长方矩阵转置的追踪法

程瑞英

（数学系）

电子计算机的发展，使人类避免了重复性的劳动，而且有效的缩短工作时间，长方矩阵的转置本来人工是很易完成的，但是如果矩阵的维数很大时，人工转置容易出错且很费时间，借助于电子计算机则能很快的完成。由于机器编译程序的识别能力，要求矩阵的元素按列存贮，这样，矩阵的一列元素就占用较小的存区，而矩阵的一行元素就分散在较大的存区中。现在有的计算机已有小量的超高速存贮器，要求参加运算的数据集中在一个较小的存区中，如果是矩阵的行元素参加运算（例如线性代数组的迭代法），则将矩阵转置成按行存贮，给高速运算创造了有利条件。若一个程序要求它既能对矩阵A进行运算又能对转置矩阵A^T进行运算，这就要求有一个程序将矩阵A在同一存区中转置成A^T。特别当内存与外部设备（如磁鼓磁带等）交换数据时，A与A^T是不同的数组，在分批交换时差别就更大了。所以矩阵的转置虽是小问题，却在自动处理数据的过程中显示了它的效用。本文介绍求在同一存区中矩阵A转置成A^T的算法，及*FORTRAN*语言的子程序，若再需用矩阵A时，由于$(A^T)^T=A$，可以重复使用程序。

设
$$A_{M\times N}=\begin{vmatrix} A_{11} & A_{12}\cdots\cdots A_{1N} \\ A_{21} & A_{22}\cdots\cdots A_{2N} \\ \cdots\cdots & \\ A_{M1} & A_{M2}\cdots\cdots A_{MN} \end{vmatrix}$$

转置得

$$A^T{}_{N\times M}=\begin{vmatrix} A_{11} & A_{21}\cdots\cdots A_{M1} \\ A_{12} & A_{22}\cdots\cdots A_{M2} \\ \cdots\cdots & \\ A_{1N} & A_{2N}\cdots\cdots A_{MN} \end{vmatrix}$$

在内存的次序分别为：

$A_{M\times N}$：A_{11}，A_{21}，…，A_{M1}，A_{12}，$A_{22}\cdots$，A_{M2}，…，A_{1N}，A_{2N}，…，A_{MN}。

$A^T{}_{N\times M}$：A_{11}，A_{12}，…，A_{1N}，A_{21}，A_{22}，…，A_{2N}，…，A_{M1}，A_{M2}，…，A_{MN}。

由于$A_{M\times N}$与$A^T{}_{N\times M}$的元素相同，只是在内存中排列次序不同而已，如何保证$A_{M\times N}$中的元素一丝不乱的在同一存区变成$A^T{}_{N\times M}$的元素，是一个传送过程，首先应将$A_{M\times N}$中的元素保存起来进行适时的传送。若先保存后传送，有二种方法：一是将A按行传送至另一存区B，按传送先后的次序存放，然后再将B回送至A的存区，程序较简单，但所需内存庞大，当$M\times N$相当大时，可能内存无法同时开辟两个存区，更谈不上很好的进行其他运算了。另一种方法至少需要$2(M-1)$个工作单元来保存$A_{M\times N}$的元素，且程序繁复，有时还会引起混

乱，而追踪法，只要利用一个工作单元进行交换传送，再加上一些必需的工作单元程序也不复什就能完成。

实现转置的基本规律是根据 $A_{M\times N}$ 与 $A^T{}_{N\times M}$ 在存区中地址数的对应关系来考虑。

设矩阵 $A_{M\times N}$ 的存区从第 $A(1,1)$ 个单元开始按列存放，A_{IJ} 的地址数为 $A(I,J)$，转置为 $A^T{}_{N\times M}$ 时是行与列的对换，$A^T{}_{IJ}=A_{IJ}$，但由于 M 与 N 的值不同，$A^T{}_{IJ}$ 的地址数不再是 $A(J,I)$了，如何使 A_{IJ} 正确的存放在 A^T 中该存放的 $A^T(J,I)$ 中，这就是我们要解决的。先举例说明思想方法：

例：

设 $A_{3\times5}=\begin{vmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \end{vmatrix}$ 是按列存放，

转置得 $A^T{}_{5\times3}=\begin{vmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \\ a_{14} & a_{24} & a_{34} \\ a_{15} & a_{25} & a_{35} \end{vmatrix}$ 也是按列存放

现在将存放 a_{IJ} 的地址数 $A(1,1)+(I-1)+(J-1)*M=A(1,1)+K$ 简记为 K，而矩阵的元素 a_{IJ} 地址数与 K 是一一对应的，其顺序分别为

$A_{3\times5}$: $a_{11}a_{21}a_{31}a_{12}a_{22}a_{32}a_{13}a_{23}a_{33}a_{14}a_{24}a_{34}a_{15}a_{25}a_{35}$

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

$A^T{}_{3\times5}$: $a_{11}a_{12}a_{13}a_{14}a_{15}a_{21}a_{22}a_{23}a_{24}a_{25}a_{31}a_{32}a_{33}a_{34}a_{35}$

0 3 6 9 12 1 4 7 10 13 2 5 8 11 14

得到矩阵置换的数字模型为：

$$T:\begin{pmatrix} 0\,1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,11\,12\,13\,14 \\ 0\,3\,6\,9\,12\,1\,4\,7\,10\,13\,2\,5\,8\,11\,14 \end{pmatrix}$$

这是一个有限个自然数的一一变换，也就是一个置换，其中1与3的对应表示1中存贮3的内容，3存贮1的内容。实现这个置换有两种方法，一是按自然顺序转置，另一个是将置换分解成有限个不相交循环置换的积，例中的

$$T=\begin{pmatrix}0\\0\end{pmatrix}\begin{pmatrix}1\,3\,9\,13\,11\,5\\3\,9\,13\,11\,5\,1\end{pmatrix}\begin{pmatrix}7\\7\end{pmatrix}\begin{pmatrix}2\,4\,6\,12\,8\,10\\4\,6\,12\,8\,10\,2\end{pmatrix}\begin{pmatrix}14\\14\end{pmatrix}$$

这里有三个不变点，二个不相交的循环置换，它们实现后，相应的 $A_{3\times5}$ 转置成 $A^T{}_{5\times3}$ 的工作也完成了。

在 $A^T{}_{N\times M}$ 中 $A^T{}_{IJ}=A_{IJ}$。

$$A^T(J,I):J+(I-1)*N-1=K,\ K=0,1,\cdots,MN-1. \qquad (1)$$

在 $A_{M\times N}$ 中

$$A(I,J):I+(J-1)*M-1=L,\ L=0,1,\cdots,MN-1. \qquad (2)$$

MN 是定值，IJ 是参变量，L，K是 I，J 的函数，工作的目的是将 $A_{M\times N}$ 中元素按照 $A^T{}_{N\times M}$ 按列排列的次序存放，所以应从（1）式解得：

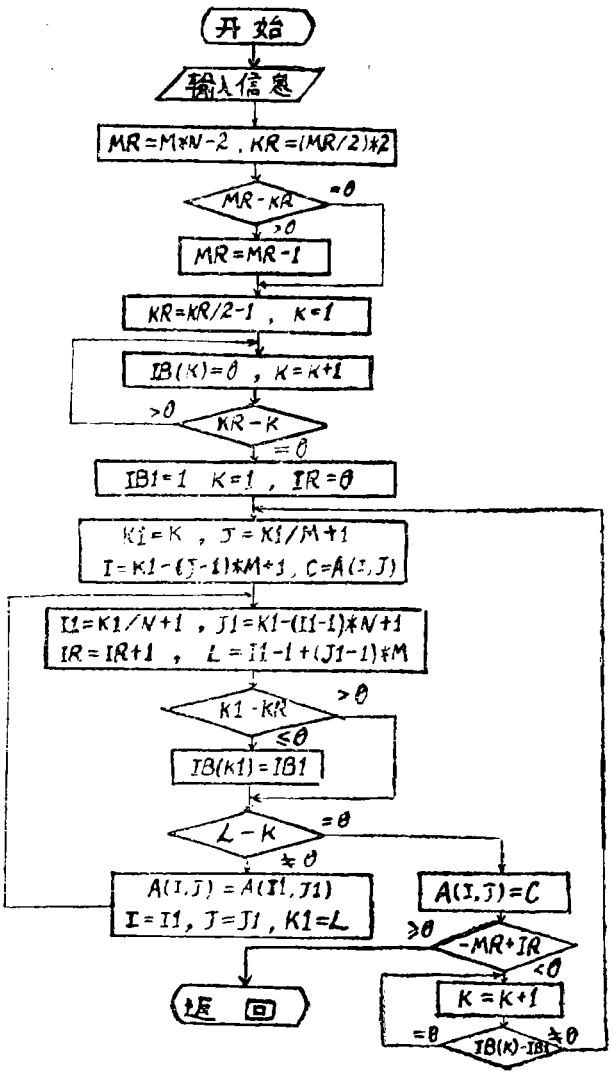
$$I=\left[K/N\right],\ J=K-(I-1)*N \qquad (3)$$

[ ]表示取整，

追踪法一：根据置换分解成有限个不相交循环置换的积的思想。在计算机中要实现有限个不相交的循环置换的步骤是：

i) 先对给定的第一个环头K（不妨设为1）进行追踪传送到头尾相等后暂停传送。具体过程是由(3)得出$A(I,J)$並将K记录在工作单元中，然后实现$A^T(I,J)$传送至另一工作单元C保存，而$A(I,J)$传送到$A^T(J,I)$，$A(I,J)$轮空等待送入，根据(2)计算得L，作为新的K由(3)得I_1,J_1，将$A(I_1,J_1)$送入$A(I,J)=A^T(J_1,,I_1)$，继续循环传送r次，直至所得L与记录在工作单元中的K相等时，将C中内容传送至$A(I_{r-1},J_{r-1})=A^T(J_r,,r)$一个循环置换结束。

(*ii*)寻找新的环头，要在未经传送的数据中找。矩阵（$M\neq N\neq 1$）转置所对应的置换框图：



换，分解成不相交的循环置换的个数，最多不超过$\left[\frac{M*N}{2}\right]$个。所以新的循环头只要在前$[M*N/2]$个元素中寻找，这就要求对已传送过的元素作出标志，便于电子计算机识别，为此不妨利用$M*N/2$个$B(K)$，当$B(K)$中存放1时，表示$A(1,1)+K$已被传送入新数，存放0时表示尚未传送到。当找出第一个尚未传送的K时，就得到新的循环头，重复$i)$中过程，就实现了新的循环置换，继续$ii),i)$的工作，直至元素的对换次数达到$M*N-2$时就停止工作，返回到主程序待命。为了避免计算机在不变点上浪费时间，设置一个奇偶门，当$M*N$是偶数时，转置$M*N-2$次，即除去头尾两点，当$M*N$是奇数时转置$M*N-3$次，因为除了头尾两点不变外，至少还有一个不变点。

程序：

```
C   THIS IS THE SUBROUTINE OF TRANSPOSE
C   OF A MATRIX 1
    SUBROUTINE TMA1(M,N,A)
    DIMENSION A(M,N),IBR(1000)
    MR=M*N-2
    KR=(MR/2)*2
    IF(MR-KR)10,15,10
10  MR=MR-1
15  KR=KR/2+1
    DO 11 K=1,KR
    IBR(K)=0
11  CONTINUE
    IB1=1
    K=1
    IR=0
20  K1=K
    J=K1/M+1
    I=K1-(J-1)*M+1
    C=A(I,J)
40  I1=K1/N+1
    J1=K1-(I1-1)*N+1
    L=I1-1+(J1-1)*M
    IR=IR+1
    IF(K1-KR)25,25,35
25  IBR(K1)=IB1
35  IF(L-K)22,33,22
22  A(I,J)=A(I1,J1)
    I=I1
```

```
      J=J1
      K1=L
      GOTO  40
   33 A(I,J)=C
      WRITE(5,12)K
   12 FORMAT(1X, I6)
      IF(IR-MR)45,50,50
   45 K=K+1
      IF(IBR(K)-IB1)20,45,20
   50 RETURN
      END
```

追踪法二：据根按自然顺序置换的思想。设例中a_{11}，a_{12}，a_{13}，a_{14}，a_{15}都已转置妥当，要取a_{21}存在第6个单元中，到那里去找呢？在$A_{3\times5}$中观察，a_{12}送第2单元时，a_{21}可被保存在第4个单元中，当a_{14}送到第四单元中，a_{21}就被转送至第10个单元中保存，所以应该将第10单元的内容送第六个单元中，就完成了a_{21}是$A^T_{5\times3}$的第二列的第一个元素的事。

从上例中看出由第六个单元，如何找出第2，4，10个单元是转置的关键，其具体过程是从$K=1$开始，由(3)式得I,J，先将$A^T(J,I)$中元素保存在工作单元C，根据(2)算得L，当$L>K$时，表$A_{M\times N}$中的元素尚未置换过，可将$A(I,J)$送入$A^T(J,I)$中，並将C送入$A(I,J)$完成了$A^T(J,I)$与$A(I,J)$的对换。因为是A^T_{NXM}的排列，所以当$L<K$时，$A(I,J)$中已放置了A^T_{NXM}的元素，但原来的A_{IJ}已被保存在$A(I_1,J_1)$中了，将L代替K追踪由(3)算出I_1，J_1代入(2)式得L_1，若$L1<K$则重复(3)，(2)的计算，直至$L\geqslant K$为止，当$L=K$时表示元素A_{IJ}在$A^T_{N\times M}$及$A_{M\times N}$中已位于同一位置，就不必传送了，当K从1至$M*N-1$时，就完成了$A_{M\times N}$转置为$A^T_{N\times M}$的工作。整个子程序由$J=1$，N；$I=1$，M及$L\geqslant K$三个循环组成。

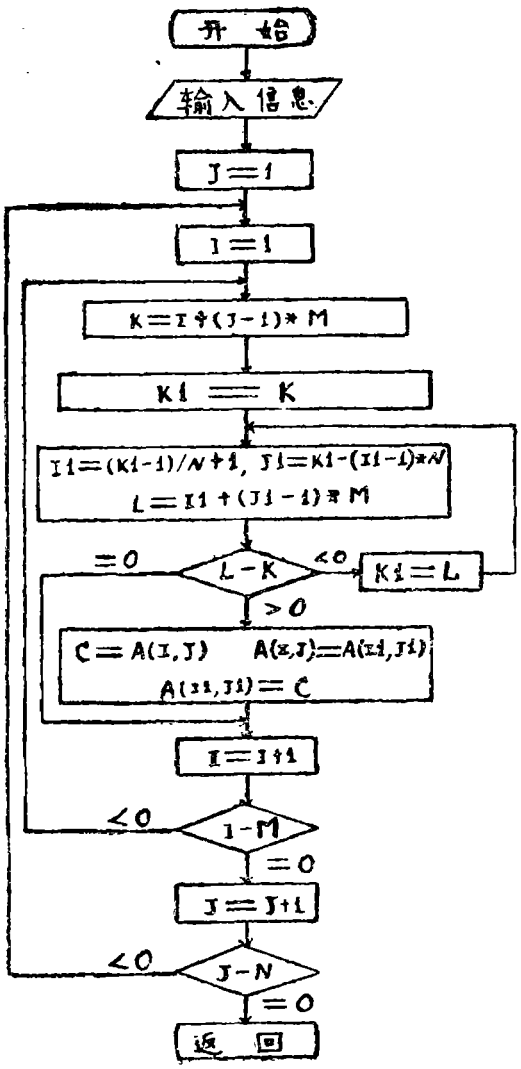
程序：

```
C  THIS IS THE SUBROUTINE FOR TRANSPOSE
C  OF A MATRIX  2
   SUBROUTINE TMA2 (M, N, A)
   DIMENSION A(M, N)
   Do 33 J=1, N
   Do 33 I=1, M
   K=I+(J-1)*M-1
   K1=K
31 I1=K1/N+1
   J1=K1-(I1-1)*N+1
   L=I1+(J1-1)*M-1
   IF(L-K)11, 33, 22
11 K1=L
```

```
      GoTo 31
22    C=A(I, J)
      A(I, J)=A(I1, J1)
      A(I1, J1)=C
33    CONTINUE
      RETURN
      END
```

说明　在主程序中，可以选取任意的M，N，只要不超出内存所允许的范围，然后应用$CALL\ TMA1(M,N,A)$或$CALL\ TMA2(M, N, A)$语句调用子程序，从子程序返回时，在A的存区中已转置成A^T的元素了。

框图：



例题 将 $A_{10\times30}$ 转置成 $A^T{}_{30\times10}$，占有同一存区，且按列排列存贮。

设
$$A_{10\times30}=\begin{vmatrix} 1.0 & 2.0\cdots\cdots\cdots\cdots29.0 & 30.0 \\ 2.0 & 3.0\cdots\cdots\cdots\cdots30.0 & 31.0 \\ \cdots\cdots & \cdots\cdots & \cdots\cdots \\ 9.0 & 10.0\cdots\cdots\cdots 37.0 & 38.0 \\ 10.0 & 11.0\cdots\cdots\cdots 38.0 & 39.0 \end{vmatrix}$$

在内存按列排列存贮，若没有建立数据库，在输入时容易出错，本例所选的数据是有规律性的，可利用电子计算机高速的优点，用计算结果的传送代替输入，可以节省时间。通过追踪法对换得到转置矩阵 $A^T{}_{30\times10}$

即
$$A^T{}_{30\times10}=\begin{vmatrix} 1.0 & 2.0\cdots\cdots\cdots\cdots9.0 & 10.0 \\ 2.0 & 3.0\cdots\cdots\cdots\cdots10.0 & 11.0 \\ 3.0 & 4.0\cdots\cdots\cdots\cdots11.0 & 12.0 \\ \cdots\cdots & \cdots\cdots & \cdots\cdots \\ 29.0 & 30.0\cdots\cdots\cdots 37.0 & 38.0 \\ 30.0 & 31.0\cdots\cdots\cdots 38.0 & 39.0 \end{vmatrix}$$

实现此过程的程序如下：

```
C   THIS IS MAIN PROGRAM
    DIMENSION A(10, 30)
    S=1.0
    C=0.0
    Do 15 J=1, 30
    C=C+S
    D=C
    Do 15 I=1, 10
    A(I, J)=D
    D=D+S
15  CONTINUE
    WRITE(5, 30) A
30  FORMAT(4X, 2HB:, /, (1X, 10F5.1/))
    CALL TMA1(10, 30, A)(或CALL TMA2(10,30,A))
    WRITE(5, 30)A
    STOP
    END
```

本文的程序全部在 PDP11/34机上通过。