第10卷第3期 控制理论与应用 Vol. 10, No. 3
1993年6月 CONTROL THEORY AND APPLICATIONS Jun., 1993

Optimization of the Supervisor for a Given Discrete Event System

JIANG Zhiping and WU Zhiming
(Department of Automatic Control, Shanghai Jiao Tong University·Shanghai, 200030, PRC)

Abstract: Based on the supervisor model proposed by Ramadge and Wonham, a new optimization design idea is introduced which makes the number of states of a supervisor comparatively minimum. A P-algorithm with full proof is suggested. Three examples are given.

Key words: discrete event systems; control; generator; optimization; algorithm

1 Introduction

A discrete-event system, according to the model established by Ramadge and Wonham[1], can be expressed using a so-called generator $G=(Q,\Sigma,\delta,q_0,Q_m)$, where Q is a state set, Σ an alphabet, $\delta:\Sigma\times Q\to Q$ a partial function (pfn), q_0 an initial state and $Q_m\subseteq Q$ a marker state set. G is generally supposed to be deterministic: i) q_0 is unique and ii) for all $\sigma\in\Sigma$, $q\in Q$, $\delta(\sigma,q)=q'\in Q$ is unique or $\delta(\sigma,q)$ is undefined.

Let $\Sigma_c\subseteq\Sigma$ and $\Sigma_u\subseteq\Sigma$ represent the controllable and uncontrollable alphabet set in Σ respectively with $\Sigma_c\cup\Sigma_u=\Sigma$ and $\Sigma_c\cap\Sigma_u=\varnothing$. Introducing a control mechnism Γ into G, we obtain a controlled discrete event system (CDES) $G_c=(Q,\Gamma\times\Sigma,\delta_c,q_0,Q_m)$, where, Q, Σ, q_0 and Q_m are as defined before; $\Gamma=\{\gamma:\gamma:\Sigma\to\{0,1\}$ and $(\forall\ \sigma)\sigma\in\Sigma_u,\gamma(\sigma)=1\}$; and $\delta_c:\Gamma\times\Sigma\times Q\to Q$(pfn) is defined only if both $\delta(\sigma,q)$ is defined and $\gamma(\sigma)=1$.

Let Σ^* denote the set of all finite strings composed of elements of Σ. The above δ (and correspondingly δ_c) can be extended to be a partial function of $\Sigma^*\times Q\to Q$ (and $\Gamma\times\Sigma^*\times Q\to Q$) still written δ (and δ_c), and for all $s\in\Sigma^*$ and $\sigma\in\Sigma$, $\delta(s\sigma,q)=\delta(\sigma,\delta(s,q))$ iff $\delta(s,q)=q'$ and $\delta(\sigma,q')$ are all defined (correspondingly, $\delta_c(s\sigma,q)=\delta_c(\sigma,\delta(s,q))$ iff $\delta_c(s,q)=q'$ and $\delta_c(\sigma,q')$ are all defined).

For a given G, the language produced by G is

$$L(G)=\{w:w\in\Sigma^*\text{ and }\delta(w,q_0)\text{ is defined}\},$$

and the language marked by G is

$$L_m(G)=\{w:w\in L(G)\text{ and }\delta(w,q_0)\in Q_m\}.$$

In general, there exist in G some state(s) we could not arrive from q_0 but of interst is a generator whose states are all reachable from q_0. A generator so defined is said to be accessible[2].

For a given G_c, the task to design its controllor is i) to determine the control signals to G_c for

Manuscript received June 17, 1991, revised July 15, 1992.

every step and ii) to make decison of how to move from one state to another. A supervisor is so called because such a controllor has the property that T supervises over G_c. Formally, a supervisor T is defined to be a pair:

$$T = (S, f)$$

where, $S=(X,\Sigma,\xi,x_0,X_m)$ is a deterministic automaton with a state set X, an alphabet Σ, a state transition (partial) function $\xi:\Sigma\times X\rightarrow X$, an initial state x_0 and a marker state set $X_m\subseteq X$; $f:X\rightarrow\Gamma$ is a (total) function mapping every state x of X into a corresponding control γ.

Coupling the T to G_c, we will get a supervised discrete event system (SDES), written T/G_c:

$$T/G_c = (X \times Q, \Sigma, \xi \times \delta_c, (x_0,q_0), X_m \times Q_m),$$

where, $\xi\times\delta_c:\Sigma\times X\times Q\rightarrow X\times Q$(pfn) is defined according to $(\xi\times\delta_c)(\sigma, x, q)=(\xi(\sigma,x), \delta_c(f(x)),\sigma, q))$.

Now let $L_c(T/G_c)=L_m(G_c)\cap L(T/G_c)$. By [1], T is proper iff

$$\overline{L}_m(T/G_c) = \overline{L}_c(T/G_c) = L(T/G_c),$$

here

$$\overline{L}_m(T/G_c): = \{w:w \in \Sigma^*\text{and } (\exists\ u)u \in \Sigma^*,\ wu \in L_m(T/G_c)\},$$

and $\overline{L}_m(T/G_c)$ could be similarly defined.

For a give G_c, we can always obtain T through efective computation[3] although T may be empty. The main contribution of this paper is to give a P-algorithm R to reduce the state number of T and to get a more efficient supervisor $T^N=(S^N, f^N)$ which will be proved having the equivalent control effect to G_c. The algorithm is superior to the one proposed in [4] in that the latter has been proved to have the exponential complexity w. r. t. the state number $\|X\|$. In addition, we will also indicate through an example that the R can only ensure the T^N is optimal w. r. t. R, which means there exist some circumstances under which R do not ensure T^N to be real optimal.

2 Equivalent Supervisors

For two supervisors T, T' given, we call T and T' equivalent if $L(T/G_c)=L(T'/G_c)$. Formally, let $T=(S,f)$ and $T'=(S',f')$ be two supervisors with

$$S = (X,\Sigma,\xi,x_0,X_m),\quad f:X\rightarrow\{0,\ 1\}^\Sigma,$$
$$S' = (X',\Sigma,\xi',x'_0,X'_m),\quad f':X'\rightarrow\{0,\ 1\}^\Sigma$$

Definition 2.1 If there exists π: $X\rightarrow X'$ satisfying

a) $\pi:X\rightarrow X'$ is surjective;

b) $\pi(x_0)=x'_0$ and $X_m=\pi^{-1}(X'_m)$;

c) $\xi'(\sigma,\pi(x))=\pi(\xi(\sigma,x))$ only if $\xi(\sigma,x)$ is defined; and

d) $f'(\pi(x))=f(x)$

then π is called to be a projection from T to T', written $\pi:T\rightarrow T'$ and S' is called a quotient of S under π.

It has been proved in [1] that if there exists such a π between T and T' and T is proper, then T' is proper, too. This implies that for a given proper supervisor T and another supervisor T', if we could find out a $\pi: T \to T'$, then, T' must also be a proper supervisor superior (or at least equal) to T in state number.

Given $T=(S,f)$, we will try to get $T'=(S',f')$ using the following construct 2. 1.

Construct 2. 1 Let $x_i, x_j \in X$ satisfy

i) $x_i, x_j \in X_m$ or x_i, $x_j \in X - X_m$;

ii) $(\forall\ \sigma)\sigma \in \Sigma, \xi(\sigma, x_i) \cup \xi(\sigma, x_j) \in \{\varnothing, x_i, x_j\}$ or $\xi(\sigma, x_i) = \xi(\sigma, x_j)$;

iii) $f(x_i) := f(x_j)$ is allowed.

Then, let x_j merge into x_i and process x_i, x_j and $f(x_j)$ as follows

a) $(\forall\ \sigma)\sigma \in \Sigma$ and $(\exists\ x)x \in \{x_i, x_j\}$, if $\xi(\sigma, x) \in \{x_i, x_j\}$ then let $\xi'(\sigma, x_i) = x$;

b) $(\forall\ \sigma)\sigma \in \Sigma$, $(\exists\ x)x \in \{x_i, x_j\}$ and $(\exists\ y)y \notin \{x_i, x_j\}$, if $\xi(\sigma, x) = y$ then let $\xi'(\sigma, x_i) = y$;

c) $(\forall\ x)x \notin \{x_i, x_j\}$ and $(\exists\ \sigma)\sigma \in \Sigma$, if $\xi(\sigma, x) \in \{x_i, x_j\}$ then let $\xi'(\sigma, x) = x_i$;

d) $f'(x_i) = f(x_j)$;

e) $x_j := $ "$*$".

The last process above is to ensure that the state having been merged will never be meet again in later processes. Intuitively, two states x_i and x_j can be merged only if they have no common event, say σ, going to different states other than $\{x_i, x_j\}$ and have the compatible control f. We have

Proposition 2. 1 There exists a projection between T and T' derived from T through construct 2. 1.

Proof Let $\pi: T \to T'$. We prove if there are two states x_i, x_j in S satisfying the conditions i)~iii) in construct 2. 1, the conditions a)~d) in definition 2. 1 could be satisfied through construct 2. 1.

1) Let $X' = X - \{x_j\}$ and $\pi(x) = x$ if $x \in X - \{x_i, x_j\}$, or $\pi(x) = x_j$ otherwise. Apparently $\pi: T \to T'$ is surjective.

2) Let $x_0' = x_i$ if $x_0 \in \{x_i, x_j\}$ or $x_0' = x_0$ otherwise, so $\pi(x_0) = x_0'$. By condition i) in construct 2. 1 and the definition of π, we know for x_i, $x_j \in X$, if $x_i \in X_m$ but $x_j \in X - X_m$ then $\pi(x_i) \neq \pi(x_j)$, which means $X_m = \pi^{-1}(X_m')$.

3) Suppose $\xi(\sigma, x)$ be defined for some $\sigma \in \Sigma$ and $x \in X$. Consider the following four cases, respectively.

i) $x \in X - \{x_i, x_j\}$ and $\xi(\sigma, x) \in X - \{x_i, x_j\}$. Let $\xi'(\sigma, x) := \xi(\sigma, x)$. Immediately, $\xi'(\sigma, \pi(x)) = \pi(\xi(\sigma, x))$;

ii) $x \in X - \{x_i, x_j\}$ and $\xi(\sigma, x) \in \{x_i, x_j\}$. By construct 2. 1 c), $\xi'(\sigma, x) = x_i$, so $\xi'(\sigma, \pi(x)) = \xi'(\sigma, x) = x_i = \pi(\xi(\sigma, x))$;

iii) $x \in \{x_i, x_j\}$ and $\xi(\sigma, x) \in X - \{x_i, x_j\}$. By construct 2. 1 b), $\xi'(\sigma, x_i) = \xi(\sigma, x)$, so $\xi'(\sigma, \pi(x)) = \xi'(\sigma, x_i) = \xi(\sigma, x) = \pi(\xi(\sigma, x))$;

iv) $x \in \{x_i, x_j\}$ and $\xi(\sigma, x) \in \{x_i, x_j\}$. By construct 2. 1 a), $\xi'(\sigma, x_i) = x_i$, so $\xi'(\sigma, \pi(x))$

$=\xi'(\sigma,x_i)=x_i=\pi(\xi(\sigma,x))$;

Thus we have proved $\xi'(\sigma,\pi(x))=\pi(\xi(\sigma,x))$ only if $\xi(\sigma,x)$ is defined.

4) By construct 2. 1 d), if $x\in\{x_i,x_j\}$ then $f'(x)=f(x)$ otherwise $f'(x)$ is defined to be equal to $f(x)$. Therefore $f'(\pi(x))=f(x)$ naturally holds. Q. E. D

Suppose T' be a proper supervisor for G_c obtained from T through construct 2. 1. By iteratively using construct 2. 1, we could eventually get a more simplified supervisor $T''=(S'',f'')$ with $\|X''\|\leqslant\|X'\|\leqslant\|X\|$. Furthermore, if G_c is finite and so is T, then $\|X\|<\infty$. Since $\|X\|$ decreases, if possible, at least one after one iteration of construct 2. 1, so after many iterations of construct 2. 1 we could ultimately get a proper supervisor $T^N=(S^N, f^N)$ which is most simplified w. r. t. construct 2. 1. The above argumentation can be summarized as following two corollaries.

Corollary 2. 1 Let $T=(S,f)$ given be a proper supervisor for a CDES G_c. Let T^N denote the supervisor from T after iteratively using construct 2. 1 N times (N is an integer and $N\geqslant 0$). Then, T^N is proper, too and $L(T/G_c)=L(T^N/G_c)$.

Corollary 2. 2 Let $T=(S,f)$ given be a proper supervisor for a CDES G_c. If G_c is finite, then there is an integer $N(N\geqslant 0)$ such that for all $p\geqslant N$ and p being an integer, $T^p=T^N$(here T^p and T^N represent using for T construct 2. 1 p and N times, respectively).

3 An Algorithm for Supervisor Optimization

For a finite CEDS G_c given, corollary 2. 2 hints us there exists $N(N\geqslant 0$ and N being an integer) such that T^N is an optimal supervisor of T (w. r. t. construct 2. 1). Moreover, by corollary 2. 1, T^N and T are equivalent with respect to the control effect to G_c.

Now let $T=(S,f)$ with $S=(X,\Sigma,\xi,x_0,X_m)$ and $f:X\to\{0,1\}^{\Sigma}$. And let $\|X\|=n$ being the state number and $\|\Sigma\|=m$ being cardinality of Σ, we have

Algorithm R Optimization for T using construct 2. 1.

Comment: flag is an indicator to show whether there are states having been merged (flag $=1$) or not (flag$=0$).

```
flag:=1;
while  flag=1 do
begin  flag:=0;
       for  i:=1  to  ||X||-1  do
       begin if  x_i≠"*"
            then for  j:=i+1  to  ||X||  do
                 begin if  x_j≠"*"  then
                     begin  Construct 2.1;
                            if  x_j≠"*"  then flag:=1
                     end
                 end
       end
```

end

Proposition 3. 1 Algorithm R can correctly compute T^N.

Proof Lines 4 and 6 indicate the construct 2. 1 will be employed for arbitary two different states of X; lines 2 and 9 of R show that lines 4 and 6 will be repeated whenever there are states in X having been merged. Since G_c is finite, R will ultimately finish when $T^N=(S^N,f^N)$ and there exists no state in S that could be further merged, i. e. T^N is an optimal supervisor of T for G_c. Q. E. D.

Proposition 3. 2 R is a P-algorithm.

Proof Consider the worst case. In algorithm R, the construct 2. 1 will run at most $n(n-1)^2/2$ times. In construct 2. 1, the number of processing steps for x_i and x_j will be less than $2m(n+1)$ (considering there are nm edges coming into and m edges going out of every state). Thus R needs at most $nm(n+1)(n-1)^2$ basic operation, that is R is an $O(mn^4)$ time algorithm.

Q. E. D.

4 Examples

In the following examples, ⊙ denotes an initial state, • a marker state and * arbitary control. Specifically, ⊙ is an ininial and marker state.

Example 4. 1 Suppose $T=(S,f)$ be given by Fig. 1 and Table 1 (rows 1 and 4).

Table 1 Control assignment

state	x_9	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
state	x_9^1	x_1^1	x_1^1	x_2^1	x_2^1	x_3^1	x_3^1	x_4^1	x_4^1
state	x_7^1	x_7^1	x_1^1	x_1^1	x_6^1	x_6^1	x_3^1	x_3^1	x_7^1
f	* *	* *	10	* *	10	01	* *	01	* *
f^1	* *	10	10	10	10	01	01	01	01
f^N	* *	* *	10	10	10	01	01	01	* *

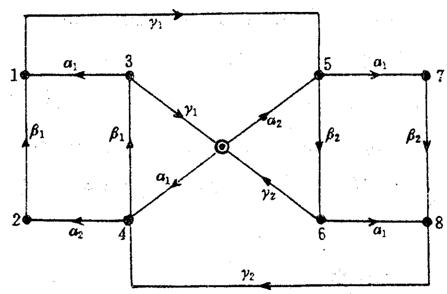

Fig. 1 A supervisor

Here, $\Sigma=\{\alpha_1,\alpha_2,\beta_1,\beta_2,\gamma_1,\gamma_2\}$, $\Sigma_c=\{\beta_1,\beta_2\}$. After applying R. we get $T^1=(S^1,f^1)$ as illustrated in Fig. 2 below and Table 1 (rows 2 and 5). It is easy to verify that T^1 is an optimal supervisor to T.

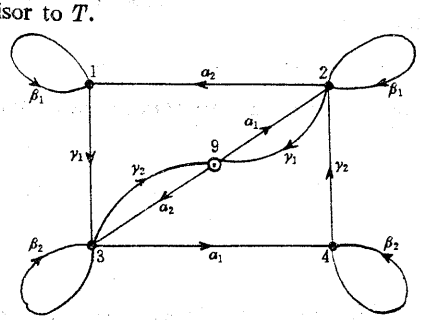

Fig. 2 One optimal supervisor of Fig. 1

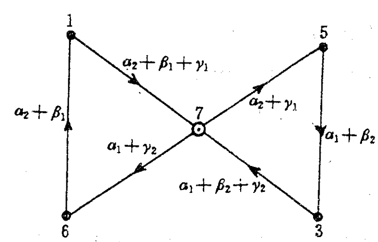

Fig. 3 Another optimal supervisor of Fig. 1

Example 4. 2 Suppose $T=(S,f)$ be as with example 4. 1 but x_1 is renumbered as x_7, x_2 as x_1, x_3 as x_2, x_4 as x_6, x_6 as x_4 and x_7 as x_3, then the simplified $T^1=(S^1, f^1)$ will be another form as illustrated in Fig. 3 below and Table 1 above (rows 3 and 6).

This example shows that for a given T, there may be more than one T' which are all optimal supervisors from T w.r.t. R.

Example 4.3 Consider a supervisor T whose corresponding S and f are given in Fig. 4 and Table 2 (rows 1 and 4) below

Table 2 Control assignment

state	x_6	x_1	x_2	x_3	x_4	x_5
state	x_6^1	x_1^1	x_1^1	x_2^1	x_3^1	x_4^1
state	x_6^N	x_6^N	x_6^N	x_6^N	x_6^N	x_6^N
f	0	*	*	*	*	*
f^1	0	*	*	*	*	*
f^N	0	0	0	0	0	0

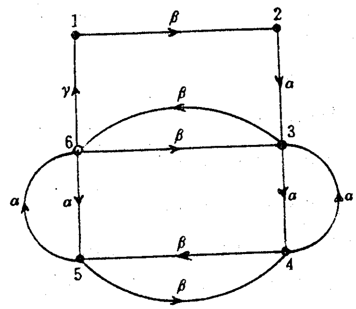

Fig. 4 A supervisor to be optimized

Here, $\Sigma=\{\alpha,\beta,\gamma\}$, $\Sigma_c=\{\gamma\}$. After applying R, we get $T^1=(S^1, f^1)$ as shown in Fig. 5 and Table 2 (rows 2 and 5). Note, however, T^1 is not a real optimal supervisor from T but an optimal supervisor w.r.t. R. In fact, the real optimal supervisor T^N from T is as illustrated in Fig. 6 and Table 2 above (rows 3 and 6).

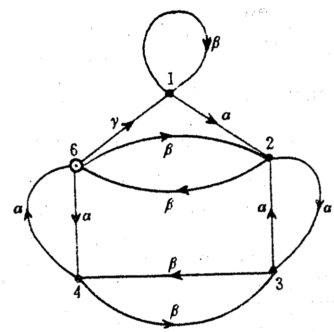

Fig. 5 An optimal (w.r.t. R) supervisor of Fig. 4

This example indicates algorithm R will sometimes result in a sub-optimal supervisor, which may be considered to be a sort of retribution for the algorithm to be polynomial-time solvable.

5 Conslusion

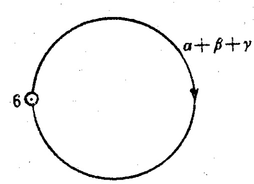

Fig. 6 A real optimal supervisor of Fig. 4

We proposed an algorithm to solve the problem of supervisors optimization. In most cases, the algorithm will transform any given supervisor into a optimal one w.r.t. R (although sometimes there may exist many such supervisors with diffrerent topological structures or control assignments). In reality, only if there were no so-called deadlocks in T, will there be a unique optimal supervisor in the sense that its states number is minimum as will be discussed in a future article.

References

[1] Ramadge, P. J. and Wonham, W. M.. Supervisory Control of a Class of Discrete-Event Processes. SIAM J. on Contr. and Optimization, 1987, Jan. 25(1):206—230

[2] Eilenberg, S.. Automata, Language and Machines. Academic Press, New York, 1974, A

[3] Wonham, W. M. and Ramadge P. J.. On the Supremal Controllable Sublanguage of a Given Language. SIAM J. on

Contr. and Optimization, 1987, May, 25(3):637—659

[4] Vaz, A. F. and Wonham, W. M.. On Supervisor Reduction in Discrete-Event Systems. Int. J. Control, 1986, 44(2): 475—491

给定离散事件系统监督器的优化

蒋智平　吴智铭

(上海交通大学自动控制系，200030)

摘要：本文基于 Ramadge 和 Wonham 等人提出的监督器模型，介绍一种新的优化设计思想，它可使监督器的状态数相对最少化并由此提出一个多项式时间算法. 最后附以三个例子.

关键词：离散事件系统；控制；产生器；优化；算法

本文作者简介

蒋智平　1959 年生. 1981 年获上海交通大学计算机专业学士学位；1987 年获上海交通大学微机研究所硕士学位，1992 年获上海交通大学自动控制博士学位. 1982 年初至 1985 年 9 月及 1987 年底至 1989 年 9 月曾在镇江船舶学院计算机系任教并被评为讲师职称. 早期研究方向有计算机系统程序设计，微型计算机应用及计算机通讯数据加密等. 1989 年后主要从事离散事件系统的仿真和控制的研究.

吴智铭　1936 年生. 1958 年后在上海交通大学电机系执教. 1979 年至 1981 年在英国 UMIST 控制系统中心进修. 1985 年任教授，1986 年任博士生导师. 主要研究方向包括离散事件动态系统的理论和仿真、智能控制与决策等.

“何潘清漪优秀论文奖”评审小组公告

鉴于 1992 年应征论文的质量未达授奖要求，评审小组决定本年度暂停授奖，将名额顺移至下一年度。希望作者们继续踊跃投稿。

“何潘清漪优秀论文奖”评审小组

一九九三年五月