



# Systolic 算法和结构求解线性系统的 极点配置问题

慕德俊 戴冠中

(西北工业大学自动控制系 西安 710072)

## 摘 要

提出了一种 Systolic 算法和阵列结构,可并行求解大规模极点配置及可控性问题。仿真结果表明,这种方法在计算上有很好的并行性。

**关键词:** Systolic 算法,阵列结构,极点配置。

## 1 引言

极点配置是控制理论中的一个重要问题。对于大规模控制系统的极点配置问题,特别需要一种并行的算法和结构进行并行处理,然而,目前这类算法却很少, Datta 提出了一种算法<sup>[1]</sup>,仅把极点配置问题的一个局部化为  $n$  个独立的 Hessenberg 方程并行计算,但对系统  $\langle A, b \rangle$  变换为 Pascal-like 规范形所需大量计算的并行化以及求解正交矩阵的并行化无能为力,造成了瓶颈效应,使整个计算过程的并行度不高。本文系统地给出了单输入线性系统极点配置所需的 Systolic 算法和阵列结构,其中包括把系统  $\langle A, b \rangle$  变换为 Pascal-like 规范形的 Systolic 算法和阵列结构,这种算法也可用于并行的可控性判别,通过对 Hessenberg 方程的算法进一步改进,用 Systolic 阵列对每个 Hessenberg 方程进一步并行计算,使得整个过程的并行性得到提高。由于多输入情况的极点配置问题可转化为相互独立的单输入极点配置问题<sup>[2,3]</sup>,因而本文所提出的单输入极点配置的算法和结构可推广到并行求解多输入极点配置问题。仿真结果表明,这种方法能提高计算的并行性。

## 2 单输入线性系统的极点配置算法

设单输入线性定常系统有如下形式:

$$\dot{X}(t) = AX(t) + bu(t), t \geq 0, \quad (1)$$

式中  $A \in R^{n \times n}$ ,  $b \in R^n$ ,  $u(t) \in R$ ,  $X(t) \in R^n$ .

上述系统的极点配置问题已有大量的研究, 并得出多种串行算法, Datta 给出了一种数值稳定的算法求解单输入情况下的极点配置问题<sup>[1]</sup>. 对于一个可控对  $\langle A, b \rangle$ , 它的极点配置方法是找出一个矢量  $f (f \in R^n)$ , 使得  $A - bf^T$  具有所要求的谱  $\mathcal{Q}$ ,  $\mathcal{Q} = \lambda_1, \lambda_2, \dots, \lambda_n$ . 此算法可分为 4 步进行.

第 1 步. 通过正交相似变换将  $\langle A, b \rangle$  化为 Pascal-like 规范型, 即找出一个正交矩阵  $P$ , 使得

$$Pb = [\alpha, 0, \dots, 0]^T = c, \quad PAP^T = H = (h_{i,j}), \quad (2)$$

式中  $H$  为上 Hessenberg 矩阵.

由文献[4]可知, 当  $\langle A, b \rangle$  为可控对时,  $H$  是不可约的上 Hessenberg 阵, 即  $H$  的次对角线  $h_{i,i-1}, i = 2, \dots, n$  不为零且  $\alpha \neq 0$ .

第 2 步. 并行求解  $n$  个 Hessenberg 方程

$$(H - \lambda_i I)y_i = C, \quad i = 1, \dots, n. \quad (3)$$

第 3 步. 求出下列方程的  $D$  阵

$$Y^T D = U, \quad (4)$$

式中,  $Y = [y_1, \dots, y_n]^T$ ,  $D = [d_1, \dots, d_n]^T$ ,  $U = [1, \dots, 1]^T$ .

第 4 步. 计算

$$f^T = D^T P, \quad (5)$$

这时  $A - bf^T$  有所要求的谱  $\mathcal{Q}$ , 证明如下:

设  $B = \text{diag}(\lambda_1, \dots, \lambda_n)$  有所需的极点, 构造  $Y = [y_1, \dots, y_n]$ , 使得

$$HY - YB = CU^T, \quad (6)$$

若  $Y$  为非奇异, 则(6)式可变为

$$H - CU^T Y^{-1} = YBY^{-1}, \quad (7)$$

由(4)式可知,  $U^T Y^{-1} = D^T$ , 因此

$$H - CD^T = YBY^{-1},$$

对(7)式初始阵进行正交相似变换得

$$P^T(H - CD^T)P = P^T H P - P^T C D^T P = A - bf^T.$$

因此  $A - bf^T$  与  $B$  有相同的极点, 证毕.

### 3 Systolic 算法及阵列结构

将  $\langle A, b \rangle$  变为 Pascal-like 规范型. 首先用 Givens 旋转法进行正交变换求出  $P_0$  使得

$$P_0 b = [\alpha, 0, \dots, 0]^T, \quad \alpha \neq 0, \quad P_0 \in R^{n \times n},$$

则  $P_0 A P_0^T = A_0$  (假设  $\langle A, b \rangle$  为可控对).

再求出  $P_1$ , 使得

$$\begin{bmatrix} 1 & 0 \\ 0 & P_1 \end{bmatrix} A_0 \begin{bmatrix} 1 & 0 \\ 0 & P_1^T \end{bmatrix} = H, P = \begin{bmatrix} 1 & 0 \\ 0 & P_1 \end{bmatrix} P_0,$$

式中  $H$  为上 Hessenberg 阵,  $P \in R^{n \times n}$  和  $P_1 \in R^{(n-1) \times (n-1)}$  均为正交阵,  $P$  阵通过旋转参数左乘求得, 初始为  $I$  阵. 能够映射到线性阵列的 Systolic 算法描述如下:

初始阶段(计算  $P_0 b, P_0$ )

For  $i = n$  downto 2

Zero ( $b_i, b_{i-1}, \theta_i$ )

For  $j = 1$  to  $n$

Rotate ( $a_{i,j}, a_{i-1,j}, \theta_i$ )

Left-multiply ( $p_{i,j}, p_{i-1,j}, \theta_i$ )

Endfor; Endfor

For  $i = 2$  to  $n$

For  $j = n$  downto 1

Right-multiply ( $a_{j,i}, a_{j,i-1}, \theta_i$ )

Endfor; Endfor

计算  $P$  阵和变  $A_0$  为上 Hessenberg 阵的 Systolic 算法描述如下:

For  $k = 1$  to  $n - 2$

For  $i = n$  downto  $k + 2$

Zero ( $a_{i,k}, a_{i-1,k}, \theta_i$ )

For  $j = k + 1$  to  $n$

Rotate ( $a_{i,j}, a_{i-1,j}, \theta_i$ )

Left-multiply ( $p_{i,j}, p_{i-1,j}, \theta_i$ )

Endfor; Endfor

For  $i = k + 2$  to  $n$

For  $j = n$  downto 1

Right-multiply ( $a_{j,i}, a_{j,i-1}, \theta_i$ )

Endfor; Endfor; Endfor

上述算法很容易映到由  $2n + 1$  个单元组成的线性阵列上, 其阵列结构见图 1 (图中  $n = 6$ )。单元 1 产生  $QR$  旋转参数, 一方面传给左、右两边的单元进行各自的旋转相乘、计算  $P$  阵, 另一方面将旋转参数存在单元 1 的堆栈内(能先取出后存储的旋转参数), 以便在旋转阶段结束后, 将参数传给右边的阵列与输入数据做右乘运算。零化, 旋转及左乘运算完成后, 从阵列下部输出的数据经多路转换器调整后, 反馈到输入端, 右边的单元 2—7 与单元 1 里的旋转参数做右乘运算, 将输出的结果再返回输入端以便进行下次的旋转计算。此期间左边的单元 8—13 处于空闲状态。

初始阶段阵列的输入及输出的数据流见图 1。每次旋转阶段完成之后、右乘阶段开始之前, 需置  $y \leftarrow r$ ,  $r \leftarrow 0$ , 使各单元内的数据输出并清零。对于  $n$  阶系统输入到阵列的数据流见图 2。

从图 2 中可以看出, 在合理地组织数据流动的情况下, 化  $\langle A, b \rangle$  为 Pascal-Like 规

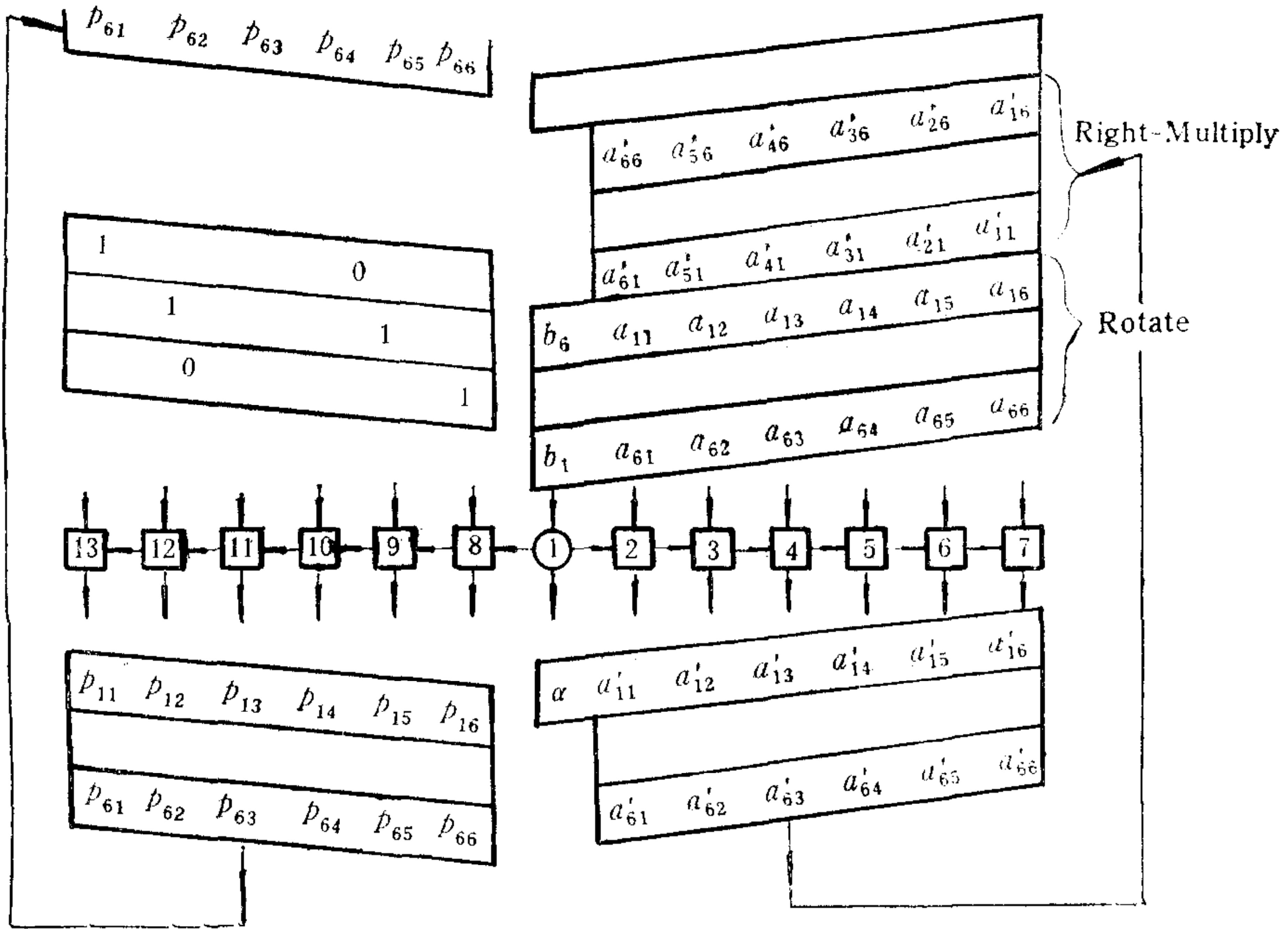


图1 实现极点配置的阵列结构及初始阶段的数据流 (n = 6)

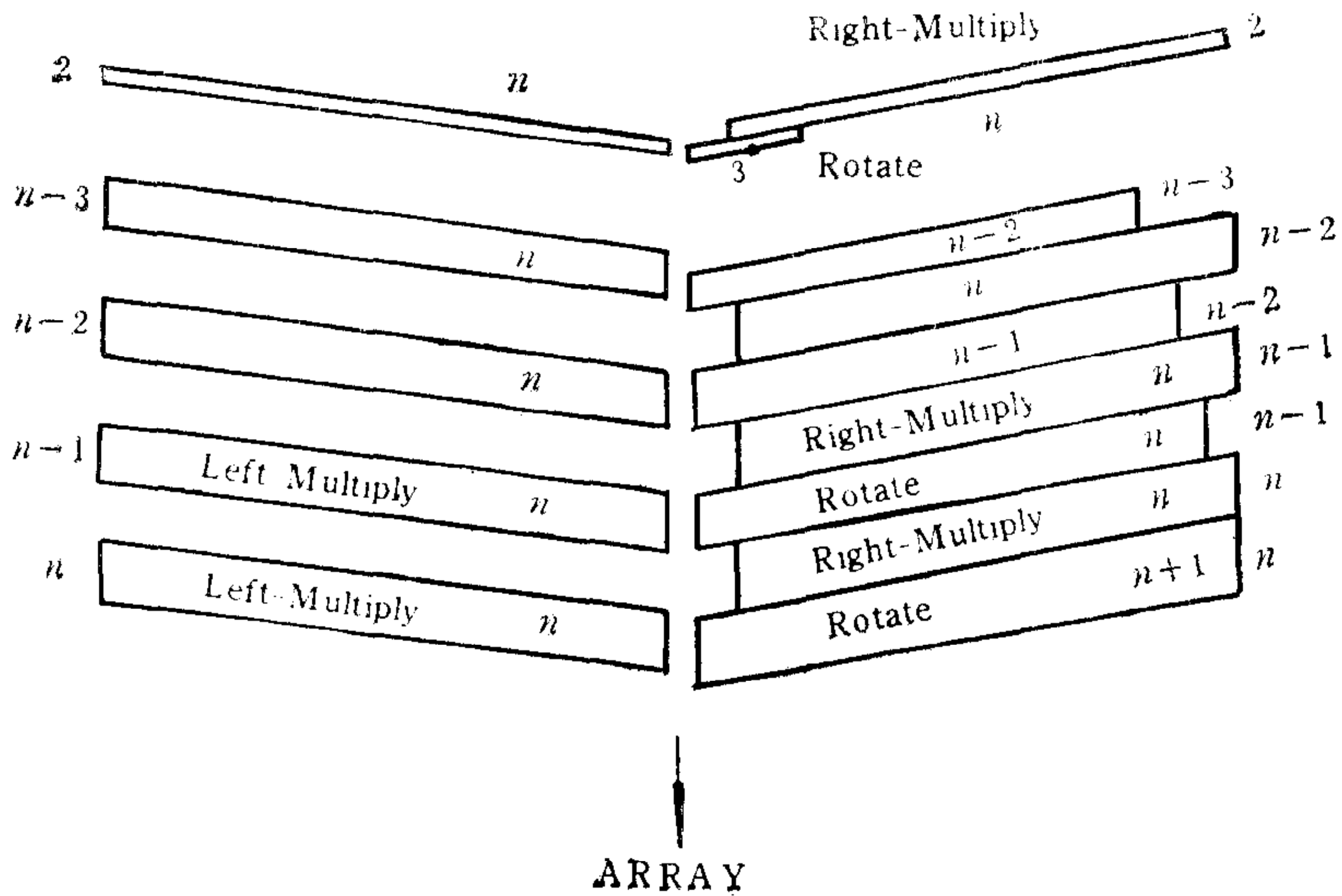


图2 对于 n 阶系统输入到阵列的数据流

范型需  $2 \sum_{i=2}^n = (n-1)(n+2)$  个单位时间, 可见, 计算的时间复杂性由原来串行时的  $O(n^3)$  变为现在的  $O(n^2)$ . 处理单元的利用率  $\eta$  为

$$\eta = \left[ \sum_{i=2}^n i(i+2n+1) \right] / \left[ 2(2n+1) \sum_{i=2}^n i \right] \approx 70\%$$

如果  $\alpha \neq 0$  以及上 Hessenberg 阵的次对角线  $h_{i,i-1} \neq 0 (i = 2, \dots, n)$ , 则系统

$\langle A, b \rangle$  可控,因此这种 Systolic 算法和阵列结构也可判别系统的可控性。

以下进行(3)–(5)式的计算。

可以采用  $n$  个相同的阵列结构并行求解  $n$  个 Hessenberg 方程,对于型如 “ $AX = b$ ” 的线性方程,可采用三角阵列附加一线性阵列结构并行求解<sup>[4]</sup>,若将边界单元的功能稍作修改,可实现本文中的线性方程  $(H - \lambda_i I)Y_i = C$  的求解。当  $Y$  求出后,与矢量  $U$  一起再输入上述三角阵列进行正交三角变换及 Gaussian 消去变换,完成(4)式求  $D$  的并行计算。最后用一个线性阵列进行矢量  $D$  与矩阵  $P$  的乘法运算,得到(5)式的  $f$  矢量。

如果同时使用  $n$  个上述结构执行(3)式的运算,完成(3)–(5)式的计算需  $6n + 1$  个单位时间。由于(2)–(5)式的计算可流水线进行,(2)式的计算需  $(n - 1)(n + 2)$  个单位时间,因此在连续地行极点配置运算时,为了提高处理的利用率,对  $Y$  的计算可采用两个上述结构,一个进行  $[n/2]$  个(3),(4)式的运算,另一个阵列进行  $n - [n/2]$  个(3)式的运算。若采用不同阶段单元执行不同功能的方法,将(5)式的计算映射到图 1 的 8—13 单元上,(2)–(5)式计算的周期为  $2n[n/2] + 3n$  个单位时间,处理器的利用率在 85% 以上。

## 4 仿真结果

在并行计算中,通常用加速比  $S$  来衡量一个并行算法的有效性。加速比用于度量算法并行性对时间的改进程度。其定义为  $S = T_1/T_p$ ,  $T_1$  为最快的串行算法在单处理机上的运行时间,  $T_p$  为并行算法在  $p$  个处理单元上的运行时间。

表 1 不同阶系统串、并行算法的运行时间

阶 数	5	10	15	20	25
串行时间 (s)	0.0052	0.033	0.098	0.215	0.403
并行时间 (s)	0.00075	0.0032	0.0061	0.0112	0.0164

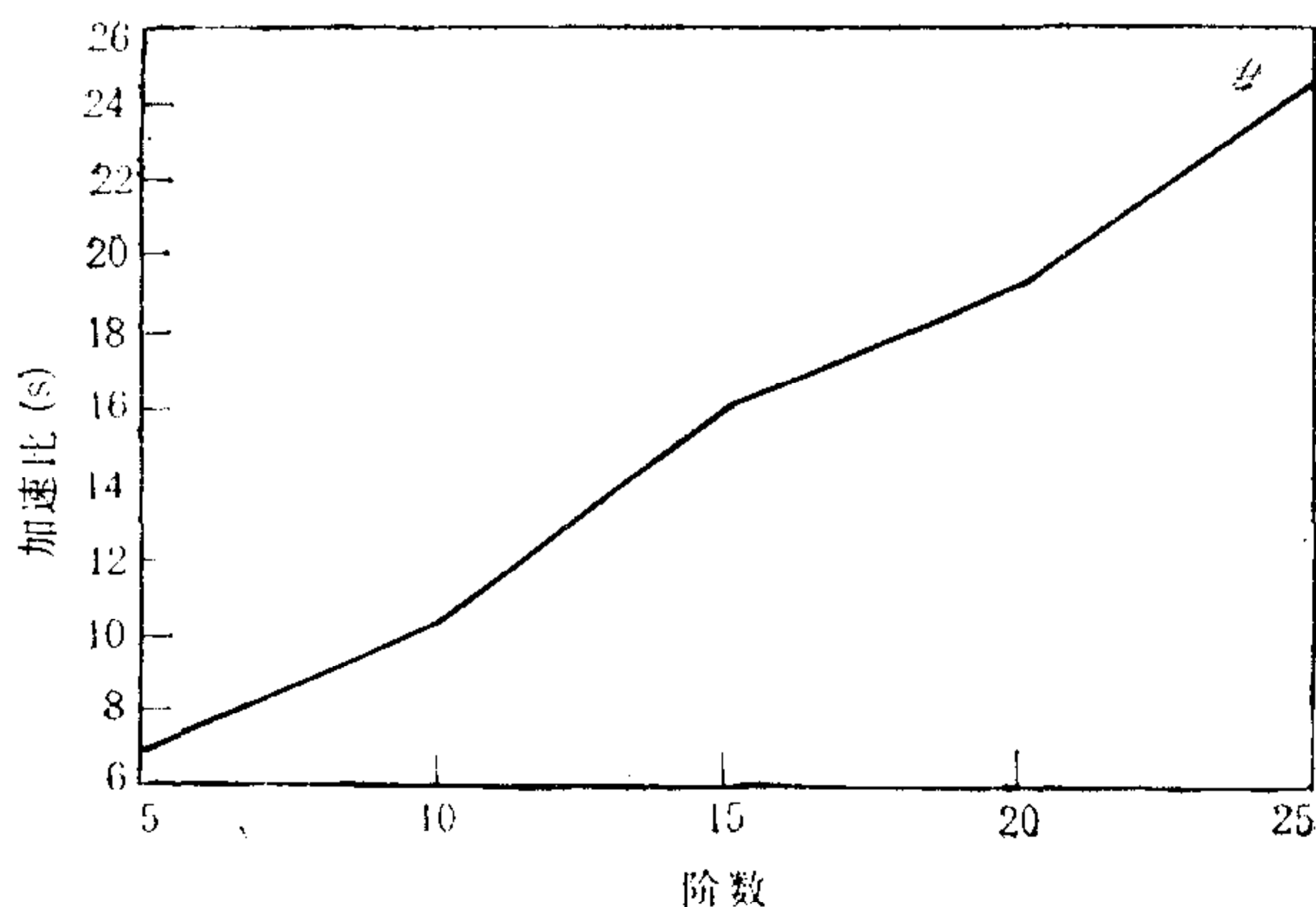


图 3 并行加速比曲线

在 PC-386 加 387 上用 MATLAB 对上述不同阶系统的串、并行算法进行仿真计算,其运行时间见表 1,加速比曲线见图 3。

从图中可以看出,该算法具有良好的并行性。随着问题规模的增大,加速比也随着增加,但这种增加是以硬件开销为代价。

## 5 结束语

本文系统地给出了从 Pascal-Like 规范型的变换到求解线性方程的 Systolic 算法和阵列结构,来并行地求解单输入情况下的极点配置问题,这种方法可有效地进行大规模的极点配置运算,同时也能并行地进行系统可控性的判别。由于多输入反馈极点配置问题可转换为相互独立的单入极点配置问题,因而本文所提出的算法和阵列结构可推广到并行求解多输入极点配置问题。由仿真结果可知,这种算法的加速比较高,计算的复杂性降低。处理器在不同阶段进行不同功能的运算,还可减少全局通讯。

### 参 考 文 献

- [1] Datta B N. Parallel and large-scale matrix computation in control: some ideas. *Linear Algorithm and its Application*, 1989, **121**:243—264.
- [2] 蔡大用,储德林. 极点配置问题的一个稳定方法. *数值计算与计算机应用*,1991,(2): 95—101.
- [3] Patel R V. Numerical algorithms for eigenvalue assignment by state feedback. *Proc. of IEEE*, 1984, **72**(12):1755—1764.
- [4] Kong S Y. VLSI array processors. Preteice Hall, 1988.

## SYSTOLIC ALGORITHM AND ARCHITECTURE FOR POLE ASSIGNMENT PROBLEM

MU DEJUN      DAI GUANZHONG

(Dept. of Automatic Control, Northwestern Polytechnical Univ. Xi'an, 710072)

### ABSTRACT

In this paper, a Systolic algorithm is proposed for large-scale pole-assignment and controllability problems. This algorithm can be efficiently transformed to a linear Systolic array. Simulation result shows that this algorithm has a higher speed-up ratio. This algorithm can be implemented in parallel in  $O(n^2)$  time-steps.

**Key words:** Systolic algorithm, array architecture, pole assignment.