第15卷 第4期 郑州工学院学报 Vol.15 No.4
1994年 12 月 Journal of Zhengzhou Institute of Technology Dec. 1994

行处理算法的存储技术*

李学相 安学庆

(郑州工学院数力系)

摘 要: 本文对行处理算法的存储技术进行了一些有益的探讨，给出了一种较为合理的行处理算法解大型稀疏线性方程组的存储分配方法。

关键词: 行处理，稀疏矩阵，内存分配

中图分类号: O241

行处理算法（简称 RA 方法）是求解大型稀疏线性方程组的一类实用的行之有效的迭代算法。它具有下述性质:

①对原始矩阵不作任何改变;

②不使用以矩阵为单位的任何运算;

③在每一迭代步中只用到矩阵的一个行;

④后一步迭代值仅需用前一步就能算出。

由于我们要解决的是大型稀疏问题，而行处理算法具有保持稀疏性、存贮量及运算量较小等特点，因而它是解决大型稀疏问题的一种有效算法，在实现这种算法时，稀疏矩阵的存储技术就显得相当重要。

1 稀疏矩阵的存储格式

我们用三个一维数组: IA，JA 和 A 来实现稀疏矩阵 M 的存储，这是一种压缩存贮格式。首先把 M 的非零元按行存储在实型数组 A 中，为了区分在一个行中的单个的非零元，还需知道每个元所在的列，整型数组 JA 由对应于 M 的非零元的列号组成，即如果 A (k) =M (I, J)，则 JA (k) =J; 此外，我们还需知道每个行从哪里开始以及每行有多长，整型数组 IA 由 M 中每个行的第一个非零元在 A 和 JA 中的号码组成，即如果 M (I, J) 是第 I 行的第一个非零元，并且 A (k) =M (I, J)，则 IA (I) =k，另外，把 IA (N+1) 定义为 JA 和 A 中最后一行最末一个元素之后的第一个位置，这样第 I 行中的元素的个数便可由 IA (I+1) −IA (I) 给出，并且知道第 I 行的非零元是依次存放在 A (IA (I))，A (IA (I) +1)，…，A (IA (I+1) −1) 之中的，而对应的列号则存放在

* 收稿日期: 1993−08−28

JA（IA（I）），JA（IA（I）+1），…，JA（IA（I+1）-1）之中的。

2　存储技术探讨

在微机上求解大型问题时，内存不足，这一方面是由于问题大，所需存储也大；另一方面也与内存组织不够好、利用不充分有关。下面我们就行处理算法的内存技术进行具有实际意义的探讨。

2.1　IBM-PC 系列机 DOS 运行 FORTRAN 的内存组织

IBM-PC 机的内存是按段划分的，每个段的大小最多可达 64K 字节（B），具有同一类名的所有段彼此相邻地装在一起。具有同一组名的所有段必须装在一个内存区域内，该区域的大小不得超过 64KB，也就是一个组中的段可由一个段寄存器存取。

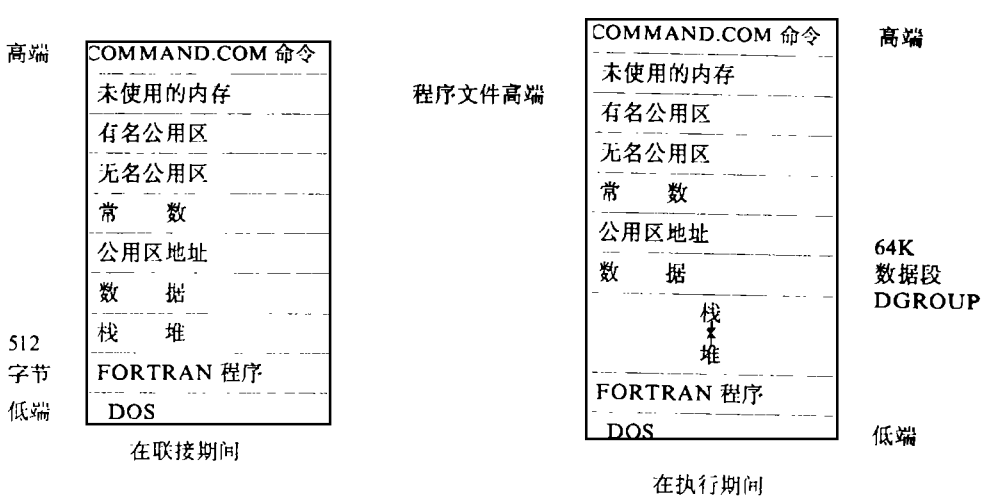
IBM FORTRAN 定义了一个单独的段，该段命名为 DGROUP，它用 DS 或 SS 段寄存器进行寻址。通常 DS 或 SS 的值是一样的，但 DS 的值可能暂时变化而指向某个其它的段，然后又变回原来的段值。SS 寄存器的值从不变化。对于长地址，例如 IBM FORTRAN 有名公用区，则使用 ES 段寄存器作寻址用。

组 DGROUP 内的内存区用来分配给所有的静态变量、内存中的常数、栈、堆、无名公用区和有名公用区的地址。有名公用区本身不在组 DGROUP 内，而是另在它们自己的段内。组 DGRUOP 有两部分：

①包含堆和栈的是可变长的低位部分。

②包含静态变量、常数、无名公用区和有名公用区地址的是定长的高位部分。

当程序装入后，固定的高位部分尽可能往上移动留出地方给低位部分。如果有足够的内存空间，组 DGROUP 可能扩充到 64KB，当然也只能扩充到 64KB。



存储组织图

关于内存组织及连接时连接图象的转移请见存储组织图。

下面我们根据行处理算法的特点，结合内存的组织寻求一种较为合理的行处理算法解大型稀疏线性方程组的存储分配技术。

2.2 内存分配技术

在这里我们以前面介绍的（IA，JA，A）一维压缩存储，以 6000× 6000 的方程，每行大约 10 个非零元为例进行探讨。

设机器的内存为 640KB（一般 PC／XT，AT、286、386 机内存均在 64KB 或 640KB 以上），整数以 2 字节存储，实数以 4 字节存储。在行处理算法中主要存储的数组有下列六个（以 6000 阶为例）

IA（6000），JA（60000），A（60000），B（6000），C（3000），X（6000）。

算法中的三个模块即单位化、排序、求解都可以独立形成执行文件，得到各自的结果而让结果存入外层，而下一个需要用到其结果的模块可以从外层读入，这样可以使得 FORTRAN 的 EXE 文件减少占内存的大小。根据我们的编程，最大的求解模块形成的 EXE 文件不到 64K。若以 DOS3.X 版本为操作系统，那么 DOS.和 Command.com 共占内存有 64K，这样可供文件执行的内存有 8× 64K。对于这些内存的合理组织、分配可以至少解 6000 阶的方程而不需要内外层的交换，这是一般其它的方法不可能实现的。

作为行处理算法的一个最大特点就是算法简单，工作变量少，程序嵌套简单，逻辑嵌套（即 IF／THEN／ELSE／ENDIF），也较为简洁。因此对于固化数据段 DGROUP 中除了静态变量、普通常数、以及用于堆栈用的空间外，有一定的剩余空间．我们若把 X、C 用无名公用区放入 DGROUP 段中，那么 X，C 共占 36K 的空间，而这时在 DGROUP 中仍还有 28K 空间，这足够行处理算法使用了。剩下的 IA，JA，A，B，用有名公用区进行存储、传送、它们总共可用的空间为 7× 64KB。

我们知道每一有名公用区不能超过 64K，因此 A 需分成 4 块，JA 需分成 2 块，B 和 IA 可以合用一块。具体为:

A 分为 4 块，每块 15000 个实数。

JA 分为 2 块，每块 30000 个整数;

IA，B 合为一个块为 6000 个整数、6000 个实数。

这反应在行处理算法的具体执行过程中为: 把矩阵 M 按行划为 4 块，分别称为 A_1，A_2，A_3，A_4，那么相应的 A_1，A_2 共用块 JA_1，A_3，A_4 共用 JA_2，这使得程序稍稍复杂一点，但是经过这样的处理之后，完全不需要内外存交换，而且充分利用了微机有限的内存，效率极高。因此这种分配技术的使用和实施是非常必要和可取的。

当然我们上面对内存的存储分析是非常粗糙的，可以看出，上面的分析对内存的分配仍有充分的余地。如果加之对实际问题的具体考虑，对于更高阶方程用行处理算法不需内外存交换也是完全可以实现的。

如果方程高于 6000 阶，内存无法满足的话，可以看出对于 12000 阶以内的方程组每一次迭代最多也只进行两次内外存的交换。实际也就是从外存中读，而并不需要向外存写。具体实施如下:

设 $A=\begin{pmatrix}A_1\\A_2\end{pmatrix}A_1$，$A_2$ 的行数均小于6000，我们把 B、X、IA 均可全部放入内存，在每一步迭代时，先把 A_1 与其相应的 $JA(K)$ ′$A(K)$ 读入内存，进行逐次的计算，然后释放 JA、A 所占的空间，把 A_2 与其相应的 $JA(K)$，$A(K)$ 读入内存，重复进行上述工作，即可完成一次迭代的全过程。

由于矩阵的稀疏性，因此由方程决定的诸超平面中两两正交的可能性极大，即有大量的两两正交的超平面。这使行处理算法的收敛速度很快，迭代次数也很少，内外存交换的次数是可以接受的。

参　考　文　献

1 美国耶鲁大学.稀疏矩阵程序包ysmp.1983

2 熊西文.数值代数.华中工学院出版社.1985

3 Y. Censor. Row-Action method for huge and sparse systems and their applications. SIAM Review 23(1981) pp.444-466

4 Sergio Pissanetzky. Sparse Matrix Technology. Academic Press 1984

A Memory Technique for the Row-Action Method

Li Xuexiang　　An Xueqing
(Zhengzhou Institute of Technology)

Abstract: In this paper, a memory technique for the row-action method is given. Experiments show that this method is efficient in solving huge and sparse linear systems.

Keywords: Row-Action, SParse matrix, Memory