FORMULAS FOR CALCULATING SUPREMAL CONTROLLABLE AND NORMAL SUBLANGUAGES [1]

R. D. Brandt[2], V. Garg[3], R. Kumar[3], F. Lin[2], S. I. Marcus[3], and W. M. Wonham[4]

[2]Department of ECE, Wayne State University, Detroit, MI 48202
[3]Department of ECE, The University of Texas at Austin, Austin, TX 78712
[4]Department of EE, University of Toronto, Toronto, Ontario M5S 1A4, Canada

ABSTRACT

Supremal controllable and normal sublanguages have been shown to play an important role in supervisor synthesis. In this paper, we discuss the computation of supremal controllable and normal sublanguages. We derive formulas for both supremal controllable sublanguages and supremal normal sublanguages when the languages involved are closed. As a result, those languages can be computed without applying recursive algorithms. We also discuss the computational aspects of these formulas.

1. INTRODUCTION

In supervisory control, discrete event systems are modeled by controlled automata, and their behaviors described by the associated formal languages [1]- [3], [5]-[17] Control is exercised by a supervisor, whose action is to enable or disable events so that the controlled system generates some prespecified desired language. The subset of all events that can be selectively disabled by the supervisor is the set of *controllable* events. The supervisor may also be constrained to observe only events in a specified set of *observable* events. It has been shown that such a supervisor exists if and only if the language to be synthesized is both controllable and observable [6],[9],[12]. Hence, the formal synthesis problem for a supervisor is posed as one of synthesizing the largest possible controllable and observable sublanguage of a specified "desirable" or "legal" language. This largest possible sublanguage is not necessarily unique. Therefore, in order to make the synthesis problem well-defined, a slightly stronger version of observability, called *normality*, has been introduced [6],[9]. It has been shown in [9] that normality implies observability, and that a unique supremal controllable and normal sublanguage is guaranteed to exist. Hence, supremal controllable and normal sublanguages play an important role in supervisor synthesis.

Supremal controllable sublanguages were discussed in [16] and a recursive algorithm was developed to compute the supremal controllable sublanguage of a given

[1]This paper merges the results of Kumar, Garg, and Marcus[5] and Lin, Brandt, and Wonham[7] which were presented at the 27th Annual Allerton Conference on Communication, Control, and Computing, 1989. This work was supported in part by the Air Force office of Scientific Research under Grant AFOSR-86-0029, in part by the National Science Fondation under Grant ECS-8617860, and in part by the Natural Sciences and Engineering Research Council of Canada under Grant A-7399.

language. However, no formula for such languages has previously been derived. In this paper, we will be primarily concerned with languages that are prefix closed (every prefix of a string in the language is also contained in the language). We will derive formulas for supremal controllable as well as supremal normal sublanguages when the languages involved are closed. As a result, we can compute those languages without using recursive algorithms.

2. CONTROLLABILITY AND NORMALITY

Let Σ denote the (finite) set of *events* and Σ^* the set of all finite strings of events of Σ including the empty string ϵ. A subset $L \subseteq \Sigma^*$ is a *language*. The *closure* $\overline{L}$ of a language L is the set of all prefixes of strings in L. L is *closed* if $L = \overline{L}$. In this paper, we will assume the languages involved to be closed.

Let M be a fixed language over Σ, which represents "feasible" or "physically possible" behaviors. Let $B \subseteq M$ be a language representing "desirable" or "legal" behavior. Methods for computing B from a set of specifications of operating rules was discussed in [11]. Our task is to synthesize a supervisor when B is given. It is important to take into account the possibility that certain events cannot be disabled by the supervisor or that certain events may not be observable to the supervisor. Therefore we must introduce the concepts of controllability and normality [1],[6],[9],[12]. It is assumed that some events are controllable, in the sense that their occurrence can be prevented, and some events are observable, in the sense that their occurrence can be observed. Thus, Σ is partitioned as

$$\Sigma \;=\; \Sigma_c \,\dot{\cup}\, \Sigma_{uc} \;=\; \Sigma_o \,\dot{\cup} \Sigma_{uo}$$

where Σ_c denotes the set of *controllable events*, Σ_{uc} the set of *uncontrollable events*, Σ_o the set of *observable events*, and Σ_{uo} the set of *unobservable events*.

A language L is said to be controllable, if there does not exist any uncontrollable event $\sigma \in \Sigma_{uc}$ which might possibly lead to an illegal behavior, i. e., $s\sigma \in L$ for all s such that $s\sigma \in M$.

Definition 1 A language $L \subseteq M$ is *controllable* with respect to M if and only if, for all $s \in L$ and $\sigma \in \Sigma_{uc}$, if $s\sigma \in M$, then $s\sigma \in L$.

A notion of observability has been defined in [1], [6], [9], but a stronger condition, called normality [6], [9], will be used in this paper. To define normality, let us first define $P : \Sigma^* \to \Sigma_o^*$ to be

$$P\epsilon \;=\; \epsilon$$

$$P\sigma \;=\; \epsilon \;,\; \sigma \in \Sigma - \Sigma_o$$

$$P\sigma \;=\; \sigma \;,\; \sigma \in \Sigma_o$$

$$P(s\sigma) \;=\; (Ps)(P\sigma),\;\; s \in \Sigma^*,\;\; \sigma \in \Sigma$$

That is, P is a projection whose effect on a string $s \in \Sigma^*$ is just to erase the elements, σ, of s that do not belong to Σ_o.

A language L is said to be normal if one can check whether a string is legal, by checking whether its projection belongs to PL. In other words, information on occurrences of unobservable events is not needed in determining the legal status of a string.

Definition 2 A language $L \subseteq M$ is *normal* with respect to M if and only if, for all $s \in M$, $s \in L$ if and only if $Ps \in PL$.

The following theorem shows that the class of controllable and normal languages are algebraically well-behaved, in the sense that the supremal controllable and supremal normal sublanguages of any given language exist and are unique [9], [16].

Theorem 1 Let $B \subseteq M$ and consider the classes of languages

$$C(B) = \{L \subseteq B : L \text{ is closed and controllable}\}$$

$$N(B) = \{L \subseteq B : L \text{ is closed and normal}\}$$

Then $C(B)$ and $N(B)$ are nonempty and closed under arbitrary unions. Therefore, $\sup C(B)$ exists in $C(B)$ and $\sup N(B)$ exists in $N(B)$.

3. MAIN RESULTS

Let us first discuss $\sup C(B)$. A recursive algorithm to compute $\sup C(B)$ (for languages that may or may not be closed), has been presented in [16] but no formula has been obtained. When languages are closed we can derive a formula for $\sup C(B)$. To prove the formula, we need the following lemma, which states that, in the definition of controllability, $\sigma \in \Sigma_{uc}$ can be replaced by a string $u \in \Sigma_{uc}^*$.

Lemma 1 Let $L \subseteq M$; then L is controllable with respect to M if and only if $L\Sigma_{uc}^* \cap M \subseteq L$, where Σ_{uc}^* denotes the set of finite strings of uncontrollable events.

Proof: (If): Assume that $L\Sigma_{uc}^* \cap M \subseteq L$. Then

$$L\Sigma_{uc} \cap M \subseteq L\Sigma_{uc}^* \cap M \subseteq L$$

$$\{\text{since } \Sigma_{uc} \subseteq \Sigma_{uc}^*\}$$

Thus L is controllable.

(Only if): Assume now that L is controllable. We prove that

$$L\Sigma_{uc}^n \cap M \subseteq L$$

by induction on n, where Σ_{uc}^n denotes the set of strings of uncontrollable events of lenght n. Clearly it is true for $n = 0$. Suppose that it is true for $n = i$. Then

$$L\Sigma_{uc}^{i+1} \cap M$$

$$\subseteq L\Sigma_{uc}^{i}\Sigma_{uc} \cap M$$

$$\subseteq L\Sigma_{uc}^{i}\Sigma_{uc} \cap M\Sigma_{uc} \cap M$$

$$\{\text{since } L\Sigma_{uc}^{i}\Sigma_{uc} \cap M \subseteq M\Sigma_{uc}\}$$

$$\subseteq (L\Sigma_{uc}^{i} \cap M)\Sigma_{uc} \cap M$$

$$\subseteq L\Sigma_{uc} \cap M$$

$$\{\text{by induction hypothesis}\}$$

$$\subseteq L$$

Thus $L\Sigma_{uc}^* \cap M \subseteq L$. $\square$

The formula for $\sup C(B)$ involves the application of a mapping from Σ^* to Σ^* which has the effect of cutting off the "uncontrollable tail" of strings. Let $D_{uc} : \Sigma^* \to \Sigma^*$ be defined by

$$D_{uc}(s) = \text{the longest prefix of } s \text{ whose last event is controllable}$$

If s contains no controllable events, then $D_{uc}(s) = \epsilon$. Intuitively, $M - B$ consists of the illegal strings and $D_{uc}(M - B)$ consists of the strings that have the potential of becoming illegal (by adding uncontrollable events). Therefore, they have to be removed from B. To guarantee that the resulting language is closed, Σ^* must be concatenated to $D_{uc}(M - B)$. Thus, the formula for $\sup N(B)$ is presented in the following theorem.

Theorem 2

$$\sup C(B) = B - D_{uc}(M - B)\Sigma^*$$

Proof: ($\supseteq$): Let the right hand side be denoted H. First, we show that $\sup C(B) \supseteq H$. Clearly H is closed and contained in B. Therefore, we only need to show that it is controllable. Consider $s \in \Sigma^\star, \sigma \in \Sigma_{uc}$; and let $s' \in \Sigma^\star$ be the longest prefix of s with a controllable event at its end (s' is the empty string if $s \in \Sigma_{uc}^\star$) such that

$$(s \in H) \wedge (s\sigma \in M)$$

$$\Rightarrow (s \in B) \wedge (s \notin D_{uc}(M - B)\Sigma^\star) \wedge (s\sigma \in M)$$

$$\{\text{since } H = B - D_{uc}(M-B)\Sigma^\star\}$$

$$\Rightarrow (s \in B) \wedge (s' \notin D_{uc}(M-B)) \wedge (s\sigma \in M)$$

$$\{\text{By definition of } (\cdot)\Sigma^\star\}$$

$$\Rightarrow (s \in B) \wedge (s\sigma \notin (M-B)) \wedge (s\sigma \in M)$$

$$\{\text{By definition of } D_{uc}(\cdot)$$

$$\Rightarrow (s \in B) \wedge (s\sigma \in B)$$

$$\{(s\sigma \in M) \wedge (s\sigma \notin (M-B)) \Rightarrow s\sigma \in B\}$$

$$\Rightarrow (s \in B) \wedge (s\sigma \in B) \wedge (s\sigma \notin D_{uc}(M-B))$$

$$\{t \in D_{uc}(\cdot) \Rightarrow \text{no uncontrollable event at the end of } t\}$$

$$\Rightarrow (s \in B) \wedge (s\sigma \in B) \wedge (s\sigma \notin D_{uc}(M-B)\Sigma^\star)$$

$$\{D_{uc}(M-B) \text{ is closed (since M, B are closed)} \Rightarrow \text{no prefix of } s\sigma \in D_{uc}(M-B)\}$$

$$\Rightarrow (s\sigma \in H)$$

$\{H = B - D_{uc}(M-B)\Sigma^\star\}$ Thus H is controllable with respect to M.
$(\subseteq)$: Next we show that $\sup C(B) \subseteq H$. Since $\sup C(B) \subseteq B$, we only need to show

$$D_{uc}(M-B)\Sigma^* \cap \sup C(B) = \emptyset$$

Assume the contrary. Then there exists $s \in D_{uc}(M-B)$ and $t \in \Sigma^*$ such that

$$((st) \in D_{uc}(M-B)\Sigma^*) \wedge ((st) \in \sup C(B))$$

$$\Rightarrow (s \in D_{uc}(M-B)) \wedge (s \in \sup C(B))$$

$$\{\text{since supC(B) is closed}\}$$

$$\Rightarrow ((\exists u \in \Sigma_{uc}^*) \text{ s. t. } (su \in M-B)) \wedge (s \in \sup C(B))$$

$$\{\text{by definition of } D_{uc}(\cdot)\}$$

$$\Rightarrow (\exists u \in \Sigma_{uc}^*) \text{ s. t. } (su \notin B) \wedge (su \in M) \wedge (s \in \sup C(B))$$

$$\{\text{by expanding } su \in (M-B)\}$$

$$\Rightarrow (\exists u \in \Sigma_{uc}^*) \text{ s. t. } (su \notin B) \wedge (su \in \sup C(B))$$

$$\{\text{since from Lemma 1 }, (su \in M) \wedge (s \in \sup C(B)) \Rightarrow (su \in \sup C(B))\}$$

Contradiction! $\{(su \notin B) \wedge (su \in \text{supC(B)})\}$ □

Let L_1/L_2 denote the quotient of language L_1 with respect to L_2 defined as:

$$L_1/L_2 = \{s \in \Sigma^* : \exists t \in L_2 \text{ such that } st \in L_1\}$$

Then the above formula for supC(B) can equivalently be written as:

$$\sup C(B) = B - ((M - B)/\Sigma_{uc}^*)\Sigma^*$$

The above expressions for $\sup C(B)$ can be used to develop an algorithm for generating the language $\sup C(B)$. Let m and n be the number of states in the minimal finite state machine (FSM) realizations of the languages M and B respectively. Then the FSM that generates $\sup C(B) = B - D_{uc}(M - B)\Sigma^*$ is constructed in $O(mn^2)$ time, because the FSM that generates the language $(M - B)$ has $O(mn)$ states, the FSM for generating $D_{uc}(M - B)\Sigma^*$ is constructed in $O(mn)$ time and finally the FSM that generates $B - D_{uc}(M - B)\Sigma^*$ has $O(mn^2)$ states. Thus, the computational complexity of the algorithm that computes supC(B) using the above formula, is $O(mn^2)$.

Next, we present the formula for supN(B). Intuitively, $M - B$ are illegal strings and $P^{-1}P(M-B)$ are the strings that look the same as illegal strings. Therefore, they have to be removed from B. To guarantee that the resulting langauge is closed, Σ^* must be concatenated to $P^{-1}P(M - B)$. Thus, the formula for $\sup N(B)$ is presented in the following theorem.

Theorem 3

$$\sup N(B) = B - P^{-1}P(M - B)\Sigma^*$$

Proof: ($\supseteq$): Denote the right hand side by H. First, we show that $\sup N(B) \supseteq H$. Clearly H is closed and contained in B. Therefore, we only need to show that it is normal. Assume the contrary; then there exists $s \in \Sigma^*$ such that

$$(s \in M) \wedge (Ps \in PH) \wedge (s \notin H)$$

$$\{\text{by definition of normality}\}$$

$$\Rightarrow (s \in M) \wedge ((\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \in H)) \wedge (s \notin H)$$

$$\{\text{by expanding } Ps \in PH\}$$

$$\Rightarrow (s \in M) \wedge (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge ((s' \in B) \wedge (s' \notin P^{-1}P(M - B)\Sigma^*)) \\ \wedge((s \notin B) \vee (s \in P^{-1}P(M - B)\Sigma^*))$$

$$\{\text{since } H = B - P^{-1}P(M - B)\Sigma^*\}$$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \notin P^{-1}P(M - B)\Sigma^*) \\ \wedge((s \in M - B) \vee (s \in P^{-1}P(M - B)\Sigma^*))$$

$$\{\text{since } (s \in M) \wedge (s \notin B) \Rightarrow s \in (M-B)\}$$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \notin P^{-1}P(M-B)\Sigma^*) \wedge (s \in P^{-1}P(M-B)\Sigma^*)$$

$$\{\text{since } (M-B) \subseteq P^{-1}P(M-B)\Sigma^*\}$$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \notin P^{-1}P(M-B)\Sigma^*) \wedge (s' \in P^{-1}P(M-B)\Sigma^*)$$

$$\{\text{since } Ps' = Ps\}$$

Contradiction! $\{(s' \notin P^{-1}P(M-B)\Sigma^*) \wedge (s' \in P^{-1}P(M-B)\Sigma^*)\}$
($\subseteq$): Next we show that sup$N(B) \subseteq H$. Since sup$N(B) \subseteq B$, we only need to show

$$P^{-1}P(M-B)\Sigma^* \cap \text{sup}N(B) = \emptyset$$

Assume the contrary; then there exists $s \in \Sigma^*$ such that

$$(s \in P^{-1}P(M-B)) \wedge (s \in \text{sup}N(B))$$

$$\Rightarrow ((\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \in M-B)) \wedge (s \in \text{sup}N(B))$$

$$\{\text{by definition of } P^{-1}P(\cdot)\}$$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge ((s' \notin B) \wedge (s' \in M)) \wedge (s \in \text{sup}N(B))$$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps = Ps') \wedge (s' \notin B) \wedge (s' \in \text{sup}N(B))$$

{supN(B) is normal, so $(Ps = Ps') \wedge (s' \in \text{sup}N(B)) \wedge (s' \in M) \Rightarrow (s' \in \text{sup}N(B))$}
Contradiction! $\{(s' \notin B) \wedge (s' \in \text{sup}N(B))\}$ □

The above formula for the supremal normal sublanguage of B with respect to M can be used to develop an algorithm to compute sup$N(B)$. We can show that the computational complexity of this algorithm is exponential in the product of the number of states in the FSM realizations of the languages M and B. This is so because the FSM that generates $P(M-B)$ is *non-deterministic* and the algorithm that transforms a non-deterministic FSM into a deterministic FSM is of exponential complexity [4].

Finally, we prove the following theorem, which shows that we can decompose the computation of sup$CN(B)$ and apply the previous two formulas to compute sup$CN(B)$.

Theorem 4

$$\text{sup}CN(B) = M \cap P^{-1}\text{sup}C_p(P\text{sup}N(B))$$

where for $D \subseteq PM$, $C_p(D)$ is defined to be:

$$C_p(D) = \{L \subseteq D : L \text{ is controllable w.r.t. } PM\}$$

Proof: $(\supseteq)$: Denote the right hand side by H. First, we show that $\sup CN(B) \supseteq H$. Clearly H is closed and normal. Also, $H \subseteq B$, for

$$H \subseteq M \cap P^{-1} P \sup N(B) (= \sup N(B)) \subseteq B$$

It can be shown that H is controllable (Proposition 3.1 of [8]). Thus, $\sup CN(B) \supseteq H$.
$(\subseteq)$: Next we show that $\sup CN(B) \subseteq H$. Since $\sup CN(B) = M \cap P^{-1} P \sup CN(B)$, We only need show

$$P \sup CN(B) \subseteq \sup C_p(P \sup N(B))$$

This is true because first,

$$\sup CN(B) \subseteq \sup N(B) \Rightarrow P \sup CN(B) \subseteq P \sup N(B)$$

and secondly, $P \sup CN(B)$ is controllable w.r.t. PM. Assume the contrary; then there exist $s \in \Sigma^*$ and $\sigma \in \Sigma$ such that

$$(s \in P\sup CN(B)) \wedge (\sigma \in \Sigma_{uc} \cap \Sigma_o) \wedge (s\sigma \in PM) \wedge (s\sigma \notin P\sup CN(B))$$

$$\Rightarrow (s \in P\sup CN(B)) \wedge (\sigma \in \Sigma_{uc} \cap \Sigma_o) \wedge ((\exists s' \in \Sigma^*) \text{ s. t. } (Ps' = s) \wedge (s'\sigma \in M)) \wedge (s\sigma \notin P\sup CN(B))$$

$\{$by expanding $s\sigma \in PM\}$

$$\Rightarrow (s' \in \sup CN(B)) \wedge (\sigma \in \Sigma_{uc} \cap \Sigma_o) \wedge ((\exists s' \in \Sigma^*) \text{ s. t. } (Ps' = s) \wedge (s'\sigma \in M)) \wedge (s\sigma \notin P\sup CN(B))$$

$\{$since $\sup CN(B)$ is normal, $(Ps' = s) \wedge (s \in P\sup CN(B) \Rightarrow (s' \in \sup CN(B))\}$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps' = s) \wedge (s'\sigma \in \sup CN(B)) \wedge (s\sigma \notin P\sup CN(B))$$

$\{$since $\sup CN(B)$ is controllable, $(s'\sigma \in M) \wedge (s' \in \sup CN(B)) \Rightarrow (s'\sigma \in \sup CN(B))\}$

$$\Rightarrow (\exists s' \in \Sigma^*) \text{ s. t. } (Ps' = s) \wedge (P(s'\sigma) \in P\sup CN(B)) \wedge (s\sigma \notin P\sup CN(B))$$

But $P(s'\sigma) = s\sigma$, thus we obtain a contradiction. $\{(s\sigma \in P\sup CN(B)) \wedge (s\sigma \notin P\sup CN(B))\}$ $\square$

The formula that results from the theorem is

$$\sup CN(B) = M \cap P^{-1}(P\sup N(B) - D_{uc}(PM - P\sup N(B))\Sigma_o^*)$$

where

$$\sup N(B) = B - P^{-1}P(M - B)\Sigma^*$$

The above formula for the supremal controllable and normal sublanguage of B with respect to M can be used to construct an algorithm to compute $\sup CN(B)$. We can show that the computational complexity of this algorithm is exponential in the product of the number of states in the FSM realizations of the languages M and B.

4. CONCLUSION

In this paper, we have shown that when the legal language is closed (contains all its prefixes), there are simple formulas for the supremal controllable sublanguage and the supremal normal sublanguage of the legal language. We have discussed the computational aspects of the algorithms that can be constructed using these formulas.

The closure condition is not restrictive in investigating "safety" properties of systems (e.g. a system should never leave a desired region), because in such cases, legal languages are in fact closed. In investigating "liveness" properties of systems (e.g. a system should eventually accomplish some task from a prescribed set of completed or terminal tasks), the closure condition may not be satisfied. In such cases, the recursive algorithm developed in [16] is applicable, as would be a recursive application of the formula we have given for closed languages.

REFERENCES

[1] R. Cieslak, C. Desclaux, A. Fawaz and P. Varaiya, 1988. Supervisory control of discrete-event processes with partial observations. *IEEE Transactions on Automatic Control 33(3)*, pp. 249-260.

[2] H. Cho and S. I. Marcus. Supremal and maximal sublanguages arising in supervisor synthesis problems with partial observations. to appear in *Mathematical Systems Theory*.

[3] H. Cho and S. I. Marcus. On supremal languages of class of sublanguages that arise in supervisor synthesis problems with partial observations. *Math. Control Signal Systems 2*, pp. 47-69.

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

[5] R. Kumar, V. Garg and S. I. Marcus, 1989. Supervisory Control of Discrete Event Systems: Supremal Controllable and Observable Languages. to appear in the *Proceedings of 1989 Allerton Conference.*

[6] F. Lin, 1987. On Controllability and Observability of Discrete Event Systems. *Ph. D. Thesis*, Department of Electrical Engineering, University of Toronto.

[7] F. Lin, R. D. Brandt, and W. M. Wonham, 1989. A Note on Supremal Controllable and Normal Sublanguages. to appear in the *Proceedings of 1989 Allerton Conference.*

[8] F. Lin and W. M. Wonham, 1988. Decentralized supervisory control of discrete-event systems. *Information Sciences, 44(3)*, pp. 199-224.

[9] F. Lin and W. M. Wonham, 1988. On observability of discrete event systems. *Information Sciences, 44(3)*, pp. 173-198.

[10] F. Lin and W. M. Wonham, 1988. Decentralized control and coordination of discrete event systems with partial observation. *Systems Control Group Report No. 8909*, Department of Electrical Engineering, University of Toronto.

[11] F. Lin, A. F. Vaz and W. M. Wonham, 1988. Supervisor specification and synthesis for discrete event systems. *International Journal of Control, 48(1)*, pp. 321-332.

[12] P. J. Ramadge and W. M. Wonham, 1987. Supervisory control of a class of discrete event processes. *SIAM J. Control and Optimization, 25(1)*, pp. 206-230.

[13] P. J. Ramadge and W. M. Wonham, 1988. Modular supervisory control of discrete event systems. *Mathematics of Control, Signal and Systems, 1(1)*, pp. 13-30.

[14] P. J. Ramadge and W. M. Wonham, 1989. The control of Discrete Event Systems. *Proceedings of IEEE, 77(1)*, pp. 81-98.

[15] W. M. Wonham, 1988. A control theory for discrete-event systems. *Advanced Computing Concepts and Techniques in Control Engineering* (M. J. Denham and A. J. Laub Ed.), NATO ASI Series, F47, Springer-Verlag, Berlin, pp. 129-169.

[16] W. M. Wonham and P. J. Ramadge, 1987. On the supremal controllable sublanguage of a given language. *SIAM J. Control and Optimization, 25(3)*, pp. 635-659.

[17] H. Zhong, 1987. Control of discrete-event systems: decentralized and hierarchical control. *M. A. Sc. Thesis*, Department of Electrical Engineering, University of Toronto.