.

*Proof.* (i) $\mathbf{C}_\infty \subseteq \mathbf{L}^{\mathbf{CM}}$. We must show that $\forall C_\infty \in \mathbf{C}_\infty$, $\exists L^{CM} \in \mathbf{L}^{\mathbf{CM}}$ such that $C_\infty = L^{CM}$. Given $C_\infty$, there exists $L^{CM} \subseteq C_\infty$ by Proposition 4.2. But from Proposition 4.1, $\exists C'_\infty \in \mathbf{C}_\infty$ such that $C'_\infty \subseteq L^{CM}$. Thus $C'_\infty \subseteq L^{CM} \subseteq C_\infty$. But from Proposition 4.3, we must have that $C'_\infty = L^{CM} = C_\infty$.

(ii) $\mathbf{L}^{\mathbf{CM}} \subseteq \mathbf{C}_\infty$. We must show that $\forall L^{CM} \in \mathbf{L}^{\mathbf{CM}}$, $\exists C_\infty \in \mathbf{C}_\infty$ such that $L^{CM} = C_\infty$. Again, Propositions 4.1 and 4.2 imply that given an $L^{CM}$, we can find $C_\infty$ in $\mathbf{C}_\infty$ and $L^{CM}_1$ in $\mathbf{L}^{\mathbf{CM}}$ such that $L^{CM}_1 \subseteq C_\infty \subseteq L^{CM}$. But by definition of $\mathbf{L}^{\mathbf{CM}}$, we must have that $L^{CM}_1 = C_\infty = L^{CM}$, which proves the result. Q.E.D.

The following example will serve to establish some interesting properties of the set $\mathbf{L}^{\mathbf{CM}}$. Let $\Sigma = \{c, d, u\}$, $\Sigma_u = \{u\}$ and consider the languages

$$M = (c + d)[u(c + d)]^* \quad \text{and} \quad L = \{c\}.$$

(Refer to Figure 8.) An extension sequence could be

$$D_0 = \overline{\{c\}},$$

$$D_1 = \overline{\{cud\}},$$

$$D_2 = \overline{\{cuduc\}},$$

$$D_3 = \overline{\{cuducud\}},$$

$$D_4 = \overline{\{cuduc(ud)^2\}},$$

$$\vdots$$

$$D_k = \overline{\{cuduc(ud)^2uc(ud)^3 \ldots uc(ud)^n\}},$$

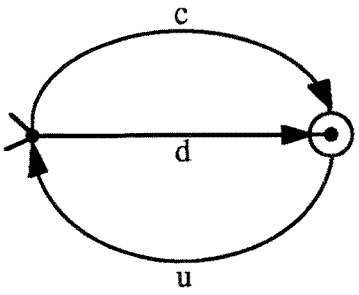


*Figure 8.*

where

$$k = \sum_{i=1}^{n} i + (n - 1) = \frac{n^2 + 3n - 2}{2},$$

and so forth.

Let $\mathfrak{R}$ denote the class of regular languages over $\Sigma$. Then the above extension sequence leads to $C_\infty \notin \mathfrak{R}$, which shows that $\mathbf{C}_\infty \not\subset \mathfrak{R}$ even though $L \in \mathfrak{R}$.

On the other hand, we can get a regular $C_\infty$ if the previous extension sequence is modified to $(\tilde{D}_i,\ i \geq 0)$, where

$$\tilde{D}_j = D_j, \qquad 0 \leq j \leq k = \sum_{i=1}^{n} i + (n - 1),$$

$$\tilde{D}_{k+1} = \overline{\{cuduc(ud)^2 \ldots (ud)^n uc\}},$$

$$\tilde{D}_{k+2} = \overline{\{cuduc(ud)^2 \ldots (ud)^n (uc)^2\}},$$

$$\vdots$$

$$\tilde{D}_{k+m} = \overline{\{cuduc(ud)^2 \ldots (ud)^n (uc)^m\}},$$

and so forth. For any fixed $k$, the extension sequence yields $C_\infty \in \mathfrak{R}$. Since $k$ can be chosen arbitrarily large, the set $\mathbf{C}_\infty \cap \mathfrak{R}$ is infinite.

We group these two important results in the following theorem

THEOREM 4.5. Let $L$ be regular. In general,

i) $\mathbf{L}^{\mathbf{CM}} \not\subset \mathfrak{R}$.
ii) $|\mathbf{L}^{\mathbf{CM}} \cap \mathfrak{R}| \not< \infty$.

In the next section, we present an algorithm based on finite-state machines that can generate some languages in $\mathbf{L}^{\mathbf{CM}} \cap \mathfrak{R}$ when $L$ is assumed to be regular.

## 5. Generating Regular Languages in $L^{CM}$

Based on the discussion of the previous section, we now propose the following algorithm to calculate a minimal controllable and $M$-closed superlanguage of a regular language $L$ when finite-state generators $G$ and $F$ of the languages $M$ and $L$, respectively, are given. The algorithm converges in finite steps.

### *5.1 Algorithm*

*Input.* $G = (Q, \Sigma, \delta, q_0, Q_m)$ and $F = (X, \Sigma, \xi, x_0, X_m)$ such that $L_m(G) = M$ and $L_m(F) = L$.

*Step 1.* Determine the map $h: X \to Q$ such that

$$(\forall\, s \in \Sigma^*)\xi(s, x_0) = x \Rightarrow \delta(s, q_0) = h(x).^1$$

*Step 2.* For all $q \in Q$ such that there are controllable events defined by $\delta$ at $q$.

$\text{select}(q) :=$ a controllable event defined at $q$.[2]

*Step 3.* For all $q \in Q$, compute $G^q$ as follows:

$Q_q^0 := \{q\}$;

$\delta_q^0 := \emptyset$;[3]

$i := -1$;

repeat

$i \leftarrow i + 1$;

$Q_q^{i+1/2} := Q_q^i \cup \{q' \in Q\colon (\exists q'' \in Q_q^i)(\exists s \in \Sigma_u^*)\delta(s, q'') = q'\}$;

$\delta_q^{i+1/2} := \delta_q^i \cup (\delta\,|(\Sigma_u \times Q_q^{i+1/2} \times Q_q^{i+1/2}))$;

$\delta_q^{i+1} := \delta_q^{i+1/2} \cup \{(\text{select}(q'), q', \delta(\text{select}(q'), q'))\colon$
$q' \notin Q_m \wedge (\forall \sigma \in \Sigma)(\forall q'' \in Q)(\sigma, q', q'') \notin \delta_q^{i+1/2}\}$;

$Q_q^{i+1} := \{q' \in Q\colon (\exists s \in \Sigma^*)\delta_q^{i+1}(s, q_0) = q'\}$;

until $Q_q^{i+1} = Q_q^i$;

$Q^q := Q_q^i$;

$\delta^q := \delta_q^i$;

$G^q := (Q^q, \Sigma, \delta^q, q, Q^q)$.

*Step 4.* For all $x \in X$, paste $G^{h(x)}$ to $F$ as follows:

$$J := \left(X \cup \bigcup_{x \in X} Q^{h(x)}, \Sigma, \xi \cup \bigcup_{x \in X} (\delta^{h(x)} \cup \{(\epsilon, x, h(x))\}), x_0, X \cup \bigcup_{x \in X} Q^{h(x)}\right).$$

*Step 5.* Transform $J$ to a deterministic (finite-state) generator.
*Step 6.* $H := J \times G$.
*Output.* $L_m(H) = L^{CM} \in \mathbf{L}^{\mathbf{CM}}$.

*Remark 5.1.* If $B_1$ is regular, then by (H4) $B_{1\max}$ is also regular, and thus the above algorithm can be applied to construct a solution of SCPT-NB by setting $L = B_{1\max}$ and $M = B_{2m}^{\uparrow}$ and by choosing a function select for Step 2.

### 5.2. *Proof of the Algorithm*

Let $L^{CM}$ be the element of $\mathbf{L}^{\mathbf{CM}}$ such that the operation $(\cdot)^e$ in the extension sequence that produces $L^{CM}$ always extends $s$ to $s(\text{select}(\delta(s, q_0)))$. We prove that the above algorithm indeed calculates $L^{CM}$; i.e., $L_m(H) = L^{CM}$.

From the construction of $H$, it is clear that $L^{CM} \subseteq L_m(H)$. Let us now prove that $L_m(H) \subseteq L^{CM}$. Let

$$E := L(J) - L(F).$$

Then

$$L_m(H) = (L(F) \cup E) \cap M.$$

From the previous section,

$$L^{CM} = D_\infty \cap M.$$

Therefore,

$$L_m(H) \not\subseteq L^{CM}$$

$$\Rightarrow (\exists s \in \Sigma^*) s \in L_m(H) \wedge s \notin L^{CM}$$

$$\Rightarrow (\exists s \in \Sigma^*)(s \in L(F) \vee s \in E) \wedge s \in M \wedge (s \notin D_\infty \vee s \notin M)$$

$$\Rightarrow (\exists s \in \Sigma^*)(s \in L(F) \vee s \in E) \wedge s \in M \wedge s \notin D_\infty$$

$$\Rightarrow (\exists s \in \Sigma^*) s \in M \wedge s \in E \wedge s \notin D_\infty \qquad \text{since } L(F) \subseteq D_\infty.$$

Since $D_\infty$ is closed,

$$(\exists t\sigma \leq s) t\sigma \in E \wedge t \in D_\infty \wedge t\sigma \notin D_\infty$$

$$\Rightarrow (\exists t\sigma \leq s)(\exists i \geq 1) t\sigma \in E \wedge t \notin D_{i-1} \wedge t \in D_i \wedge t\sigma \notin D_\infty$$

$$\Rightarrow (\exists t\sigma \leq s)(\exists i \geq 1) t\sigma \in E \wedge t \notin D_{i-1} \wedge t \in D_i \wedge t\sigma \notin D_{i+1}.$$

Let us consider two possibilities. If $\sigma \in \Sigma_u$, then

$$t\sigma \in E \Rightarrow t\sigma \in \bar{M} \Rightarrow t\sigma \in D_i^{\downarrow},$$

which contradicts $t\sigma \notin D_{i+1}$. If $\sigma \in \Sigma_c$, then

$$t\sigma \in E \Rightarrow t \notin M \wedge (\forall \sigma' \in \Sigma_u) t\sigma' \notin \bar{M} \Rightarrow t \in TBE(D_i^{\downarrow}) \Rightarrow t\sigma \in (D_i^{\downarrow})^e,$$

which contradicts $t\sigma \notin D_{i+1}$. Therefore,

$$L_m(H) \subseteq L^{CM}.$$

Let us illustrate this algorithm by the following example.

### 5.3. Example

Take the same $B_1$, $B_2$, and $L_m(G)$ as in Section 2.3. $B_{1\max}$ and $B_{2m}^{\uparrow}$ are generated by the generators $F$ and $G$ and Figures 5 and 9, respectively. We apply the algorithm to calculate $(B_{1\max})^{CM}$.

*Step 1.* The map $h$ is defined as $h(x_i) = q_i$.

*Step 2.* select($q$) is chosen as

| $q$ | select($q$) |
|---|---|
| $q_0$ | $\alpha_2$ |
| $q_1$ | $\alpha_2$ |
| $q_2$ | $\alpha_2$ |
| $q_3$ | $\alpha_1$ |
| $q_4$ | undefined |
| $q_5$ | undefined |

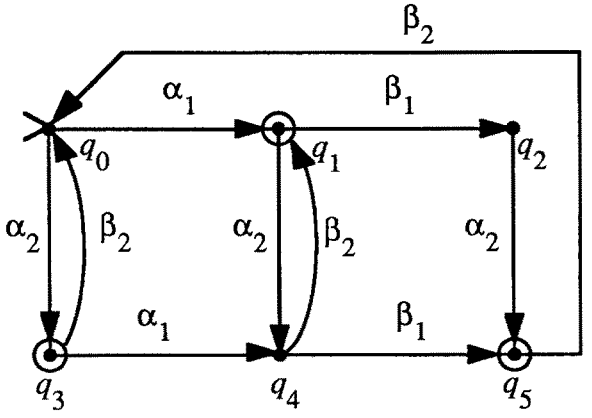


*Figure 9.*

*Step 3.* Compute $G^q$: $G^{qi}$, $i = 0, \ldots, 5$, are shown in Figures 10–15, respectively.
*Step 4.* Paste $G^{h(x)}$ to $F$.
*Step 5.* Transform the resulting generator to a deterministic one.
*Step 6.* The generator $H := J \times G$ is shown in Figure 6.

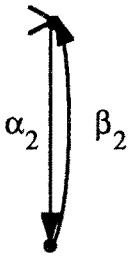


*Figure 10.*

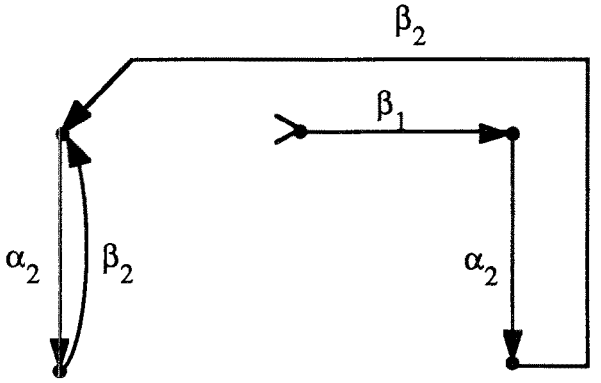


*Figure 11.*

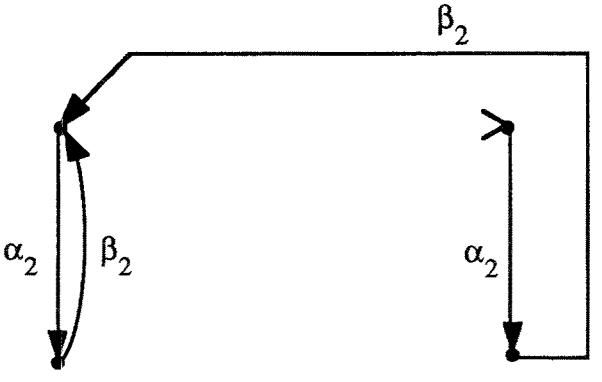


*Figure 12.*

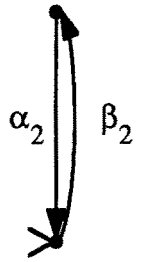


*Figure 13.*

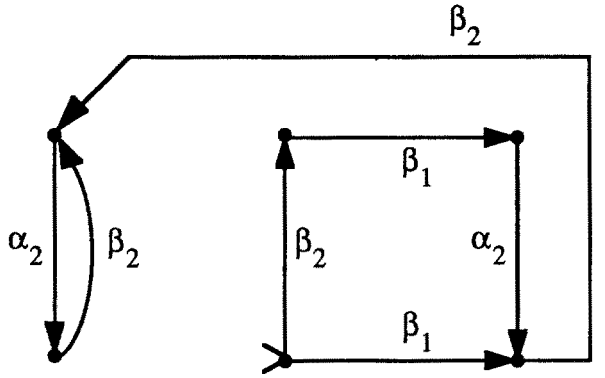


*Figure 14.*

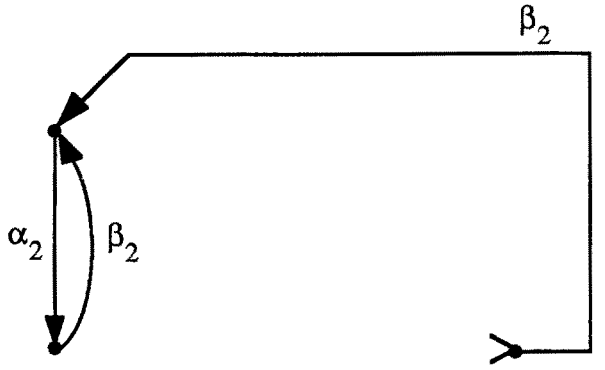


*Figure 15.*

## 6. Conclusion

We have proposed a new approach to supervisory control problems for which a formulation in terms of an ideal desired behavior and a larger tolerated behavior is appropriate. It turns out that this supervisory control problem with tolerance does not, in general, have an optimal

nonblocking solution in the presence of uncontrollable events. In order to characterize its incomparable minimal nonblocking solutions, we have developed some new results on controllable and $L_m(G)$-closed superlanguages of a given language. These results are also of independent interest.

## Acknowledgment

The authors would like to acknowledge useful discussions with E. Chen and P. Ramadge. They would also like to thank an anonymous referee for pointing out a mistake in Section 5.

## Appendix

### *A. Background Material on Supervisory Control Theory*

In this appendix, we summarize some concepts and results from supervisory control theory.

Let $\Sigma$ be a nonempty finite alphabet (i.e., set of "events"), and denote by $\Sigma^*$ the set of all finite traces (strings) of elements of $\Sigma$, including the empty trace $\epsilon$ (* is called the Kleene closure). A subset $L \subseteq \Sigma^*$ is a *language* over $\Sigma$. If $s, s', t \in \Sigma^*$ with $s't = s$, then $s'$ is a *prefix* of $s$. The *closure* $\bar{L}$ of $L$ is the language consisting of all the prefixes of traces in $L$. $L$ is *closed* if $L = \bar{L}$. Two languages $L_1$ and $L_2$ are said to be *nonconflicting* (see [Wonham and Ramadge 1988]) if $\overline{L_1 \cap L_2} = \bar{L}_1 \cap \bar{L}_2$. Closed languages are nonconflicting.

Let $M$ be a fixed language over $\Sigma$, and let $\Sigma_u$ be a fixed subset of $\Sigma$ denoting the set of uncontrollable events. A language $K \subseteq \Sigma^*$ is said to be *controllable* w.r.t $M$ and $\Sigma_u$ if $\bar{K}\Sigma_u \cap M \subseteq \bar{K}$ [Ramadge and Wonham 1987]. The set $\Sigma_c = \Sigma - \Sigma_u$ is the set of controllable events.

An uncontrolled DES is modeled by a *generator* [Ramadge and Wonham 1987], which is a deterministic automaton $G = (Q, \Sigma, \delta, q_0, Q_m)$, where $Q$ is the state space, $\Sigma$ is the set of events, $\delta: \Sigma^* \times Q \to Q$ is the transition function (a *partial* function), $q_0$ is the initial state, and $Q_m \subseteq Q$ is the set of marked states. The closed language *generated* by $G$ is $L(G) := \{s \in \Sigma^*: \delta(s, q) \text{ is defined}\}$. The language *marked* by $G$ is $L_m(G) := \{s \in L(G): \delta(s, q_0) \in Q_m\}$.

A language $K$ is said to be $L_m(G)$-*closed* if $K = \bar{K} \cap L_m(G)$. A language is *regular* iff it is marked by a finite state generator. In that case, $||L||$ denotes the minimum number of states among all generators that mark $L$.

The basic problem in supervisory control is to design a controller (or *supervisor*) whose task is to enable and disable the controllable events such that the resulting closed-loop system obeys some prespecified operating rules. Formally, a controller is a pair $S = (R, \varphi)$, where $R = (X, \Sigma, \xi, x_0, X)$ is a generator called a *recognizer* and $\varphi: \Sigma \times X \to \{0, 1\}$ is the *feedback map* satisfying

$$\varphi(\sigma, x) = 1 \qquad \text{if } \sigma \in \Sigma_u, x \in X,$$

$$\varphi(\sigma, x) \in \{0, 1\} \qquad \text{if } \sigma \in \Sigma_c, x \in X.$$

$R$ is considered to be driven externally by the stream of event symbols generated by $G$, while in turn, with $R$ in state x, the transitions $\sigma$ of $G$ are subject to the control $\varphi(\sigma, x)$. If $\varphi(\sigma, x) = 0$, then $\sigma$ is "disabled" (prohibited from occurring); if $\varphi(\sigma, x) = 1$, then $\sigma$ is "enabled" (permitted but not forced to occur). In this way, there results a closed-loop feedback structure $S/G$, called the *supervised DES*. (See [Ramadge and Wonham 1987] for further details on the definition of $S/G$.) The behavior of the supervised DES is described by the languages $L(S/G)$ and $L_m(S/G) := L(S/G) \cap L_m(G)$. In general, $\overline{L_m(S/G)} \subseteq L(S/G)$. $S$ is said to be *nonblocking* if $\overline{L_m(S/G)} = L(S/G)$.

The following important results are proved in [Ramadge and Wonham 1987]:

THEOREM A.1

i) Let $L$ be a nonempty sublanguage of $L(G)$. There exists a controller $S$ such that $L(S/G) = L$ iff $L$ is closed and controllable.
ii) Let $L$ be a nonempty sublanguage of $L_m(G)$. There exists a *nonblocking* controller $S$ such that $L_m(S/G) = L$ iff $L$ is controllable and $L_m(G)$-closed.

If a given language $L \subseteq M$ is not controllable, one may calculate the unique largest controllable sublanguage of $L$. This supremal controllable sublanguage of $L$ is denoted $L^{\uparrow}$ and is defined by

$$L^{\uparrow} := \bigcup \{K: (K \subseteq L) \wedge (\bar{K}\Sigma_u \cap M = \bar{K})\}.$$

On the other hand, one may also calculate the infimal closed controllable superlanguage of $L$, denoted $L^{\downarrow}$ and defined by

$$L^{\downarrow} := \bigcap \{K: (\bar{L} \subseteq K \subseteq M) \wedge (K = \bar{K}) \wedge (\bar{K}\Sigma_u \cap M = \bar{K})\}.$$

Both $L^{\uparrow}$ and $L^{\downarrow}$ exist and can be computed in finite steps in the regular case. Brandt et al. [1991], Lafortune and Chen [1990], and Ramadge and Wonham [1989] can be consulted for more details on these two important concepts.

Let $L_1$ and $L_2$ be two sublanguages of $M$. The following results are proved in [Wonham and Ramadge 1988] and [Lafortune and Chen 1990].

$$(L_1 \cup L_2)^{\uparrow} \supseteq L_1^{\uparrow} \cup L_2^{\uparrow}; \qquad (L_1 \cap L_2)^{\uparrow} \subseteq L_1^{\uparrow} \cap L_2^{\uparrow};$$

$$(L_1 \cup L_2)^{\downarrow} = L_1^{\downarrow} \cup L_2^{\downarrow} \qquad \text{but} \qquad (L_1 \cap L_2)^{\downarrow} \subseteq L_1^{\downarrow} \cap L_2^{\downarrow}.$$

When $L_1$ and $L_2$ are nonconflicting, then

$$(L_1 \cap L_2)^{\downarrow} = L_1^{\downarrow} \cap L_2^{\downarrow}.$$

Similarly, when $L_1^{\uparrow}$ and $L_2^{\uparrow}$ are nonconflicting, then

$$(L_1 \cap L_2)^{\uparrow} = L_1^{\uparrow} \cap L_2^{\uparrow}.$$

*B. Some Technical Results*

LEMMA B.1. Let $K = \bar{K} \subseteq \Sigma^*$. Then $K \cap L_m(G)$ is $L_m(G)$-closed.

*Proof.* Clearly, $K \cap L_m(G) \subseteq \overline{K \cap L_m(G)} \cap L_m(G)$. For the reverse inclusion,

$$\overline{K \cap L_m(G)} \cap L_m(G) \subseteq \bar{K} \cap \overline{L_m(G)} \cap L_m(G)$$

$$= K \cap L_m(G).$$

LEMMA B.2. Let $A \subseteq B \subseteq C \subseteq \Sigma^*$. If $A$ is controllable w.r.t. $B$ and $B$ is controllable w.r.t. $C$, then $A$ is controllable w.r.t. $C$.

*Proof.*

$$\bar{A}\Sigma_u \cap \bar{C} \subseteq \bar{A}\Sigma_u \cap \bar{B}\Sigma_u \cap \bar{C}$$

$$\subseteq \bar{A}\Sigma_u \cap \bar{B}$$

$$\subseteq \bar{A}.$$

LEMMA B.3. Let $M$ be livelock-free and let $L \subseteq M$ be $M$-closed. Then $L$ is livelock-free.

*Proof.* By contradiction, let us assume that $L$ is not livelock-free. This means that

$$(\exists\, s \in \bar{L})(\forall n \in \mathbb{N})(\exists t \in \Sigma^*)(|t| \geq n \wedge st \in \bar{L} \wedge (\forall u \in \Sigma^*)(s \leq u \leq st \Rightarrow u \notin L)).$$

Let $s$ be the trace that satisfies the above condition. Since $s \in M$ and $M$ is livelock-free, we can find $k(s) \in \mathbb{N}$ such that the livelock-free condition for $s \in M$ is satisfied; i.e.,

$$(\forall t \in \Sigma^*)|t| \geq k(s) \wedge st \in \bar{M} \Rightarrow (\exists u \in \Sigma^*)(s \leq u \leq st \wedge u \in M).$$

Now, return to the above assumption concerning $s \in \bar{L}$, take $n \geq k(s)$ there, and let $t$ be the appropriate suffix that works with $n$. We have that $st \in \bar{L} \Rightarrow st \in \bar{M}$, but for all $u$ such that $s \leq u \leq st$, $u \notin L$. However, since $M$ is livelock-free and since we chose $n \geq k(s)$, $\exists u' \in \Sigma^*$ such that $s \leq u' \leq st$ and $u' \in M$. Moreover, $u' \in \bar{L}$. Since $L$ is $M$-closed, $u' \in L$. This yields a contradiction. Thus $L$ is livelock-free.

## Notes

1. If necessary, this can be done by taking the new $F$ as the product of $F$ and $G$: $F := F \times G$. For the new $F$, states are pairs $(x, q)$. $h$ is then defined as $h((x, q)) = q$.
2. If there is more than one controllable event defined at $q$, arbitrarily select one.
3. We can also view a transition function as a relation; i.e., $\delta(\sigma, q) = q'$ iff $(\sigma, q, q') \in \delta$.

## References

R.D. Brandt, V. Garg, R. Kumar, F. Lin, S.I. Marcus, and W.M. Wonham, "Formulas for calculating supremal controllable and normal sublanguages," *Systems Control Lett.*, vol. 15, pp. 111–117, 1990.

E. Chen and S. Lafortune, "Dealing with blocking in supervisory control of discrete event systems," *IEEE Trans. Automat. Control*, vol. 36, no. 6, 1991.

S. Lafortune, "Modeling and analysis of transaction execution in database systems," *IEEE Trans. Automat. Control*, vol. 33, pp. 439–447, 1988.

S. Lafortune and E. Chen, "The infimal closed controllable superlanguage and its application in supervisory control," *IEEE Trans. Automat. Control*, vol. 35, pp. 398–405, 1990.

F. Lin, "Supervisory control of stochastic discrete event systems," in *Book of Abstracts, SIAM Conf. Control in the 90's*, San Francisco, 1989.

F. Lin and W.M. Wonham, "On observability of discrete-event systems," *Inform. Sci.*, vol. 44, pp. 173–198, 1988.

P.J. Ramadge and W.M. Wonham, "Supervisory control of a class of discrete event systems," *SIAM J. Control Optim.*, vol. 25, pp. 206–230, 1987.

P.J. Ramadge and W.M. Wonham, "The control of discrete event systems," *Proc. IEEE*, vol. 77, pp. 81–98, 1989.

W.M. Wonham and P.J. Ramadge, "Modular supervisory control of discrete-event systems," *Math. Control Signals Syst.*, vol. 1, pp. 13–30, 1988.