第13卷 第1期 自 动 化 学 报 Vol. 13, No. 1
1987 年 1 月 ACTA AUTOMATICA SINICA Jan., 1987

提高计算机控制系统实时性的一个策略

唐 正 中
（湖南省计算技术研究所）

摘 要

本文叙述了提高计算机控制系统实时性的一个策略. 文中将实时任务分为若干子任务，并运用统筹方法，按其紧急程度重新统一排序，以此来减少任务的响应时间和总的等待时间.

一、前 言

计算机控制系统与一般数据处理系统有着明显的不同，为了成功地控制它操作的环境(受控设备)，它必须对环境的变化迅速地给予响应，这就是实时性. 这个特性通常用系统的响应时间来度量. 响应时间是系统为了反应它的环境的变化所占用的时间[1]. 由于环境状态变化的信息通过中断或检测进入计算机后，需先排队等待，然后调度和处理，因此响应时间是等待时间和处理时间之和. 为了增加系统的处理能力或降低建造费用，要设法提高系统的实时性，这在使用微机控制时尤为重要. 下面叙述从软件上提高实时性的一个策略.

二、实时任务调度问题

系统一般控制多台设备，当它们几乎同时输入信息时，若顺次逐个处理，则处理后面的信息的时延会超出允许范围，从而引起故障. 这就可能成为计算机处理任务的“瓶颈口”.

为了解决这一问题，引进‘准实时’概念[2]，将每一个任务再细分为若干子任务，它们是任务中紧急程度近似且相对独立的工作部分. 不同的子任务紧急程度不同，最紧急的称为‘实时的’，而其他的统称为‘准实时的’，且可根据紧急程度而赋予不同的优先权.

现设系统控制N台设备 $D_1, D_2, \cdots, D_n$，其相应的处理任务为 $W_1, W_2, \cdots, W_n$. 将 W_i 分为m个子任务，记为

$$W_i = w_{i1}w_{i2}\cdots w_{im}, \quad (i = 1, 2, \cdots, n)$$

还可以形式地表示为一矩阵

本文于 1984 年 9 月 3 日收到.

$$W=\begin{bmatrix} w_{11} & w_{12}\cdots w_{1m} \\ w_{21} & w_{22}\cdots w_{2m} \\ & \vdots \\ w_{n1} & w_{n2}\cdots w_{nm} \end{bmatrix} \tag{1}$$

设 w_{ij} 的处理时间为 t_{ij}，最大允许等待时间(或时限)为 d_{ij}，则得到相应的处理时间矩阵 $T=(t_{ij})$ 和时限矩阵 $D=(d_{ij})$，$i=1,2,\cdots,n$；$j=1,2,\cdots,m$。

此外，对于任务处理有工序约束，用 $A\to B$ 表示 A 先于 B，现规定(1)中的任务有：

$$w_{ij}\to w_{i,j+1},\quad (j=1,2,\cdots,m-1) \tag{2}$$

设 w_{ij} 的实际等待时间为 $r_{ij}\geqslant 0$，则有时限约束

$$r_{ij}\leqslant d_{ij}. \tag{3}$$

问题的完整提法是，有 $n\times m$ 个任务 $w_{ij}(i=1,2,\cdots,n;j=1,2,\cdots,m)$ 需要实时处理，只有一个加工者(计算机)，如何安排加工次序，使得在满足约束条件(2)，(3)的同时，总的等待时间最小或接近最小。这是一个组合调度或排序问题。

三、一个调度策略

首先说明两个约束条件的关系。若任务 A,B 有 $A\to B$ 关系，则其实际等待时间 r_A，r_B 满足 $r_A<r_B$，并可假定最大允许等待时间亦有上述类似关系，即 $d_A<d_B$ (d_A，d_B 为 A,B 任务的时限)。

现对W矩阵中 w_{ij} 进行统一排序，其结果为所需的处理队列 Q。为此将矩阵中任一行的元素 $w_{i1},w_{i2},\cdots,w_{im}(i=1,2,\cdots,n)$ 看成一个队列，称为 Q_i，求Q的算法分为两步：第一步求出满足工序约束和总等待时间最小的队列 Q；第二步对Q中元素进行检验与调整，使之能满足时限约束。

算法 *A*

(1) 每次选取 $Q_i(i=1,2,\cdots,n)$ 中第一个元素，并从它们中找出其处理时间最小者，如 t_{kl}。若某 Q_i 中元素已被挑选完，为空队列，则不从中选取元素。

(2) 确定所选的元素 t_{kl} 对应的任务 w_{kl}，将它加入队列Q的末尾，且从 Q_k 中抹去 w_{kl}。

(3) 对 Q_1，Q_2，$\cdots$，Q_m 所余元素重复(1)，(2)步骤，直至全部挑选完。

显然，队列Q有 $n\times m$ 个任务，其加工次序保证了工序约束，且总的等待时间最小。

设Q的元素为 $q_1,q_2,\cdots,q_{nm}$，从左至右逐个进行时限检验与调整，使之还满足时限约束条件。为叙述方便，引用下述记号：i 为检验项在Q中的位置计数；m 为检验项前移次数；l 为对换目标项与检验项的间隔；E 为移动项的集合；S 为移动项所在队列的元素的集合；t_k，r_k，d_k 分别为 q_k 的处理时间、实际等待时间和时限。

算法 *B* (时限检验与调整)

(1) 开始检验。$i\leftarrow 2$，$S=\phi$，$E=\phi$。

(2) 对 q_i 进行时限约束检验。先求出 r_i，d_i，且 $r\leftarrow r_i$，$d\leftarrow d_i$，此处

$$r_i = \sum_{k=1}^{i-1} t_k.$$

若 $r \leqslant d$ 成立,即满足时限约束,转(3),否则,转(4).

(3) 准备检验下一项. $i \leftarrow i+1$, 若 $i = n \times m + 1$ 则结束, 此即所求解的队列 Q, 否则,转(2).

(4) 准备前移检验项. $m \leftarrow 0$, $l \leftarrow 1$, 且 $E = \{q_i\}$, $S = \{Q_i'\}$ (Q_i' 为 q_i 所属队列的元素的集合,下同), $t' \leftarrow t_{i-l}$.

(5) 对换目标项工序检验. 若 $q_{i-l} \in S$, 因工序约束不能对换,转(10);否则往下.

(6) 计算 E 中的项的总的处理时间 $T = \sum_{k=i-l+1}^{i-m} t_k$.

(7) 对换目标项时限检验. 若 $\Delta t \geqslant T$,则能对换,转(8);否则转(10)(Δt 为 q_{i-l} 的剩余时间, $\Delta t = d_{i-l} - r_{i-l}$).

(8) 对换. $q_{i-l+1}q_{i-l+2}\cdots q_{i-m}$ (E 中的项)向前推移一项,而 q_{i-l} 移至其后,且计移动次数 $m \leftarrow m+1$.

(9) 再次时限检验. 若 $r - t' \leqslant d$ 成立, 则满足时限约束,转(3);否则往下.

(10) q_{i-l} 加入前移项. $E \leftarrow EU\{q_{i-l}\}$, $S \leftarrow SU\{Q_{i-1}'\}$ (集合 E, S 加入元素后仍表示为 E 和 S).

(11) 准备对换前一项. $l \leftarrow l+1$, 若 $l = i$ 则结束,问题无解;否则转(5).

从上述算法可以看出,因是从左至右进行检验,在检验 q_i 时, $q_1, q_2, \cdots, q_{i-1}$ 已满足(2),(3)式的约束条件. 在对换时, 向后移动的项 q_{i-l} 是在满足工序约束与时限约束条件下进行的,而前移的项等待时限显然减少,故最后得到的解符合要求.此外,调整是将检验项逐渐前推,当达到时限要求时就停止,故对总等待时间的影响最小. 上述方法是一般性的.在实际使用时,常控制同类设备,故可假定 $W_1, W_2, \cdots, W_m$ 相同,且 $t_{1j} = t_{2j} = \cdots = t_{nj} (j = 1,2,\cdots,m)$. 同时,紧急的子任务处理时间短, 因而可假定 $0 \leqslant t_{ii} \leqslant t_{i,i+1}$. 按上述算法;$W$ 阵中任务是按列处理,这时等待时间可用下式求得:

$$R'_{ij} = \sum_{l=1}^{j-1} \sum_{k=1}^{n} t_{kl} + \sum_{k=1}^{i-1} t_{kj}.$$

原来按行处理时, w_{ij} 的等待时间为

$$R_{ij} = \sum_{k=1}^{i-1} \sum_{l=1}^{m} t_{kl} + \sum_{l=1}^{j-1} t_{il}.$$

显然, 对于 W_i 中的紧急部分 w_{i1} 有 $R'_{i1} \leqslant R_{i1}$. 特别是成为“瓶颈口”的 w_{n1}, 有 $R'_{n1} \ll R_{n1}$, 得到很大的改善. 笔者在建立织袜车间控制系统时, 采用了上述方法,大大提高了实时性,增强了系统能力[2]. 在程序设计时, 可采用多任务队列,各队列的优先权不同,按‘优先第一’算法调度[3]. 此外, 还可采用优化程序判断次序的方法来减少任务的处理时间[4].

参　考　文　献

[1] Allworth, S. T., Introduction to Realtime Software Design, Macmilan Publishers LTD. 1981.
[2] 唐正中等，织袜车间微型机分布式控制系统软件设计，计算技术与自动化，1985.
[3] 张尤腊等，计算机操作系统，科学出版社，1979 年.
[4] 唐正中，程序中逻辑判断次序的优化，计算机技术 1982 年第 6 期.

THE POLICY FOR ENHANCING REAL-TIME RESPONSIVENESS IN COMPUTER SYSTEMS

TANG ZHENGZHONG
(Hunan Institute of Computing Technology)

ABSTRACT

In this paper, a policy for enhancing real-time responsiveness in computer control systems is described.

The real-time task is divided into many subtasks which are rearranged according to the degree of instantaneousness with operational research method so as to reduce the response time of the task and the total waiting time.