第 9 卷第 1 期
1992 年 2 月

控制理论与应用
CONTROL THEORY AND APPLICATIONS

Vol. 9, No. 1
Feb., 1992

离散事件动态系统状态空间描述的进一步研究*

汪自勤　宋文忠　冯纯伯
(东南大学自动化研究所·南京，210018)

摘要：本文对离散事件动态系统的状态空间描述问题作了进一步的研究. 指出了原有结果的错误和不足，讨论了既有串行阻塞又有并行阻塞的复杂阻塞现象，建立了适用于一般系统的状态方程.

关键词：离散事件动态系统；状态方程；阻塞

1 引　言

离散事件动态系统理论的研究近年来受到国内外学者的高度重视，离散事件动态系统(DEDS)与传统的连续变量动态系统(CVDS)有着本质的区别. 系统动态受离散事件的驱动，仅在一些离散时刻发生变化，其它时刻则保持不变. DEDS 不能用描述 CVDS 的微分方程来描述，至今也还没有一个能与微分方程相媲美的简明有效的模型，这是 DEDS 研究工作所面临的最大难题，我们在文[1]中对 DEDS 的建模分析方法及现状作了较详细的介绍，在此不再赘述.

状态方程向来是系统建模和分析的一个有力的数学工具. 由于研究摄动分析的需要，Ho 和 Cassandras 在文[2]中建立了时间域上 DEDS 的状态方程. 后来他们又加以改进，在文[3]中建立了事件域上的状态方程. 现简介如下：

考虑的对象为一由 M 个服务台组成的离散事件动态系统. 严格地定义事件为某服务台对一顾客完成了服务，而将其它事件称之为诱发事件，如 NI 事件，FO 事件.

定义：

k 为系统所有事件按照发生时间先后排列的事件序号，$k=1,2,3,\cdots$.

$t_i(k)$ 为第 i 服务台上一个事件的发生时间.

$C_i(k)$ 为第 i 服务台的队长.

$S_i(k)$ 为当前正在第 i 台接受服务的那个顾客的服务时间.

$D_i(k)$ 为上述顾客在服务台 i 服务完毕后的去向.

如果第 i 台正处于 NI 或 FO 状态，则称服务台 i 属于非活动集 $\bar{A}(k)$，反之则属于活动集 $A(k)$.

定义下一个事件的发生函数为

$$S(k) \triangleq \arg\min_{i\in A(k)}\{t_i(k)\}. \tag{1}$$

定义第 i 服务台的阻塞函数为

* 863计划 CIMS 资助项目.
本文于1990年5月25日收到. 1991年4月20日收到修改稿.

$$d_i(k)=\begin{cases} d_m(k), & C_m(k)>B_m, \quad m=D_i(k), \\ i, & \text{否则}. \end{cases} \tag{2}$$

在经过上述准备后文[3]给出了如下状态方程：对所有的 $i=1,\cdots,M$ 有：

$$C_i(k+1)=\begin{cases} C_i(k)+1, & S(k)=n, \quad D_n(k)=i, \quad d_i(k)\neq n, \\ C_i(k)-1, & S(k)=d_i(k)=b, \quad D_b(k)\neq i, \\ C_i(k), & \text{否则}. \end{cases} \tag{3}$$

$$t_i(k+1)=\begin{cases} t_n(k)+S_i(k), & S(k)=n, \quad D_n(k)=i, \quad C_i(k)=0, \\ t_b(k)+S_i(k), & S(k)=d_i(k)=b, \\ & [C_i(k)\neq 1; \quad C_m(k)\neq B_m, \quad D_i(k)=m], \\ & \text{或 } D_b(k)=i, \\ t_i(k), & \text{否则}. \end{cases} \tag{4}$$

与时间域相比，事件域上的状态空间描述更加自然、简单，更切合离散事件系统的特点，因此现已被普遍采用. 然而上述结果没有考虑多个服务台被同时阻塞的现象，只适用于较为简单的有串行阻塞现象的系统. 本文将进一步研究复杂阻塞现象，建立适应于一般 DEDS 的事件域上的状态方程.

此外，需要指出的是，公式(1)～(4)还隐含着两个错误. 一是阻塞函数的定义式(2)并不充分，还需要补充一个条件，即 i 属于非活动集；二是，当一个被阻塞的顾客离开 i 台时，根据式(3)$C_i(k)$将在阻塞的开始和结束时都减去 1，这显然是不对的.

2 具有复杂阻塞系统的状态方程

多个服务台被同时阻塞(Simultaneous blocking)又称为并行阻塞，除串行生产线外，在一般的 DEDS 中这一现象通常是不可避免要发生的. 同时阻塞和串行阻塞链并存是一个相当复杂的现象，图 1 给出了一个典型的例子. 这里服务台 b 阻塞了 i_1 和 i_2 台，而 i_1 台又阻塞了 i_3 和 i_4 台，i_3 台阻塞了 i_5 台，i_4 台又阻塞了 i_6 和 i_7 台. 文[4]首先注意并研究了这一问题，虽然最后结果仍不正确，但文中提出的一些概念和分析结果对这一问题的解决还是很有益的.

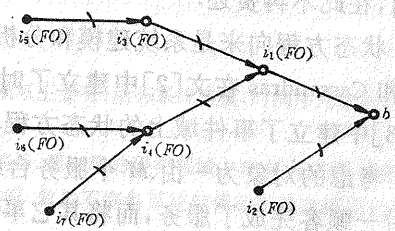

图 1 一个复杂阻塞的典型例子

下面我们将给出一个正确的描述. 首先重新定义阻塞函数：

$$d_i(k)=\begin{cases} d_m(k), & C_m(k)>B_m, \quad m=D_i(k), \quad i\in\bar{A}(k), \\ i, & \text{否则}. \end{cases} \tag{5}$$

$d_i(k)$是状态变量和输入变量的函数. 对于图 1 中的例子将有 $d_j(k)=b, \forall\ j=1,2,\cdots,7$. 直接根据定义(5)估计 $d_i(k)$是比较费时的，当存在阻塞链等复杂阻塞现象时甚至是很困难的. 这是因为必须从阻塞链的根部开始逐级向上递推估计方能得到正确结果. 关于 $d_i(k)$的估计问题至今未引起人们的足够注意. 下面我们将给出一种方法，即在系统的演化过程中对 $d_i(k)$作必要的更新，而不是在每一个状态下估计 $d_i(k)$.

每一个被服务台 b 阻塞的服务台 i 与 b 之间都形成了一条阻塞链，我们用 $Ch_{bi}(k)$表示在这一阻塞链上的服务台集合，于是

$$Ch_{bi}(k)=\{m_j: m_j=D_{m_{j-1}}(k), \quad j=1,2,\cdots,n, \quad m_1=i, m_n=b\}. \tag{6}$$

同时定义 FO 解除集 $M_b(k)$为当第 k 次事件发生在服务台 b 时将被解除阻塞的服务台集合. 它也是一条阻塞链.

FO 解除集的概念是文[4]提出的. 下面我们分几种情况来讨论 $M_b(k)$和 $d_i(k)$的求法.

1) $S(k)=b$, $D_b(k)=m$, $C_m(k)\geqslant B_m$, $d_m(k)\neq b$.

此时服务台 b 将被服务台 m 阻塞. 不管 b 台是否已阻塞了其它台，显然都有：

$$M_b(k)=\{\varphi\}, \tag{7}$$

$\{\varphi\}$表示空集. 服务台 b 以及所有被服务台 b 阻塞的服务台其阻塞函数值都将发生如下变化：

$$d_i(k+1)=d_m(k), \quad \forall\, i: d_i(k)=b. \tag{8}$$

2) $S(k)=b$, $D_b(k)=m$, $C_m(k)<B_m$, $C_b(k)>B_b$.

此时服务台 b 已经阻塞了其它服务台，而服务台 b 又将不会被阻塞，故 k 次事件后，服务台 b 将解除部分(有同时阻塞时)或全部(仅有串行阻塞链时)由服务台 b 引起的阻塞. 遇到同时阻塞时，阻塞的解除顺序由事先给定的优先权而定，这里我们假定采用先阻塞先解除的原则，于是

$$M_b(k)=\{m_j: m_j=\underset{\substack{i\\ d_i(k)=b,\\ D_i(k)=m_{j-1}}}{\arg\min} t_i(k), \quad j=1,2,\cdots, \quad m_1=b\}. \tag{9}$$

上式递推地给出了每一个被解除了阻塞的服务台号，它们构成了一个由 b 开始的串行服务台链. 它只是图1所示的树状结构中的一条线. 对于任何一个被服务台 b 阻塞但是不属于 $M_b(k)$的服务台 i，阻塞链 $Ch_{bi}(k)$与被解除阻塞链 $M_b(k)$有一个分权点，分权点的服务台号为

$$b_{bi}(k)=\begin{cases} n, & d_i(k)=b,\ i\notin M_b(k),\ D_j(k)=n\in M_b(k),\\ & j\in Ch_{bi}(k), j\notin M_b(k),\\ \text{无定义}, & \text{否则}. \end{cases} \tag{10}$$

因此

$$d_i(k+1)=\begin{cases} i, & d_i(k)=b,\ i\in M_b(k),\\ b_{bi}(k), & d_i(k)=b,\ i\notin M_b(k). \end{cases} \tag{11}$$

上式说明，对于被服务台 b 阻塞但是不属于 $M_b(k)$的服务台，虽然 k 次事件后阻塞未被解除，但是引起阻塞的原因已由服务台 b 变为 $b_{bi}(k)$.

3) $S(k)=b$, $D_b(k)=m$, $C_m(k)>B_m$, $d_m(k)=b$.

此时表明已形成了阻塞回路，如仍遵守先阻塞先解除的原则，则阻塞回路可能出现死锁，这是正常系统应避免的现象. 因此作为先阻塞先解除原则的一个例外，当出现阻塞回路时，规定处在这一回路上的服务台具有输出顾客的优先权，因此

$$M_b(k)=Ch_{bm}(k)=\{m_i: m_i=D_{m_{i-1}}(k), \quad i=1,\cdots,n, \quad m_1=m, m_n=b\}, \tag{12}$$

$$d_i(k+1)=\begin{cases} i, & d_i(k)=b,\ i\in M_b(k),\\ b_{bi}(k), & d_i(k)=b,\ i\notin M_b(k). \end{cases} \tag{13}$$

综合上述各种情况我们有：

$$M_b(k)=\begin{cases}\{m_j: m_j=\underset{\substack{i\\ d_i(k)=b,\\ D_i(k)=m_{j-1}}}{\arg\min} t_i(k),\quad j=1,2,\cdots,\quad m_1=b\},\\ \qquad S(k)=b,\quad D_b(k)=m,\quad C_m(k)<B_m,\quad C_b(k)>B_b,\\ \{m_i: m_i=D_{m_{i-1}}(k),\quad i=1,\cdots,n,\quad m_1=m, m_n=b\},\\ \qquad S(k)=b,\quad D_b(k)=m,\quad C_m(k)\geqslant B_m,\quad d_m(k)=b,\\ \{\varphi\},\quad \text{否则}.\end{cases}\tag{14}$$

$$d_i(k+1)=\begin{cases}d_m(k), & S(k)=d_i(k)=b,\quad D_b(k)=m,\\ & C_m(k)\geqslant B_m,\quad d_m(k)\neq b,\\ i, & S(k)=d_i(k)=b,\quad i\in M_b(k)\neq\{\varphi\},\\ b_{bi}(k), & S(k)=d_i(k)=b,\quad i\notin M_b(k)\neq\{\varphi\},\\ d_i(k), & \text{否则}.\end{cases}\tag{15}$$

在作了上述准备后，我们可以列出状态方程如下，对所有的 $i=1,2,\cdots,M$ 有

$$C_i(k+1)=\begin{cases}C_i(k)+1, & S(k)=n,\quad D_n(k)=i,\quad d_i(k)\neq n,\\ C_i(k)-1, & S(k)=b,\quad D_b(k)=m\neq i,\quad i\in\{b\}\cup M_b(k),\\ & [C_m(k)<B_m\quad\text{或}\quad C_m(k)\geqslant B_m,\quad d_m(k)=b],\\ C_i(k), & \text{其它}.\end{cases}\tag{16}$$

$$t_i(k+1)=\begin{cases}t_i(k)+S_i(k+1), & S(k)=i,\quad [C_i(k)\neq 1\text{ 或 }D_i(k)=i],\\ & D_i(k)=m,\quad [C_i(k)<B_m\text{ 或 }C_m(k)\geqslant B_m,\quad d_m(k)=i],\\ t_n(k)+S_i(k+1), & S(k)=n,\quad D_n(k)=i,\quad C_i(k)=0,\\ t_b(k)+S_i(k+1), & S(k)=b\neq i,\quad i\in M_b(k),\\ & [C_i(k)\neq 1\text{ 或 }D_b(k)=i],\\ t_i(k), & \text{其它}.\end{cases}\tag{17}$$

现在我们对这一状态方程作一解释. 式(16)的第一部分与式(3)相同. 第二部分可以分为两种情况，一为 $b=i$，此时表明第 i 台刚服务完一个顾客，如果这个顾客是去其它服务台且不被阻塞，服务台 i 前的顾客数将减1. 另一为 $b\neq i$，此时表明服务台 b 刚服务完一个顾客，如果服务台 i 属于被解除集 $M_b(k)$ 且 i 不是该顾客去的服务台，则服务台 i 前的顾客数必将减1. 式(17)的第一种情况对应了一个正常的服务结束，k 次事件发生在 i 台，k 次事件后 i 台既不会空闲也不会被阻塞. 第二种情况是 k 次事件发生在服务台 n，离开 n 台的顾客来到了 i 台从而结束了服务台 i 的空闲状态. 第三种情况是 k 次事件发生在服务台 b，但是 k 次事件解除了服务台 i 的阻塞且 i 台解除阻塞后没有处于空闲状态.

式(1)、(5)、(10)、(14)～(17)就构成了最一般的 DEDS 的状态空间模型体系. 它不仅可以用于 DEDS 的描述和分析，而且本身就是一个仿真递推算法. 在离散事件系统的仿真过程中，给定初始条件后，只要不断地用随机模拟产生出 $S_i(k)$ 和 $D_i(k)$ 并代入上述模型，就可得到 DEDS 的一个样本轨迹. 摄动分析算法也可以直接嵌入其中.

参 考 文 献

[1] 汪自勤，宋文忠，冯纯伯. 离散事件动态系统的分析和优化——排队网络模型方法(上)、(下). 信息与控制，1989，18(6)：31—40；1990，19(1)：35—44

[2] Ho, Y. C. and Cassandras, C. G.. A New Approach to the Analysis of Discrete Event Dynamic Systems. Automatica, 1983, 19(2): 149—167

[3] Cassandras, C. G and Ho, Y. C.. An Event Domain Fomalism for Sample Path Perturbation Analysis of Discrete Event Dynamic System. IEEE Trans. Automatic Control, 1985, 30(12): 1217—1221

[4] 李彦平. 一类离散事件动态系统状态模型的研究, DEDS 理论及其在 CIMS 中的应用. 北京: 1988年学术讨论会论文集, 70—74

A Further Study on State-Space Representation of Discrete Event Dynamic Systems

WANG Ziqin, SONG Wenzhong and FENG Chunbo

(Research Institute of Automation, Southeast University·Nanjing, 210018, PRC)

Abstract: The problem of state-space representation of discrete event dynamic systems is further studied in this paper. The errors and shortages in existing references are pointed out, and the complicated blocking with both tanded blocking chains and simultaneous blocking are considered. A set of state equations is derived for general discrete event dynamic systems.

Key words: discrete event dynamic system; state equation; blocking

本文作者简介

汪自勤　1983年毕业于武汉水电学院. 1986年于南京工学院获硕士学位. 1990年在东南大学获控制理论与应用专业博士学位. 现在挪威工学院读博士后, 目前正从事系统鲁棒性的研究.

宋文忠　1960年毕业于南京工学院. 现任东南大学自动化研究所教授. 目前正从事生产过程综合自动化和离散事件系统的研究.

冯纯伯　1950年毕业于浙江大学. 1953年毕业于哈尔滨工业大学研究生班. 1958年在苏联列宁格勒工业大学获博士学位. 现任东南大学研究生院副院长, 东南大学自动化研究所教授、所长. 目前正从事非线性理论和智能控制的研究.