Control of Discrete Event Systems Modeled with Deterministic Büchi Automata *

Stanley Young, Damir Spanjol, Vijay K. Garg
Department of Electrical and Computer Engineering,
University of Texas, Austin, TX 78712
stanley@pine.ece.utexas.edu

January 6, 1994

Abstract

A discrete event system (DES) is a dynamic system which evolves in response to the occurrence of specific events at discrete points in time. Ramadge and Wonham have established a control theory of DES modeled by state machines which has been expanded by others to include the concepts of observability and stability. Previous work by Ramadge extended the concept of controllable languages to infinite languages and presented conditions for the existence of a supervisor for systems modeled by Büchi automata. The goal of this paper is to derive requirements for the existence of a supervisor under less restrictive conditions on the constraint language for plants which satisfy certain conditions. This supervisor is shown to approach the prescribed closed loop behavior and retain all behaviors within a specified error bound of the desired behavior. Both deterministic and nondeterministic supervisors are considered. The construction for such a supervisor is given along with examples of its application [1].

*Supported in part by NSF CCR-911065, and in part by the University Research Institute, University of Texas at Austin.

[1]A preliminary version of this paper appeared as [20]

1 Introduction

A discrete event system is a dynamic system which evolves in time in response to the occurrence of specific events at discrete points in time [7]. The dynamics of a general discrete event dynamic system are described as nondeterministic transitions which occur in response to events occurring at random asynchronous times. The state of such systems can be described with logical, symbolic, or numeric values depending on the application, such as manufacturing systems or computer networks.

Control of discrete event dynamic systems consists of supervising the system so that the performance of the system conforms to some previously specified behavior. System control is accomplished by enabling or disabling events based on the previous inputs, the observed output, or the current state of the system. By enabling or disabling certain events, the supervisor can insure that the system conforms to the desired behavior if the system meets certain controllability constraints [17, 19, 14]. Controlling resources on a computer network such as access to printers or network gateways can be modeled and accomplished using discrete event formalism and control concepts.

Infinite strings are used to model infinite sequences of events which a discrete event dynamic system can accomplish. Büchi [2] originally used infinite strings in the investigation of a decision procedure for a system of logic known as "the Sequential Calculus." In a more related application, Muller [13] used infinite strings as a means of describing and analyzing the behavior of asynchronous switching circuits. When systems are modeled using infinite strings, nonterminating behaviors can be explicitly considered and various asymptotic, or steady-state, results can be investigated [16]. McNaughton [12] showed that the language recognized by a Büchi machine is a regular ω-language; hence, Büchi automata are sufficient to model systems which can be described by regular ω-languages.

Often one wants to insure that the asymptotic behavior of systems, which can be described by discrete event formalisms, such as Büchi automata, conforms to certain constraints, or meets certain conditions [16, 1, 14, 18]. In order to insure that certain asymptotic behaviors occur, one must be able to make assertions concerning the infinite behavior of systems described by Büchi automata. Previous work requires that the ω-languages describing the system and the desired behavior have certain characteristics for the existence of supervisors [19, 16, 10, 18]. If these conditions are not met, then little can be determined about the kind of supervisory performance which can be obtained. A goal of this work is to relax some of the conditions on the constraint language required by the previous work and determine what type of supervisor performance can be achieved. In this study, we limit our attention to systems which can be represented by deterministic Büchi automata. In this work, the requirement that the language which describes the desired behavior be topologically closed with respect to the language which describes the system behavior is relaxed. We obtain a constructive proof which provides that the behavior of the closed loop system can be within a specified error bound of the desired behavior if the constraint language is controllable and stabilizable with respect to the language which describes the plant behavior [16, 1, 14, 18].

Section 2 briefly describes some of the results relevant to infinite strings, finite automata, and controllable languages. Section 3 presents the main results and compares the results to some previous results for systems described with Büchi automata. Section 4 provides some examples of how the various approaches apply in different situations.

We use calculational style of proofs for many of our Theorems. A proof that $[A \equiv C]$ will be rendered in our format as

A
$=$ { hint why $[A \equiv B]$ }
B
$=$ { hint why $[B \equiv C]$ }
C.

We also allow implies ($\Rightarrow$) in the leftmost column. For a thorough treatment of this proof format we refer interested readers to [4].

2 Description of the Model

2.1 Infinite Strings

We first define the set, A^*, as the set of all finite, unbounded strings of elements from the alphabet, A, including the empty string, ε, and let $A^+ = A^* - 1$. A subset $L \subseteq A^*$ is called a language over A.

In order to extend the concept of finite strings to infinite strings[2], we define a morphism $e : N \rightarrow A$ and the set of such morphisms as A^ω. This morphism defines an infinite sequence of elements of A indexed by N, where $N = \{1, 2, \ldots\}$. Then for a given $n \in N$, we denote by e^n the element

$$e^n = e(0)e(1)e(2)...e(n) \in A^*.$$

The element e^n is called the initial segment or ***prefix of length n*** of e and e^0 is defined to be ε, the empty string. Also observe that $e(k) \in A$, $k \in N$, is the k^{th} symbol in the string e. Note that uv denotes string u with string v concatenated to the end of u. One defines $s \in A^*$ as a prefix of $e \in A^\omega$, denoted $s < e$, if for some $j \in N, s = e^j$. Prefixes are related to infinite strings by the relation that for a given sequence $s_0 < s_1 < \ldots < s_n < \ldots$ of elements of A^*, there exists a unique $e \in A^\omega$ such that $e^k = s_n$ with $k = |s_n|$ for all $n \in N$. One writes $e = lim_{n\rightarrow\infty} s_n$.

For finite languages, one defines the prefix closure of $L \subseteq A^*$ to be the subset $cl(L) \subseteq A^*$ defined by $cl(L) = \{u : uv \in L \text{ for some } v \in A^*\}$, i.e., $cl(L)$ is the set of all prefixes of strings in L. L is prefix closed if $L = cl(L)$.

If $X \subseteq A^*$ is a language of finite strings, define X^∞, the ***limit of X***, as the set of infinite strings $\sigma \in A^\omega$ having an infinity of prefixes in the set X [15]. Hence, for $\sigma \in A^\omega$,

$$\sigma \in X^\infty \text{ if and only if } \exists^\infty k : \sigma^k \in X.$$

where $\exists^\infty k$ means that there exists an infinity of k which satisfy the predicate.

A set $B \subseteq A^\omega$ is called an ω-language over A. The prefix of B is the set of all strings in A^* which are prefixes of some string in B. Formally, one defines the ***prefix of*** $B \subseteq A^\omega$ as the set $pr(B) \subseteq A^*$ such that $pr(B) = \{e^j : e \in B, j \geq 0\}$. When $B = \{e\}$, for $e \in A^\omega$, we write $pr(e)$ for $pr(\{e\})$.

The ***adherence*** or ***limit*** of $L \subseteq A^*$ is the ω-language

$$L^\infty = \{e : e \in A^\omega \text{ and } \exists^\infty j : e^j \in L\}.$$

From the definitions given above, one can derive the following relationships between the prefix and adherence operators: for any $B \subseteq A^\omega$ and prefix closed $K \subseteq A^*$

$$B \subseteq pr(B)^\infty, \text{ and}$$

[2]This material on infinite strings follows the presentations given in [5, 15, 16].

$$pr(K^\infty) \subseteq K.$$

A metric, d, can be defined on A^ω as follows:

$$d(e_1, e_2) = \begin{cases} 1/n & \text{if } e_1^k = e_2^k, 0 \leq k < n, \text{ and } e_1(n) \neq e_2(n) \\ 0 & \text{if } e_1 = e_2. \end{cases}$$

The ***topological closure*** of a set $B \subset A^\omega$ with respect to the above metric is denoted $\overline{B}$. The topological closure, prefix and adherence operations for $B \subseteq A^\omega$ are related by:

$$\overline{B} = pr(B)^\infty.$$

For subsets B and S, which satisfy $B \subseteq S \subseteq A^\omega$, we say that B is topologically closed relative to S if $\overline{B} \cap S = B$. If $B \subseteq S$, it is always the case that $B \subseteq \overline{B} \cap S$, and, consequently, B is topologically closed relative to S if and only if $\overline{B} \cap S \subseteq B$.

As discussed later, the concept of topological closure can play a role in determining existence of a class of supervisors for a discrete event dynamic systems.

2.2 Automata Description of a Discrete Event System

Since the behavior of discrete event systems can be represented as sequences of discrete events, the behavior of many discrete event systems can be modeled as a prefix closed regular language $L \subseteq A^*$. Henceforth, languages used to describe discrete event systems will be considered to be prefix closed and regular. Modeled in this way, L represents all possible (finite, unbounded) strings of events which a discrete event system (DES) can generate. An extension of this model considers the adherence of L, or a subset of it, in order to discuss the long term or limit behavior of the DES. For this extension, one models a DES as a pair $P = (L, S)$ where L is a prefix closed subset of A^* and S is a subset of L^∞ [16].

One can also describe this class of discrete event dynamic systems with finite automata. As mentioned above, McNaughton [12] and others demonstrate that the behavior of such systems can be described with rational (regular) ω-languages, often denoted by RAT or $DRAT$ for deterministic rational languages. Eilenberg [5] provides a development of the relationships between finite machines and infinite languages.

For our automaton model, we define a finite state machine, called a generator, $G = (Q, A, q_0, \delta)$, to be a dynamic system consisting of a state set Q, an event set or alphabet A, an initial state q_0, and a transition function $\delta : Q \times A \to Q$. In general, δ is only a partial function, i.e. not defined on the entire space $Q \times A$. Also, δ can be extended in a natural sense to a function on $Q \times A^*$ by $\delta(q_0, w\sigma) = \delta(\delta(q_0, w), \sigma)$. We write $\delta(q, w)!$ to denote that $\delta(q, w)$ is defined.

A machine is called a ***trim*** machine if every state is a vertex in some successful path [5]. This concept is related to accessible and co-accessible automata in [16] and live systems in [14].

The language generated by the machine G is defined to be the set

$$L(G) = \{w : w \in A^* \text{ and } \delta(q_0, w)!\}.$$

Languages generated in this manner are closed and regular [5, 8].

An infinite path in the machine G is an infinite sequence $\sigma \in A^\omega$ such that $\delta(q_0, \sigma^j)!$ for all $\sigma^j \in pr(\sigma)$. Such sequences are used to model the asymptotic behavior of DES. Consequently, one extends the definition of the generator to that of a Büchi automaton in order to capture this infinite or limit behavior.

The infinite or limit behavior, $S(G)$, of a finite state Büchi machine G can be specified in the following manner. A set of states called the marked states is added to the description of machine $G = (A, Q, \delta, q_0)$. This set is denoted by $Q_m \subseteq Q$. Q_m corresponds to the set of marked or final states in the standard finite state machine description. To each sequence of events $e \in L(G)^\infty$ there corresponds a unique state trajectory $s_e : N \to Q$ satisfying $s_e(j) = \delta(q_0, e^j)$. Related to s_e, for any $e \in A^\omega$, one defines $In(e)$, the states visited an infinite number of times by the infinite string e, by

$$In(e) = \{q \in Q : (\exists^\infty i : s_e(i) = q)\}.$$

The sequence e and trajectory s_e are said to be accepted if s_e visits the set Q_m infinitely often, i.e. $In(e) \cap Q_m \neq \emptyset$. The set of event sequences generated by $G = (A, Q, \delta, q_0, Q_m)$ is then defined to be

$$S(G) = \{e : e \in L(G)^\infty, \text{ and } s_e \text{ is accepted}\}[16].$$

In a related notation, a machine denoted by M_B is a machine which generates language B.

From the definitions of infinite behavior and the limit of a language it follows that $S(G) \subseteq L(G)^{\infty}$ with equality if $Q_m = Q$; and that $pr(S(G)) \subseteq L(G)$. In general, one need not have equality in $pr(S(G)) \subseteq L(G)$ since there may be strings in $L(G)$ that cannot be extended to an accepted event sequence. If $pr(S(G)) = L(G)$, then G is said to be ***nonblocking***. There is equality if and only if for every $q \in Q$ there exists $w \in A^*$ with $\delta(q, w) \in Q_m$, and for every $q \in Q_m$ there exists an event $\sigma \in A$ with $\delta(q, \sigma)!$ [16].

2.3 Controllability and Stability

The event set A for a DEDS described as a plant modeled by the automaton $P = (L, S)$ can be partitioned into two disjoint sets: A_c and A_u. A_u contains all events which are uncontrollable, i.e. no supervisor may affect the behavior of the plant with respect to these events. A_c contains all events which a supervisor can enable or disable in order to effect the desired behavior.

With the concept of controllable and uncontrollable events, one can define a supervisor for a plant which affects the controllable events which a plant can execute based on some specified constraints. A supervisor for a DES is a map

$$f : L \rightarrow 2^A$$

which specifies the next enabled inputs which can be applied based on the string of generated events. One can consider the closed loop system consisting of a supervisor, f, and plant, P, where the finite and infinite closed loop behaviors are denoted by L_f and S_f respectively defined as follows:

1. $\varepsilon \in L_f$;
2. $w\sigma \in L_f$ if and only if $w \in L_f, \sigma \in f(w)$, and $w\sigma \in L$;
3. $S_f = L_f^{\infty} \cap S$.

Note that the definition of L_f implies that $L_f \subseteq L$ and that $cl(L_f) = L_f$ [16].

Also from the properties given above, it follows that $pr(S_f) \subseteq L_f$. When $pr(S_f) = L_f$, one says that f is nonblocking for P [16]. A result concerning the finite behavior of nonblocking supervisors is that for all $t \in L_f$ there exists a $b \in A$ such that $tb \in L_f$.

The language $K \subseteq L$ is said to be ***controllable*** with respect to L if $cl(K)A_u \cap L \subseteq cl(K)$. A supervisor, f, is said to be ***complete*** with respect to a given plant $P = (L, S)$ if all uncontrolled semantics of the plant are respected, in other words, if $x \in L$, $x \in L_f$ and $x\sigma_u \in L$ then $x\sigma_u \in L_f$ where $\sigma_u \in A_u$ and L_f is the closed loop behavior as discussed above. For nonempty $K \subseteq L$, there exists a complete, nonblocking supervisor f such that $L_f = K$ if and only if K is prefix closed and controllable [17].

Also, the family of closed and controllable sublanguages of L is closed under set union and for any closed $K \subseteq L$ there exists a unique supremal closed and controllable language, $K^\uparrow = K - ((L - K)/A_u^*)A^*$ [17, 9].

The ***region of weak attraction***, as discussed in [1, 11, 14], can be directly related to the existence of certain types of supervisors. The algorithm given below is essentially the same one given in [1] for determining the region of weak attraction. The main result of our work relates conditions on the region of weak attraction to certain supervisor characteristics for systems which can be modeled by finite state machines or whose language descriptions are in DRAT.

For this paper, the region of weak attraction, $\Omega_M(Q_m)$, for a given machine, $M = (Q, A, \delta, q_0, Q_m)$, with a given set of marked states, Q_m, is defined by the following algorithm [1]:

Initial Step:	Set $P_0 = Q_m$ and $i = 1$.
Iteration Step:	Define $P_i = P_{i-1} \cup \left\{ q \in Q : \begin{array}{c} \exists \sigma : \delta(q, \sigma) \in P_{i-1} \\ \text{and} \\ \not\exists \sigma_u \in A_u : \delta(q, \sigma_u) \notin P_{i-1} \end{array} \right\}$
Termination Step:	If $P_i = P_{i-1}$, stop and set $\Omega_M(Q_m) = P_i$, else $i = i + 1$; goto iteration step .

This algorithm builds the region of weak attraction starting from Q_m. Each iteration of the algorithm adds states to the region defined in the previous iteration. A state is added to the region of weak attraction only if there is an event σ which describes a transition into the region defined in the previous iteration and there does not exist an uncontrolled event which provides a path to a state which is not in the region defined by the previous iteration of the algorithm. The algorithm is guaranteed to terminate by the finite state description of the machine. An efficient algorithm is given in [11] which computes the region of weak attraction in $O(|Q|)$ time.

Stability and stabilizability characteristics are often used to describe systems modeled with infinite strings. The stability of a discrete event dynamic system can be related to the region of weak attraction [1, 11, 14].

Definition 2.1 An ω-language B is ***finitely stabilizable*** if and only if $B \in DRAT$, and for all $M_B = (Q, A, \delta, q_0, Q_m)$ such that M_B is trim and recognizes B, we get that $\Omega_{M_B}(Q_m) = Q$.

Although finite stabilizability is defined in terms of machines which recognize a language, the following lemma demonstrates that finite stabilizability is a characteristic of a language and is independent of the structure of a particular machine used to generate the language.

Lemma 2.1 Let $B \in DRAT$.

If

$$\text{there exists a trim } M_B = (Q, A, \delta, q_0, Q_m) \text{ with } \Omega_{M_B}(Q_m) = Q,$$

then

$$\text{for all trim } M'_B = (Q', A, \delta', q'_0, Q'_m),\ \Omega_{M'_B}(Q'_m) = Q'.$$

Proof:
We use proof by contradiction:
Assume that there exists a machine $M_1 = (Q, A, \alpha, q_0, Q_m)$ such that:

$$M_1 \text{ is trim}, B = S(M_1), \text{ and } \Omega_{M_1}(Q_m) = Q,$$

and that there exists another machine $M_2 = (R, A, \beta, r_0, R_m)$ such that:

$$M_2 \text{ is trim}, B = S(M_2), \text{ but that } \Omega_{M_2}(R_m) \neq R.$$

Given these two machines:

$$\Omega_{M_2}(R_m) \neq R$$

$$\Rightarrow$$

$$(\exists w \in pr(B)) \wedge (r \in R : \beta(r_0, w) = r \notin P_{2,k}, \forall k \geq 0)$$

$\Rightarrow \qquad \{(\Omega_{M_1}(Q_m) = Q) \wedge (w \in pr(B))\}$

$\alpha(q_0, w) = q \in P_{1,k}$ for some k

where $P_{i,k}$'s are the sets generated in the k^{th} iteration of the algorithm which determines the region of weak attraction for the i^{th} machine.

$(\Omega_{M_1}(Q_m) = Q) \wedge (w \in pr(B))$

$\Rightarrow$

$\exists v_1 \in A^* : (\alpha(q_0, wv_1) \in Q_m) \wedge (\alpha(q_0, wv_1(0) \ldots v_1(n)) \in P_{1,k-n}, \forall n, 0 \leq n \leq k)$

$\Rightarrow \qquad \{(\Omega_{M_2}(R_m) \neq R) \wedge (M_2 \text{ trim})\}$

$(\exists u_1 < v_1) \wedge (\sigma_1 \in A_u : (u_1\sigma_1 \not\leq v_1) \wedge (\beta(r_0, wu_1\sigma_1) \notin P_{2,k}, \forall k \geq 0))$.

Let l_1 be such that $v_1(0) \ldots v_1(l_1) = u_1$.

Definition of $P_{1,k}$

$\Rightarrow$

$(\alpha(q_0, wu_1) \in P_{1,k-l_1}) \wedge (\alpha(q_0, wu_1\sigma_1) \in P_{1,k-l_1-1})$.

Let $w_2 = wu_1\sigma_1$ and $k_1 = k - l_1$.

$(\Omega_{M_1}(Q_m) = Q) \wedge (w_2 \in pr(B))$

$\Rightarrow$

$\exists v_2 \in A^* : (\alpha(q_0, w_2v_2) \in Q_m) \wedge (\alpha(q_0, w_2v_2(0) \ldots v_2(n)) \in P_{1,k_1-n}, \forall n, 0 < n \leq k_1 - 1)$

$\Rightarrow \qquad \{\Omega_{M_2}(R_m) \neq R\}$

$\exists u_2 < v_2, \sigma_2 \in A_u : \beta(r_0, w_2u_2\sigma_2) \notin P_{2,k}, \forall k \geq 0$.

Continue choosing v_i such that:

$\alpha(q_0, w_iv_i) \in Q_m$

$\Rightarrow \qquad \{(\Omega_{M_1}(Q_m) = Q) \wedge (w_i \in pr(B)) \wedge (\Omega_{M_2}(R_m) \neq R)\}$

$\exists u_i < v_i, \sigma_i \in A_u : \beta(r_0, w_i u_i \sigma_i) \notin P_{2,k}, \forall k \geq 0$ where $w_i = w_{i-1} u_{i-1} \sigma_{i-1}$

$\Rightarrow$ $\{M_1, M_2 \text{ trim}\}$

$\exists w_j \in pr(B) : ((\alpha(q_0, w_j) \in Q_m) \wedge (\beta(r_0, w_j) \notin \Omega_{M_2}(R_m)) \wedge (|w_j| > |Q| + |R|))$

$\Rightarrow$

$\exists$ cycle $w_j(m) \ldots w_j(n)$, $0 \leq m < n \leq |w_j|$, where $(w_0 w_j(m) \ldots w_j(n) \leq w_j) \wedge (w_0 \leq w_j(0) \ldots w_j(m))$

$\Rightarrow$

$(In_1(w_0(w_j(m) \ldots w_j(n))^\omega) \cap Q_m \neq \emptyset) \wedge (In_2(w_0(w_j(m) \ldots w_j(n))^\omega) \cap R_m = \emptyset)$

where In_1 and In_2 denote the states which are visited an infinite number of times by w_j in M_1 and M_2 respectively.

Using this technique, we have constructed an infinite string which is accepted by M_1 and not accepted by M_2, which contradicts the fact that they both are trim machines which recognize B. Hence, the assumption that $\Omega_{M_2}(R_m) \neq R$ could not be correct and the lemma is proved.
Q.E.D.

Armed with these concepts one can discuss applications in the area of supervisory control.

3 Supervisory Control of Systems Modeled with Infinite Strings

3.1 Deterministic Supervisors

For this paper, finite automata on infinite strings are used to model supervisory control of DES. The motivation for such an application arises from two considerations.

The first is that often plants or systems which need to be controlled can be modeled as deterministic or nondeterministic automata; hence, one needs a technique to analyze and synthesize supervisors for such systems.

The second consideration arises from the fact that the events which describe plant activity can be partitioned into controllable, A_c, and uncontrollable, A_u, events. For the control of the system, the supervisor must not attempt to block any actions which include uncontrollable events.

The primary motivation for this work is a result by Ramadge [16] relating to the controllability of systems modeled by Büchi automata. The relevant result is stated in the following theorem:

Theorem 3.1 ([16]) If $B \subseteq S \subseteq A^\omega$ is nonempty, then,

> there exists a complete, nonblocking supervisor f for plant $P = (L, S)$ such that $S_f = B$,

if and only if

> $pr(B)$ is controllable with respect to $L, i.e., pr(B)A_u \cap L \subseteq pr(B)$, and
>
> B is topologically closed relative to S, $i.e., \overline{B} \cap S = B$.

One of the characteristics which limit the application of the above theorem is the requirement that B must be topologically closed with respect to S. One of the primary goals of this work is to determine when a supervisor exists for systems which do not satisfy this requirement. For situations in which a supervisor is shown to exist, we give a construction for a complete, nonblocking supervisor.

Example 1:

This example demonstrates a discrete event dynamic system $P = (L, S)$ with the event set $A = \{a, b\}$ which illustrates the application of Theorem 3.1. The dynamics of the system are described by

$$S = a^\omega + a^*ba^\omega \text{ and } L = a^* + a^*ba^*(= pr(S)).$$

The uncontrollable events are described by $A_u = \{b\}$. The finite state machine shown in Figure 1 generates this language.

Consider the constraints imposed by $B = a^*ba^\omega \subset S$. Since $pr(B) = a^* + a^*ba^* = L$, it is clear that $pr(B)A_u \cap L \subseteq pr(B)$, or that $pr(B)$ is controllable. This controllability implies that a supervisor, f, exists such that $L_f = pr(B)$ [16]. However,

$$pr(B)^\infty \cap S = S \neq B$$

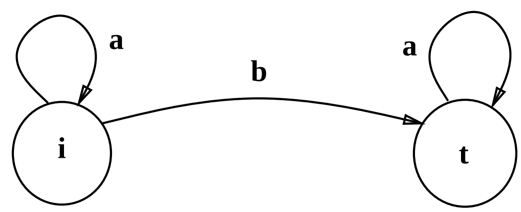

Figure 1: Machine for Example 1.

i.e., B is not topologically closed relative to S. According to the above theorem, this fact implies that there is not a nonblocking supervisor f such that $S_f = B$; yet supervisors for such systems are still needed.

Lemma 3.1 and Theorem 3.2 extend previous results to demonstrate the existence of a complete nonblocking supervisor, f, for a given plant, $P = (L, S)$ when certain criteria are met by the plant and required behavior. The motivation for the extension comes from situations as in the example above where the language describing the desired behavior is not topologically closed with respect to the language describing the plant behavior. The closed loop behavior is shown to be contained in the desired behavior but not necessarily equivalent to it; the supervisor does not allow any behavior which is "bad" but might restrict some of the "good" behaviors. In some applications, such a supervisor would be adequate. The strategy for this extension consists of specifying a supervisor which will restrict the plant to sequences in the topological closure of the constraint language then to provide an extension to this supervisor which allows the closed loop behavior to stay within the constraint language yet approach to within a specified distance of the boundary of the constraint language.

Lemma 3.1 provides that the machine, M_B, which recognizes the desired behavior, B, acts as a supervisor which gives a closed loop behavior as the topological closure of B if $pr(B)$ is controllable with respect to $pr(S)$.

Lemma 3.1 Let $B \subseteq S \subseteq A^\omega$, and M_S nonblocking, where $pr(S) = L$, and $B \in DRAT$, and $M_B = \{Q, A, \delta_B, q_0, Q_m\}$ be a trim machine. Then

$pr(B)$ is controllable with respect to L,

if and only if

$f(w) = \{\sigma \in A : \delta_B(q_0, w\sigma)!\}$ is a supervisor for $P = (L, S)$ such that:

1. f is complete,
2. f is nonblocking,
3. $S_f = \overline{B} \cap S$.

Proof:
($\Leftarrow$)
This assertion follows directly from [17].
($\Rightarrow$)

1. f is complete:

   Let $s \in L_f$ and $s\sigma_u \in L$ where $\sigma_u \in A_u$: we must show that $s\sigma_u \in L(M_B)$.

   $$
   \begin{aligned}
   & (L_f = L(M_B)) \wedge (\text{definition of } f) \\
   \Rightarrow & \\
   & s \in L(M_B) \\
   \Rightarrow & \qquad \{(L(M_B) = pr(B)) \wedge (B \in DRAT)\} \\
   & s \in pr(B) \\
   \Rightarrow & \qquad \{pr(B) \text{ controllable}\} \\
   & s\sigma_u \in pr(B) \\
   \Rightarrow & \\
   & s\sigma_u \in L(M_B).
   \end{aligned}
   $$

2. f is nonblocking:

   We must show that $pr(S_f) = L_f$.

   Let $t \in pr(S_f)$.

   $$
   \begin{aligned}
   \Rightarrow & \qquad \{S_f = L_f^\infty \cap S\} \\
   & t \in pr(L_f^\infty \cap S) \\
   \Rightarrow &
   \end{aligned}
   $$

$t \in pr(L_f^\infty)$

$\Rightarrow$ $\{$definition of $L_f^\infty\}$

$t \in L_f$.

Let $t \in L_f$.

$\Rightarrow$ $\{L_f = L(M_B)\}$

$(t \in pr(B)) \wedge (t \in pr(L_f^\infty))$

$\Rightarrow$ $\{M_B$ is trim$\}$

$t \in pr(B \cap L_f^\infty)$

$\Rightarrow$ $\{B \subseteq S\}$

$t \in pr(S \cap L_f^\infty)$

$\Rightarrow$ $\{$definition of $S_f\}$

$t \in pr(S_f)$.

3. $S_f = \overline{B} \cap S$:

Definition of S_f

$\Rightarrow$

$S_f = L_f^\infty \cap S$

$\Rightarrow$ $\{$definitions of f and $L(M_B)\}$

$S_f = L(M_B)^\infty \cap S$

$\Rightarrow$ $\{(L(M_B) = pr(B)) \wedge (B \in DRAT)\}$

$S_f = pr(B)^\infty \cap S$

$\Rightarrow$ $\{$definition of $\overline{B}\}$

$S_f = \overline{B} \cap S$.

Q.E.D.

Theorem 3.2 gives results for the existence of a supervisor when the language describing the desired behavior is not closed relative to the language which describes the plant behavior. If finite stabilizability is added as a condition on the constraint language, then the closed loop behavior is shown to be contained in the desired behavior. We prove each part in separate lemmas and then combine the results into the main theorem, Theorem 3.2.

Lemma 3.2 Let $B \subseteq S$ be nonempty, M_S nonblocking, where $B, S \in DRAT$, and S is topologically closed with respect to A^ω.

If

B is finitely stabilizable, and

$pr(B)$ is controllable with respect to $L = pr(S)$,

then

for all $\epsilon > 0$, there exists:

nonblocking, complete supervisor f for P, such that:

$S_f \subseteq B$, and

$\forall u \in B, \exists v \in S_f : d(u, v) < \epsilon$.

Proof:
In order to show that such a supervisor exists, we will construct one.

Let M_B be a trim machine which recognizes B where $M_B = \{Q, A, \delta, q_0, Q_m\}$. Since $pr(B)$ is controllable with respect to $pr(S)$, from Lemma 3.1, M_B is a complete, nonblocking supervisor, f', such that $S_{f'} = \overline{B} \cap S$.

We must modify f' to get a supervisor f so as to retain the complete and nonblocking characteristics yet further constrain S_f such that $S_f \subseteq B$.

First, construct a machine M_c which runs in parallel with M_B. This machine, M_c, is essentially a counter which allows the supervisor to achieve the ϵ distance criterion.

Define $M_c = (Q_c, A, \delta_c, q_{c0}, Q_{mc})$ where

$$Q_c = \{q_{c0}, q_{c1}, \ldots, q_{cN-1}\}, N > 1/\epsilon, Q_{mc} = \{q_{c0}\}, \text{ and}$$

$$\delta_c(q_{ci}, \sigma) = \begin{cases} q_{c0} & \text{if } \delta(q_0, w) \in Q_m, \\ q_{ci+1} & \text{if } \delta(q_0, w) \notin Q_m \text{ and } i < N - 2, \\ q_{cN-1} & \text{otherwise .} \end{cases}$$

And define the supervisor feedback map f by

$$f(w) = \begin{cases} A'(q_w) & \text{if } \delta_c(q_{c0}, w) \neq q_{cN-1}, \\ A'(q_w) - A_{bad}(q_w) & \text{if } \delta_c(q_{c0}, w) = q_{cN-1}, \end{cases}$$

where:

$$\begin{aligned} q_w &= \delta(q_0, w), \\ A'(q_w) &= \{\sigma \in A : \delta(q_0, w\sigma)!\}, \end{aligned}$$

and

$$\begin{aligned} A_{bad}(q_w) &= \text{ set of events to disable },\\ &= \{\sigma \in A'(q_w) \cap A_c : \delta(q_0, w\sigma) \notin P_{i-1} \text{ where } \delta(q_0, w) \in P_i\}, \end{aligned}$$

with P_i and $\Omega_{M_B}(Q_m)$ defined as follows:

$$\begin{aligned} P_0 &= Q_m, \\ P_i &= P_{i-1} \cup \left\{ q : \begin{array}{l} \exists \sigma : \delta(q, \sigma) \in P_{i-1}, \text{ and} \\ \nexists \sigma_u \in A_u : \delta(q, \sigma_u) \notin P_{i-1} \end{array} \right\}, \\ \Omega_{M_B}(Q_m) &= P_i \text{ when } P_i = P_{i-1}. \end{aligned}$$

To show that f is *complete*, one must show that if an uncontrolled event is defined in the plant from a state which is reached by the closed loop system, then that event is allowed by the supervisor; i.e.

$$(s \in L_f) \wedge (s\sigma_u \in L) \wedge (\sigma_u \in A_u) \Rightarrow s\sigma_u \in L_f.$$

From the definitions of $f(s)$ and completeness it is sufficient to show that $\sigma_u \in A'(q_s)$ and $\sigma_u \notin A_{bad}(q_s)$.

Since $pr(B)$ is controllable with respect to L, one immediately has that for all $\sigma_u \in A_u$ if $s \in L_f$ and $\delta(q_0, s\sigma_u)!$ then $\sigma_u \in A'(q_s)$ as desired.

Since $\sigma_u \notin A_c$, the definition of $A_{bad}(q_s)$ provides that $\sigma_u \notin A_{bad}(q_s)$ as desired; hence, f is *complete*.

To show that f is *nonblocking*, one must show that $pr(S_f) = L_f$. By Lemma 3.1, $S_{f'} = \overline{B} \cap S$. By construction, $S_f \subseteq \overline{B} \cap S$. By the finite stabilizability of B and the construction of A_{bad}, $S_f \neq \emptyset$. The proof of nonblocking follows in a manner similar to the nonblocking part of the proof of Lemma 3.1.

To show that $S_f \subseteq B$, one must show that any string in the closed loop behavior, S_f, is contained in the desired behavior, B: i.e.

$$u \in S_f \Rightarrow u \in B.$$

We will use proof by contradiction. Assume that there exists a $u \in S_f$ such that $u \notin B$.

Definitions of S_f and M_B

$\Rightarrow$

$\exists w \in pr(u) : (\forall x : wx \in pr(u) : \delta(q_0, wx) \notin Q_m)$

$\Rightarrow$ $\{$definition of $M_c\}$

for this w, $\exists y \in A^* : ((wy \in pr(u)) \wedge (\delta_c(q_{c0}, wy) = q_{cN-1}))$

$\Rightarrow$ $\{u \in S_f$ and definition of $f(wy)\}$

$\exists y, v \in A^* : (wyv \in pr(u)) \wedge (\delta(q_0, wyv) \in Q_m),$ for $|v| \leq |Q|$.

Thus, $\delta(q_0, wyv) \in Q_m$ which contradicts the original choice of w; hence, $u \in B$.

To show that for every string in the desired behavior, B, there is a string within ϵ of it which is in the closed loop behavior, S_f, i.e. $\forall u \in B, \exists v \in S_f$ such that $d(u, v) < \epsilon$, one can consider two cases:

1. If $u \in S_f$, then let $v = u$. In this case, both u and v are in S_f; hence, $d(u, v) = 0 < \epsilon, \forall \epsilon > 0$.
2. If $u \notin S_f$, then $u \in B - S_f$.

Since $u \in B - S_f$, there exists a $w \in pr(u)$ such that $(w \in pr(B))$ and $(\delta_c(q_{c0}, w) = q_{cN-1})$. This situation arises because f must have disabled some event σ in order for $u \notin S_f$.

Pick w, σ so that it is the first string in the prefix of u such that $\delta_c(q_{c0}, w) = q_{cN-1}$ and $\sigma \notin f(w)$.

Since f is nonblocking, $w \in L_f$ implies that there exists an $\alpha \in S_f$ such that $w \in pr(\alpha)$.

Now calculate the distance between u and any $\alpha \in S_f$ such that $w \in pr(\alpha)$. As a result of the choice of w and σ, this distance satisfies the following condition:

$$d(u, \alpha) = \frac{1}{|w|}.$$

Since $\delta_c(q_{c0}, w) = q_{cN-1}, |w| \geq N$; consequently, this distance satisfies the following inequality:

$$d(u, \alpha) = \frac{1}{|w|} \leq \frac{1}{N} < \epsilon.$$

The last inequality follows from the choice of N.
Q.E.D.

Lemma 3.3 Let $B \subseteq S$ be nonempty, M_S nonblocking, where $B, S \in DRAT$, and S is topologically closed with respect to A^ω.

If

for all $\epsilon > 0$, there exists:

nonblocking, complete supervisor f for P, such that:
$S_f \subseteq B$, and
$\forall u \in B, \exists v \in S_f : d(u, v) < \epsilon$,

then

B is finitely stabilizable, and

$pr(B)$ is controllable with respect to $L = pr(S)$.

Proof:
For this proof, we will show the contrapositive: i.e. assume that if

1) B is not finitely stabilizable

(i.e. there exists $q = \delta(q_0, w)$, $q \in Q$, where Q is the state set for machine which recognizes B, such that $q \notin P_k$ where $w \in pr(B)$ and $\forall k \geq 0$),

or

2) $pr(B)$ is not controllable with respect to L

(i.e. $\exists t \in pr(B)$ such that $\exists a \in A_u$ for which $ta \notin pr(B)$ but $ta \in L$),

then for each $\epsilon > 0$ there does not exist a complete, nonblocking supervisor where for all $u \in B$ there exists a $v \in S_f$ such that $d(u, v) < \epsilon$ and $S_f \subseteq B$.

We will show that any complete, nonblocking supervisor with $d(u, v) < \epsilon$ where $u \in B$ and $v \in S_f$ does not have $S_f \subseteq B$ if $pr(B)$ is not controllable or B is not finitely stabilizable.

From the assumption that $pr(B)$ is not controllable, it follows that there exists a $t \in pr(B)$ such that there exists an $a \in A_u$ for which $ta \notin pr(B)$.

For this t, a pair choose ϵ such that $0 < \epsilon \leq \frac{1}{|t|+1}$.

For any $\beta \in B : (t \in pr(\beta) \Rightarrow d(\alpha, \beta) < \epsilon)$ and f nonblocking

$\Rightarrow$

$\exists \alpha \in S_f : t \in pr(\alpha)$

$\Rightarrow$ $\{f$ complete and $t \in L_f$ by choice of $\epsilon\}$

$ta \in L_f$

$\Rightarrow$ $\{f$ nonblocking$\}$

$\exists y \in A^\omega : tay \in S_f$

$\Rightarrow$ $\{tay \notin B$ since $ta \notin pr(B)\}$

$S_f \not\subseteq B$

which is as required for this part of the contrapositive.

If B is not finitely stabilizable, then:

M_B a trim machine

$\Rightarrow$

$\exists q = \delta(q_0, w), q \in Q : q \notin P_k, \forall k \geq 0,$ and $w \in pr(B)$

$\Rightarrow$ $\{f$ is nonblocking$\}$

$\exists \beta \in B : w \in pr(\beta)$.

Choose $\epsilon = \frac{1}{|w|+1}$.

Hypothesis that $\forall u \in B, \exists v \in S_f : d(u, v) < \epsilon$

$\Rightarrow$

$\exists \alpha \in S_f : d(\alpha, \beta) < \epsilon$

$\Rightarrow$ $\{$choice of $\epsilon\}$

$w \in pr(\alpha)$

$\Rightarrow$ $\{$definition of $S_f\}$

$w \in L_f$.

If we assume that $\alpha \notin B$, then $S_f \not\subseteq B$ as required for the contrapositive. Otherwise, assume that $\alpha \in B$:

$w \in pr(B)$

$\Rightarrow$

$\exists v : (wv \in pr(\beta) \wedge (\delta(q_0, wv) \in Q_m))$

$\Rightarrow$ $\{$else $q \in P_k$ for some k, and definition of $P_k\}$

$\exists v_1 < v : \exists \sigma_1 \in A_u : \delta(q_0, wv_1\sigma_1) \notin P_k, \forall k \geq 0$

$\Rightarrow$ $\{\alpha \in S_f$, definition of $S_f, wv \in pr(\alpha)\}$

$wv_1 \in L_f$

$\Rightarrow$ $\{wv_1 \in L_f$ and f complete$\}$

$wv_1\sigma_1 \in L_f$.

Since $\delta(wv_1\sigma_1, q_0) \notin P_k, \forall k \geq 0$, one uses similar reasoning to show that

$$\exists v_2\sigma_2 : wv_1\sigma_1v_2\sigma_2 \in L_f \text{ and } \delta(q_0, wv_1\sigma_1v_2\sigma_2) \notin P_k, \forall k \geq 0,$$

for $v_2 < v'$ such that $\delta(q_0, wv_1\sigma_1v') \in Q_m$.

Hence, as illustrated in Figure 2 where $v_i' = v_i\sigma_i$, one can construct a sequence
$v_1\sigma_1, v_2\sigma_2, \ldots, v_i\sigma_i : \delta(q_0, wv_1\sigma_1v_2\sigma_2 \ldots v_i\sigma_i) \notin P_k, \forall k \geq 0$.
Since one can construct a sequence $v_1\sigma_1, v_2\sigma_2, \ldots, v_j\sigma_j : j > |Q|$, by the pumping lemma [8], there exists a cycle in this sequence which the supervisor allows. Consequently, there exists an infinite sequence α' such that

$$\alpha\prime \in S_f : In(\alpha') \cap Q_m = \emptyset.$$

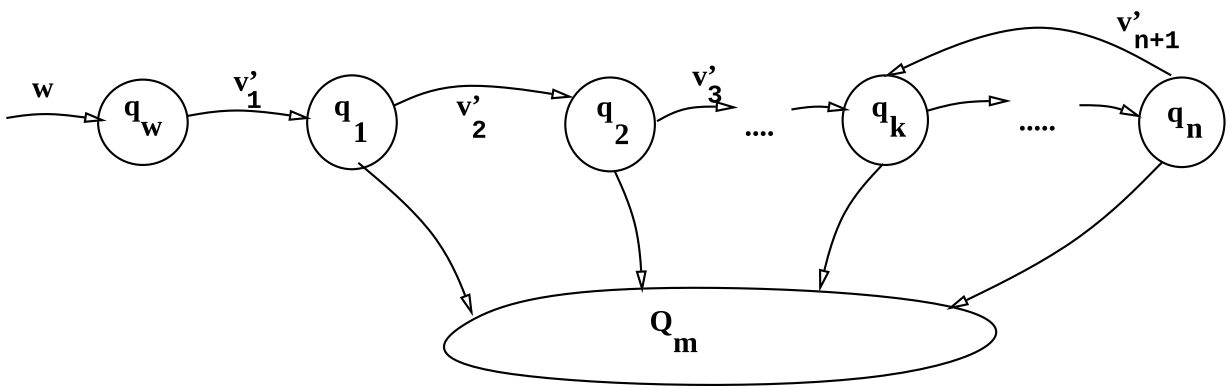

Figure 2: Cycle in a machine which is not finitely stabilizable.

This relationship follows from the fact that $pr(B)$ is controllable with respect to L and that S is topologically closed with respect to A^ω. And since there exists an $\alpha\prime \in S_f$ such that $\alpha\prime \notin B$, we have that $S_f \not\subseteq B$ as required for the contrapositive.
Q.E.D.

Theorem 3.2 Let $B \subseteq S$ be nonempty, M_S nonblocking, where $B, S \in DRAT$, and S is topologically closed with respect to A^ω. Then

for all $\epsilon > 0$, there exists:

nonblocking, complete supervisor f for P, such that:

$S_f \subseteq B$, and

$\forall u \in B, \exists v \in S_f : d(u, v) < \epsilon$,

if and only if

B is finitely stabilizable, and

$pr(B)$ is controllable with respect to $L = pr(S)$.

Proof:
This theorem follows directly from Lemma 3.2 and Lemma 3.3.

The proof of Theorem 3.2 is based on the specific strategy chosen for the supervisor. In this strategy, the supervisor limits the finite plant behavior to strings in the prefix of the desired behavior, which it can do since $pr(B)$ is controllable with respect to $pr(S)$. Then after a specified period of time without visiting a marked state, the supervisor forces the plant to take actions which cause the state trajectory to pass through a marked state in a finite number of events. If we define n as the size of the state set of the machine which recognizes B and m is the size of the state set of the machine which recognizes S, then from [11] determining the region of weak attraction is an $O(n)$ operation. From [9] we have that determining if $pr(B)$ is controllable with respect to $pr(S)$ is an $O(mn)$ operation. Combining these two operations with the size of the counter gives that constructing a supervisor is an $O(mn + 1/\epsilon)$ operation.

Lemma 3.4 demonstrates the relationship between the concepts of topological closure and finite stabilizability. A result of this relationship is that Theorem 3.2 can be considered as a generalization of Theorem 3.1 for some cases.

Lemma 3.4 Let $B \subseteq S \subseteq A^\omega$, S be topologically closed with respect to A^ω, and $B \in DRAT$, then

B is topologically closed with respect to S,

$\Rightarrow$

B is finitely stabilizable.

Proof: Let $M_B = \{Q, A, \delta_B, q_0, Q_m\}$ be a trim machine such that $S(M_B) = B$.

Definition of topologically closed

$\Rightarrow$

$\overline{B} \cap S = B$

$\Rightarrow \quad \{(S \text{ closed} \Rightarrow \overline{S} = S) \wedge (B \subseteq S)\}$

$\overline{B} = B$

$\Rightarrow \quad \{\text{definition of } \overline{B} = B\}$

all cycles in M_B go through Q_m

$\Rightarrow$

$Q - Q_m$ is acyclic

$\Rightarrow$

$\Omega_{M_B}(Q_m) = Q$ or that B is finitely stabilizable.

This last implication follows from the facts that $Q - Q_m$ is acyclic, that M_B is a trim machine and that $B \subseteq S$ [1]. This is readily observed by applying the algorithm for determining the region of weak attraction to any trim machine which accepts the desired language for a system where the desired language is topologically closed with respect to the plant language. **Q.E.D.**

That the converse is not true follows from the example presented earlier. For this example, we have:

$$S = a^\omega + a^* b a^\omega, a \in A_c, \text{ and } B = a^* b a^\omega.$$

In [16], the constraint language B is shown to be not topologically closed with respect to the plant language S; however, since,

$$P_0 = \{q_t\}, \text{ and } P_1 = \{q_t, q_i\} = Q,$$

one has that:

$$\Omega_{M_B}(Q_m) = Q, \text{ or that } B \text{ is finitely stabilizable.}$$

3.2 Nondeterministic Supervisors

A modification to the approach given in the previous section consists of using a supervisor which has some nondeterministic characteristics. The motivation for this type of supervisor stems from a desire to possibly include more of the desired behaviors from B in the closed loop response, S_f, without excessively increasing the complexity of the supervisor.

For a given error bound, $\epsilon > 0$, the basic supervisor is constructed in the same manner with the modifications outlined below. The machine M_c

is constructed in exactly the same manner but the supervisor feedback map is modified as given:

$$f(w)=\begin{cases} A'(q_w) & \text{if } \delta_c(q_{c0},w)\neq q_{cN-1}, \\ A'(q_w) & \text{if } \delta_c(q_{c0},w)=q_{cN-1} \text{ with probability } =1-p, \\ A'(q_w)-A_{bad}(q_w) & \text{if } \delta_c(q_{c0},w)=q_{cN-1} \text{ with probability } =p, \end{cases}$$

where $A'(q_w)$ and $A_{bad}(q_w)$ are as defined for the deterministic supervisor and $0 < p \leq 1$.

Lemma 3.5 can be used to demonstrate that $S_f \subseteq B$. The other aspects of the development follow directly from the proof for the deterministic case.

Lemma 3.5 Given any graph G of the form shown in Figure 3, with any initial state, and a transition function δ defined by

$$\delta(\sigma,q_i)=\begin{cases} q_1 & i=0, \text{ with probability } =1-x, \\ q_0 & i=0, \text{ with probability } =x, \\ q_{k-1} & i=k, \text{ with probability } =p, \\ q_k & i=k, \text{ with probability } =1-p, \\ q_{i-1} & i>0, \text{ with probability } =p, \\ q_{i+1} & i<k, \text{ with probability } =1-p-p_i', \\ q_i & i\neq k,0, \text{ with probability } =p_i', \end{cases}$$

where $0 < p \leq 1$, $0 \leq p_i' < 1$, $0 \leq x \leq 1$, and $p_i' + p \leq 1$, then any infinite trajectory has probability one of visiting state q_0 infinitely often.

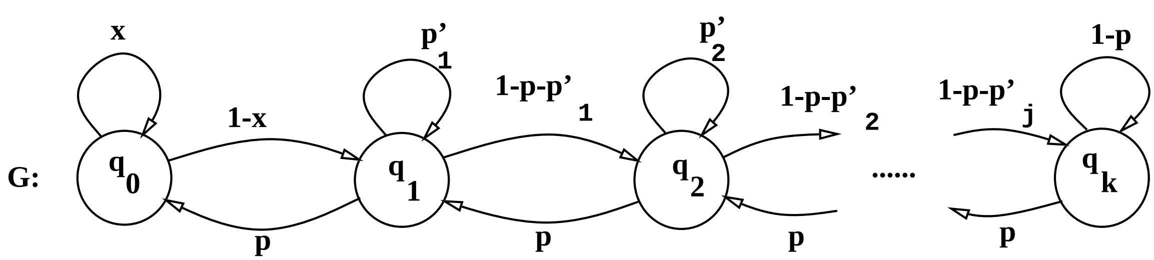

Figure 3: Modified Counting Machine.

Proof:
This problem is analogous to the one dimensional random walk problem as discussed in [6].
The proof of Lemma 3.5 can be directly derived from a more general lemma:

Lemma 3.6 For any graph of the form shown in Figure 3, with the transition function as defined in the Lemma 3.5 statement, then if state $q_j, 0 \leq j \leq k$, is visited infinitely often then every state $q_i, 0 \leq i \leq j$, is visited infinitely often.

Proof:
The proof is by induction on j.
Base Case: $j = 0$
Obvious, since q_0 is visited infinitely often.
Induction Hypothesis:
$\forall j : 0 \leq j < k$, if q_j is visited an infinite number of times, then $q_i, \forall i : 0 \leq i \leq j$, is visited infinitely often.
Induction Step:
$\forall j : 0 \leq j \leq k$, if q_j is visited an infinite number of times, then $q_i, \forall i : 0 \leq i \leq j$, is visited infinitely often.
Proof (of induction step):
If $j = 0$, then we are done. If $j \neq 0$, then $j > 0$. Since q_j is visited an infinite number of times and $p \neq 0$, the arc to q_{j-1} will be taken an infinite number of times [6]; hence, q_{j-1} will be visited an infinite number of times. By the induction hypothesis, since q_{j-1} is visited an infinite number of times and $j - 1 < k$, one has that q_i is visited an infinite number of times $\forall i : 0 \leq i \leq j - 1$. This fact combined with the fact that q_j is visited an infinite number of times completes the proof.
Q.E.D.
The proof of Lemma 3.5 follows directly since if the trajectory visits all states $q_i, 0 \leq i \leq j$, infinitely often it visits the single state, q_0, infinitely often.
Q.E.D.
From Lemma 3.5, one sees that the supervisor acts to cause the system to transition from P_i to $P_0 = Q_m$ infinitely often; hence $S_f \subseteq B$ as desired.

While there is a nonzero probability that the action taken by the deterministic and nondeterministic supervisors will be the same, there is also a nonzero probability that they will be different. In the case where the actions are different, the strings in the closed loop behavior of the system with the nondeterministic supervisor will strictly include the desired behavior of the system with the deterministic supervisor.

4 Examples

The requirements for a plant, described by ω-languages, to be controllable have been presented. These conditions can be applied to the case of plants described by Büchi automata. The following examples illustrate some of the ideas presented in this paper.

4.1 Example 1 Revisited

Recall the first example which considered a discrete event dynamic system $P = (L, S)$ with the event set $A = \{a, b\}$ [16]. The dynamics of the system are described by

$$S = a^\omega + a^* b a^\omega \text{ and } L = a^* + a^* b a^* (= pr(S)).$$

The uncontrollable events are described by $A_u = \{b\}$. The finite state machine shown in Figure 1 generates this language.

Consider the constraints imposed by $B = a^* b a^\omega \subset S$. B has been shown to be not topologically closed relative to S [16]; hence, there is not a nonblocking supervisor f such that $S_f = B$.

Using the extension presented in this paper, one can construct a complete nonblocking supervisor for the given plant which satisfies a relaxed constraint, i.e. that the desired and the actual behavior differ by some $\epsilon > 0$.

We have already shown that $pr(B)$ is controllable with respect to $pr(S)$.

Since B and S satisfy the hypothesis of Theorem 3.2, i.e. B is finitely stabilizable, as seen by constructing the region of weak attraction, P, and $pr(B)$ is controllable with respect to $pr(S)$, one can construct a supervisor, f, such that for a given $\epsilon > 0$, f is complete, nonblocking, $S_f \subseteq B$ and $\forall e \in B, \exists u \in S_f : d(e, u) < \epsilon$.

In order to construct such a supervisor, one first defines the machine

$$M_B = (Q, A, \delta, q_0, Q_m)$$

to be the machine which recognizes B, for this example

$$Q = \{i, t\}, A = \{a, b\}, q_0 = \{i\}, Q_m = \{t\},$$
$$\delta(i, a) = i, \delta(t, a) = t, \text{ and } \delta(i, b) = t.$$

The supervisor is constructed in the following manner:
Define $M_c = (Q_c, A, \delta_c, q_{c0}, Q_{mc})$ as in Theorem 3.2; and define the supervisor feedback map f by

$$f(w) = \begin{cases} \{a, b\} & \text{if } \{\delta(i, w) = i\} \text{ and } \{\delta_c(q_{c0}, w) \neq q_{cN-1}\}, \\ \{b\} & \text{if } \{\delta(i, w) = i\} \text{ and } \{\delta_c(q_{c0}, w) = q_{cN-1}\}, \\ \{a\} & \text{if } \delta(i, w) = t. \end{cases}$$

The behavior of this supervisor is illustrated by considering three cases.

1: If $e = a^k b a^\omega$ for $k < N$, then $f(e^j) = \{a, b\}$ for $j < k$, $f(e^j) = \{a, b\}$ for $j = k$ and $f(e^j) = \{a\}$ for $j > k$; consequently, $e_{j+1} \subseteq f(e^j), \forall j$, or the supervisor constructed above is such that $e \in S_f$.

2: If $e = a^k b a^\omega$ for $k \geq N$, then $f(e^j) = \{a, b\}$ for $j < N$, $f(e^j) = \{b\}$ for $j = N$ and $f(e^j) = \{a\}$ for $j > N$; consequently, $e_{j+1} \not\subseteq f(e^j), \forall j$, but there does exist a complete, nonblocking supervisor such that $\forall e \in B, \exists e' \in S_f : d(e, e') < \epsilon$, as can be seen by calculating the distance between the input string and the string in the closed loop behavior.

3: If $e = a^\omega$, then $f(e^j) = \{a, b\}$ for $j < N$, $f(e^j) = \{b\}$ for $j = N$ and $f(e^j) = \{a\}$ for $j > N$; consequently, $e_{j+1} \not\subseteq f(e^j), \forall j$, but there does exist a complete, nonblocking supervisor such that $\forall e \in B, \exists e' \in S_f : d(e, e') < \epsilon$, as can be seen by calculating the distance between the input string and the string in the closed loop behavior.

4.2 Vending Machine

Another example is a vending machine model. In this system, $VM = (A, Q, \delta, q_0)$, where

$$A = \{use, fails, repair, reject, accept\}, \text{ and } Q = \{q_0, q_1, q_2\},$$

with $A_c = \{use, repair\}$, and q_0 = machine wait, q_1 = machine broken, and q_2 = machine waits for inspection. The defined transitions are illustrated in Figure 4.

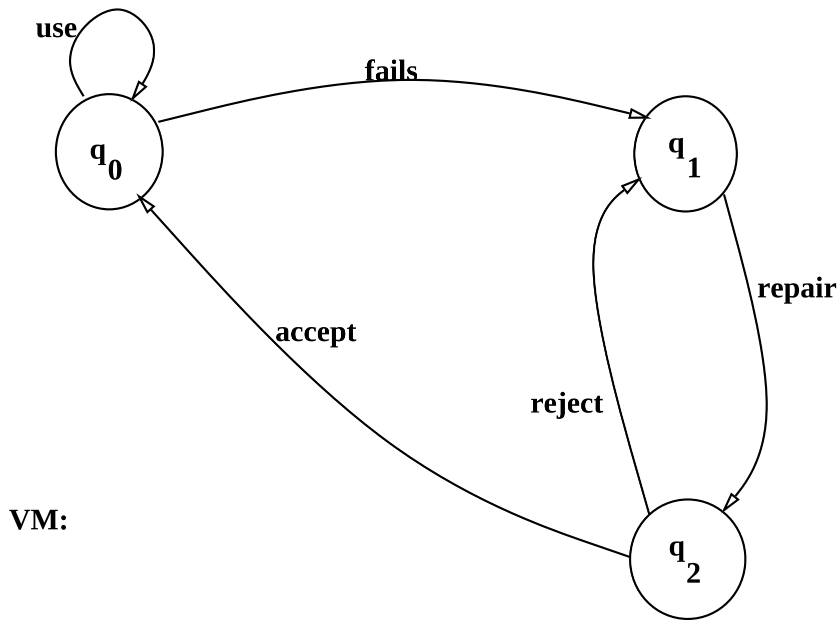

Figure 4: Vending Machine Model.

For this system,

$$S = (L(VM))^{\infty}, \text{ and } L = pr(S).$$

Since we do not want to have the machine in an infinite repair-reject cycle, the desired behavior is

$$B = S(VM) \text{ where } Q_m = q_0.$$

This infinite behavior gives the finite behavior of

$$pr(B) = L_m(VM).$$

For this example, a supervisor cannot be constructed. This result follows for both Ramadge's approach [16] and the extension presented here in Theorem 3.2, since B is not topologically closed relative to S and the region of weak attraction does not equal the entire state set.

However, if the option to buy a new machine is added in order to replace a machine which fails greater than k times, then a nonblocking supervisor can be constructed subject to the weakened constraint of $S_f \subseteq B$ and $\forall u \in B, \exists u' \in S_f : d(u, u') < \epsilon$. This option is symbolized by adding the event *replace* and $\delta(q_1, replace) = q_0$. The modified plant is illustrated in Figure 5. This additional event enlarges the region of weak attraction so that $P = \{q_0, q_1, q_2\}$; hence, a complete, nonblocking supervisor can be constructed, subject to the relaxed constraint regarding the distance between the desired behavior and the closed loop behavior.

For this modified example

$$\begin{aligned}
&S' = (L(VM2))^{\infty}, \\
&B = S(VM2) \text{ where } Q_m = q_0, \\
&pr(B) = L_m(VM2).
\end{aligned}$$

Observe that $pr(B)$ is controllable with respect to $L' = pr(S')$ and that B is finitely stabilizable. This observation provides that a complete supervisor can be constructed for a given $\epsilon > 0$.

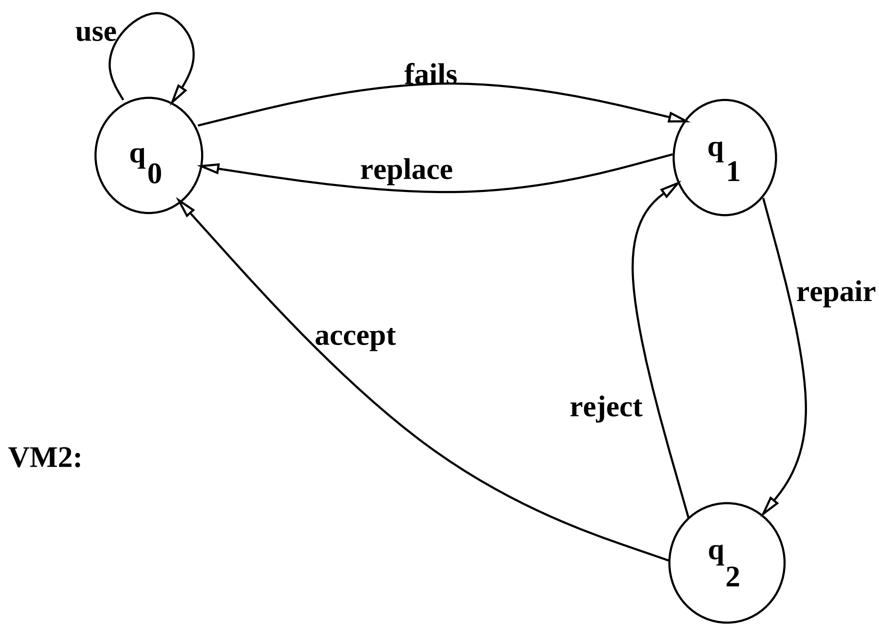

Figure 5: Modified Vending Machine Model.

For this supervisor, one constructs M_c such that $Q_c = \{q_0, \ldots, q_{N-1}\}$

where $(1/N) < \epsilon$ and δ_c such that

$$\delta_c(q_{ci}, \sigma) = \begin{cases} q_{c0} & \text{if } \delta(q_0, w) = q_0, \\ q_{ci+1} & \text{if } \{\delta(q_0, w) \neq q_0\} \text{ and } \{i + 1 < N - 1\}, \\ q_{cN-1} & \text{else,} \end{cases}$$

where $\delta_c(q_{c0}, w) = q_{ci}$.

And define the supervisor feedback map f by

$$f(w) = \begin{cases} \{use, fails\} & \text{if } \{\delta(q_0, w) = q_0\}, \\ \{replace, repair\} & \text{if } \{\delta(q_0, w) = q_1\} \text{ and } \{\delta_c(q_{c0}, w) \neq q_{cN-1}\}, \\ \{replace\} & \text{if } \{\delta(q_0, w) = q_1\} \text{ and } \{\delta_c(q_{c0}, w) = q_{cN-1}\}, \\ \{reject, accept\} & \text{if } \{\delta(q_0, w) = q_2\}. \end{cases}$$

If $e \in S$ such that M_c never enters q_{N-1}, then $e \in S_f$ since f never needs to disable any event.

If $e \in S$ such that M_c enters q_{N-1}, then $\exists e' \in S_f$ such that $d(e, e') < \epsilon$ since e and e' will agree on the prefixes up to the event where $\delta_c(q_{c0}, e^j) = q_{cN-1}$ and $e_{j+1} \notin f(e^j)$. And by the design of the supervisor, $|e^j| > N$; hence, $d(e, e') < 1/N < \epsilon$.

4.3 Non-subautomata Constraint

Another example is given by the finite state machine illustrated in Figure 6 with the language descriptions of

$$\begin{aligned} S &= ((a + b + c)^*(de)^*)^\omega, \text{ and} \\ B &= (a(c^*ba)^*c^*de)^\omega. \end{aligned}$$

In this example, the machine which recognizes the constraint language is not a strict subautomaton [3] of the machine which recognizes the plant language.

If $\{c, b, a\} \subset A_c$ and $Q_m = \{q_2\}$ for the machine which recognizes B, given in Figure 7, then the region of weak attraction P is equal to Q, and $pr(B)$ is controllable with respect to $pr(S)$; hence, there exists a complete, nonblocking supervisor as described in Theorem 3.2.

This supervisor is constructed in the following manner.
For this supervisor, one constructs M_c as in Theorem 3.2 and define the

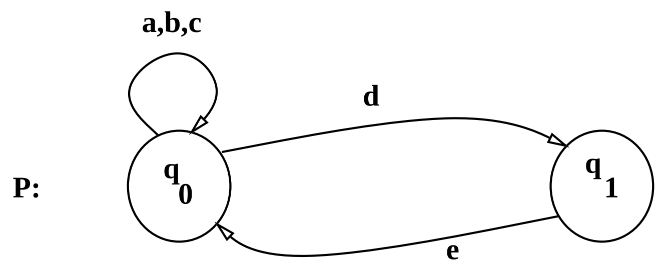

Figure 6: Plant for Non-subautomaton Example.

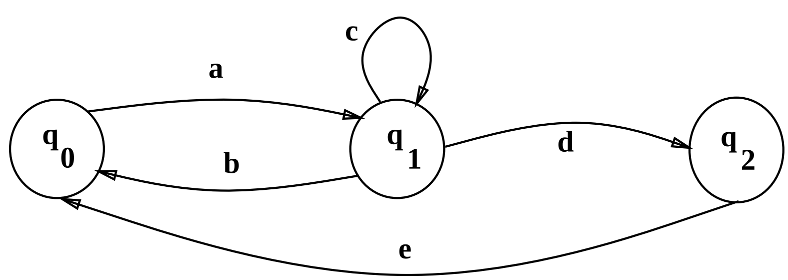

Figure 7: Constraint for Non-subautomaton Example.

supervisor feedback map f by

$$f(w) = \begin{cases} \{a\} & \text{if } \{\delta(q_0, w) = q_1\}, \\ \{b, c, d\} & \text{if } \{\delta(q_0, w) = q_1\} \text{ and } \{\delta_c(q_{c0}, w) \neq q_{cN-1}\}, \\ \{d\} & \text{if } \{\delta(q_0, w) = q_1\} \text{ and } \{\delta_c(q_{c0}, w) = q_{cN-1}\}, \\ \{e\} & \text{if } \{\delta(q_0, w) = q_2\}. \end{cases}$$

5 Conclusions

This paper has presented an extension to previous work concerning the control of discrete event systems which can be modeled with deterministic Büchi automata. The case where the language describing the desired behavior is not topologically closed with respect to the language describing the behavior of the plant is considered and conditions for the existence of a supervisor and the construction of one are given.

As demonstrated in the examples, situations do arise in which the language describing the desired behavior is not topologically closed with respect to the language describing the behavior of the plant, yet some form of a supervisor is still needed.

The resulting closed loop behavior does not necessarily exactly match the desired behavior, as in previous work, but can be shown to provide closed loop behavior that is within a specified error bound of the desired behavior.

Nondeterministic supervisors for plants are also considered. The closed loop behavior of such systems can be shown to include the behavior of the deterministic system.

In order to derive the results presented in this paper, the assumption that the plant behavior is topologically closed is made. Further work is needed to determine how weakening this condition affects the attainable closed loop behavior.

6 Acknowledgements

The authors would like to thank G. Wise and R. Kumar for beneficial discussions on this research and S.I. Marcus for reading the manuscript and providing helpful comments. The authors would also like to thank all the other reviewers for their helpful suggestions and comments.

References

[1] Y. Brave and M. Heymann. "On Stabilization of Discrete Event Processes". *Internation Journal Control*, 51:1101–1117, 1990.

[2] J.R. Buchi. "On a decision method in restricted second order arithmetic". In *Logic, Methodology, and Philosophy of Science: Proceedings 1960 International Congress*, pages 1–11. Stanford University Press, 1962.

[3] H. Cho and S.I. Marcus. "On the supremal languages of sublanguages that arise in supervisor synthesis problems with partial observation". *Mathematics of Control, Signals, Systems*, 1989.

[4] E.W. Dijkstra and C.S. Scholten. *Predicate Calculus and Program Semantics.* Springer-Verlag, New York, NY, 1990.

[5] S. Eilenberg. *Automata, Languages, and Machines Volume A.* Academic, New York, NY, 1974.

[6] W. Feller. *An Introduction to Probability Theory and Its Applications.* Wiley, 1966.

[7] Y.C. Ho. "Scanning the Issue". *Proceedings of IEEE*, 77(1):3–6, January 1989.

[8] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979.

[9] R. Kumar, V. Garg, and S.I. Marcus. "On Controllability and Normality of Discrete Event Dynamical Systems". In *Proceedings 27th Annual Allerton Conference*, August 1989.

[10] R. Kumar, V.K. Garg, and S.I. Marcus. On ω-controllability and ω-normality of deds. In *Proceedings American Control Conference*, pages 2905–2910, Boston, MA, June 1991.

[11] R. Kumar, V.K. Garg, and S.I. Marcus. Language stability and stabilizability of discrete event systems. *SIAM Journal Control Optimization*, 31(5):1294–1320, September 1993.

[12] R. McNaughton. "Testing and generating infinite sequences by a finite automaton". *Information and Control*, 9:521–530, 1966.

[13] D.E. Muller. "Infinite sequences and finite machines". In *Switching Circuit Theory and Logical Design: Proceedings 4th Annual Symposium*, pages 3–16, 1963.

[14] C.M. Ozveren, A.S. Willsky, and P.J. Antsaklis. "Stability and stabilizability of discrete event dynamic systems". *J. Association for Computing Machinery*, 38(3), July 1991.

[15] D. Perrin. "An introduction to finite automata on infinite words". In *Automata on Infinite Words*, volume 192 of *Lecture Notes in Computes Science*, 1984.

[16] P.J.G. Ramadge. "Some tractable supervisory control problems for discrete-event systems modeled by Büchi automata". *IEEE Transactions on Automatic Control*, 34:10–19, January 1989.

[17] P.J.G. Ramadge and W.M. Wonham. "Supervisory control of a class of discrete event processes". *SIAM Journal Control Optimization*, 25:206–230, January 1987.

[18] J.G. Thistle. "Control of infinite behaviour of discrete-event systems". Technical Report 9012, Systems Control Group, Department of Electrical Engineering, University of Toronto, January 1991.

[19] J.G. Thistle and W.M. Wonham. "On the synthesis of supervisors subject to ω-language specifications". In *Proceedings 22^{nd} Annual Conference Information Science Systems*, pages 440–444, Princeton, NJ, March 1988.

[20] S.D. Young, D. Spanjol, and V.K. Garg. "Control of Discrete Event Systems Modeled with Deterministic Büchi Automata". In *Proceedings American Control Conference*, pages 2814–2818, Chicago, IL, June 1992.