矩阵乘法的一个最佳算法*

蒋昌俊　　吴哲辉

（山东矿业学院，泰安）

关键词　　算法、矩阵乘法、运算次数、阶

一、引　　言

矩阵乘法是线性代数中常见的问题之一，许多数值计算问题都包含着矩阵乘法的计算．因此，降低矩阵乘法算法的时间复杂度问题，多年来一直引起算法研究者们的高度重视．

1969 年，Strassen 提出了一个时间复杂度为 $O(n^{\log_2 7})$ 的矩阵乘法算法[1]，第一次突破了 $O(n^3)$ 的界限，被誉为"在代数复杂性理论中最激动人心的结果"[2]．以后，又出现了一系列新的矩阵乘法算法[3-6]． 据了解，迄今最好的算法是 1981 年由 Pan 提出的，其时间复杂度为 $O(n^{2.494})$[7]．

矩阵乘法的算法最少需要多少时间量呢？ Baase 指出，对于两个 n 阶方阵的乘法来说，n^2 阶的乘除法运算是必须的．但他认为，达到这个下界的算法大概是不存在的[8]．

本文给出矩阵乘法的一个算法．该算法适用于有理数矩阵．对于两个 n 阶方阵相乘，即 (n,n,n) 问题，只需要 n^2 阶的运算次数．对于 n 行 m 列矩阵和 m 行 l 列矩阵的乘法，即 (n,m,l) 问题，运算次数的阶为 $O(m(n+l))$． 这样，本文提出的算法其运算次数的阶已达到了理论下界．

二、非负整数矩阵乘法的算法

已知两个非负整数矩阵

$$A=[a_{ik}]_{n\times m},$$
$$B=[b_{kj}]_{m\times l},$$

求矩阵

$$C=[c_{ij}]_{n\times l},$$

使得

$$c_{ij}=\sum_{k=1}^{m}a_{ik}b_{kj}.\quad i=1,2,\cdots,n;\ j=1,2,\cdots,l.$$

对于这个问题，可以通过下面的算法步骤来实现．

第 1 步：计算

$$a=\max\{a_{ik}\mid i\in\{1,2,\cdots,n\},k\in\{1,2,\cdots,m\}\},$$
$$b=\max\{b_{kj}\mid k\in\{1,2,\cdots,m\},j\in\{1,2,\cdots,l\}\},$$

本文 1988 年 6 月22日收到．1988 年 9 月19日收到修改稿．

* 国家自然科学基金资助项目．

$$x = mab + 1,$$
$$y = nmabx^{n-1} + 1.$$

第 2 步：对 $k = 1, 2, \cdots, m$，用 Horner 方法计算

$$a_k = \sum_{i=1}^{n} a_{ik} x^{n-i},$$

$$b_k = \sum_{j=1}^{l} b_{kj} y^{l-j}.$$

第 3 步：计算 $c = \sum_{k=1}^{m} a_k b_k$.

第 4 步：做除法．依次求商 $d_1, d_2, \cdots, d_{l-2}, r_l$ 和余数 $r_1, r_2, \cdots r_{l-1}$，使满足：

$$c = d_1 y + r_1, \quad 0 \leqslant r_1 < y$$
$$d_{j-1} = d_j y + r_j, \quad 0 \leqslant r_j < y \quad j = 2,\ 3,\ \cdots,\ l-2$$
$$d_{l-2} = r_l y + r_{l-1}, \quad 0 \leqslant r_{l-1} < y.$$

令
$$c_j = r_{l-j+1}, \quad j = 1, 2, \cdots, l.$$

第 5 步：做除法．对 $j = 1, 2, \cdots, l$，依次求商 $k_{j1}, k_{j2}, \cdots, k_{j(n-2)}, r_{jn}$ 和余数 $r_{j1}, r_{j2}, \cdots, r_{j(n-1)}$，使满足：

$$c_j = k_{j1} x + r_{j1}, \quad 0 \leqslant r_{j1} < x,$$
$$k_{j(i-1)} = k_{ji} x + r_{ji}, \quad 0 \leqslant r_{ji} < x, \quad i = 2, 3, \cdots, n-2$$
$$k_{j(n-2)} = r_{jn} x + r_{j(n-1)}, \quad 0 \leqslant r_{j(n-1)} < x,$$

令
$$c_{ij} = r_{j(n-i+1)}, \quad i = 1, 2, \cdots, n;\ j = 1, 2, \cdots, l,$$

于是求得积矩阵

$$C = [c_{ij}]_{n \times l}.$$

上面提出的算法主要涉及乘法、除法、加法和比较运算．我们可以按算法步骤的顺序分别估算一下上述四种运算的运算次数，然后综合五个步骤，可以得出各种运算的次数分别为

比较运算	$m(n + l) - 2$ 次，
乘法运算	$m(n + l - 1) + n + 2$ 次，
加法运算	$m(n + l - 1) + 1$ 次，
除法运算	$nl - 1$ 次

可见，对于非负整数矩阵的 (n, m, l) 问题，本节给出的算法，其运算次数的阶为 $O(m(n + l))$.

三、算法的正确性证明

在本节中，我们证明上节给出的算法的正确性．注意 a_{ik} 和 b_{kj} 都是非负整数，于是 x，y，a_k 和 b_k 等都是正整数．我们有

$$c = \sum_{k=1}^{m} a_k b_k$$
$$= \sum_{k=1}^{m} \left(\sum_{i=1}^{n} a_{ik} x^{n-i} \right) \cdot \left(\sum_{j=1}^{l} b_{kj} y^{l-j} \right)$$

$$= \sum_{j=1}^{l} \sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{kj} x^{n-i} y^{l-j}.$$

由于 $\forall i \in \{1,2,\cdots,n\}$，$\forall j \in \{1,2,\cdots,l\}$，都有

$$\sum_{k=1}^{m} a_{ik} b_{kj} \leqslant mab < x,$$

且 $\forall j \in \{1,2,\cdots,l\}$，都有

$$\sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{kj} x^{n-1} \leqslant nmabx^{n-1} < y,$$

所以，由 $c = d_1 y + r_1$ 可得

$$r_1 = \sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{kl} x^{n-i},$$

$$d_1 = \sum_{j=1}^{l-1} \sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{kj} x^{n-i} y^{l-j-1}.$$

进一步可以得到

$$r_j = \sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{k(l-j+1)} x^{n-i},\ j = 2,3,\cdots,l,$$

于是

$$c_j = \sum_{i=1}^{n} \sum_{k=1}^{m} a_{ik} b_{kj} x^{n-i},\ j = 1,2,\cdots,l.$$

同样理由，我们可以得到

$$r_{ji} = \sum_{k=1}^{m} a_{(n-i+1)k} b_{kj},\ i = 1,2,\cdots,n;\ j = 1,2,\cdots,l,$$

所以

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj},\ i = 1,2,\cdots,n;\ j = 1,2,\cdots,l.$$

四、讨　论

1. 在第二节提出的算法中，令 $m = l = n$，便可用于两个 n 阶方阵的乘法，即 (n, n, n) 问题．这时，算法涉及的各种运算的次数为

比较运算　$2n^2 - 2$ 次，

乘法运算　$2n^2 + 2$ 次，

加法运算　$2n^2 - n + 1$ 次，

除法运算　$n^2 - 1$ 次．

因此，对于非负整数矩阵的 (n,n,n) 问题，算法所需要的运算次数的阶为 $O(n^2)$．

2. 不难把算法推广到 a_{ik} 和 b_{kj} 为任意整数的情况．这时可以把 A、B 两个矩阵写成

$$\begin{aligned} A &= A^+ - A^- \\ &= [a_{ik}^+]_{n\times m} - [a_{ik}^-]_{n\times m}, \\ B &= B^+ - B^- \\ &= [b_{kj}^+]_{m\times l} - [b_{kj}^-]_{m\times l}, \end{aligned}$$

其中 a_{ik}^+，a_{ik}^-，b_{kj}^+ 和 b_{kj}^- 都是非负整数。而

$$A \cdot B = (A^+ - A^-) \cdot (B^+ - B^-)$$
$$= (A^+ \cdot B^+ + A^- \cdot B^-) - (A^- \cdot B^+ + A^+ \cdot B^-),$$

这样，整数矩阵乘法的运算次数等于非负整数矩阵乘法运算次数的四倍再加上 $3nl$ 次加减运算。

3. 如果 a_{ik} 和 b_{kj} 为有理数，不妨假设它们以既约分数的形式出现，且分母都是正整数。我们可以求出所有元素的分母的最小公倍数 t，由公式

$$C = (tA \cdot tB)/t^2$$

化为整数矩阵 tA 和 tB 的乘法。为了求所有元素的分母的最小公倍数，可以先求两个分母的最小公倍数，然后再求该最小公倍数与第三个分母的最小公倍数，……直到遍历所有元素的分母。而求两个数的最小公倍数，可以通过辗转相除法求最大公约数来实现。上述工作所需要的运算，不会增加总运算次数的阶。

4. 本文提出的算法，由于中间量 a_k，b_k 和 c 等可能是很大的整数，在计算机上实现会遇到一定的困难。但本文的结果至少从理论上说明，运算次数的阶为 $O(n^2)$（对于 (n,m,l) 问题来说），阶为 $O(m(n+l))$ 的矩阵乘法算法是存在的。

参 考 文 献

[1] Strassen, V., *Numer. Math.*, 13 (1969), 345—356.
[2] Pan, V.Ya, *SIAM J. of Compt.*, 9 (1980), 2:321—342.
[3] Pan, V. Ya, *Comp. & Math. with Appls.*, 7 (1981), 73—125.
[4] Bini, D., *Information Processing Letters*, 8 (1979), 5: 234—235.
[5] Bini, D., *Information Processing Letters*, 8 (1979),5:46—47.
[6] Schonhage, A., *SIAM J. of Compt.*, 10 (1981), 3: 435—355.
[7] 曹新谱，算法设计与分析，湖南科技出版社，1984，53—55.
[8] S. 巴斯，计算机算法：设计与分析引论（朱洪等译），复旦大学出版社，1985，233—240.