FAST PARALLEL ALGORITHMS FOR COMPUTING GENERALIZED INVERSES A^+ AND A^+_{MN}

WANG GUO-RONG(王国荣) LU SEN-QUAN(陆森泉)

(*Shanghai Teachers' University, Shanghai, China*)

Abstract

The parallel arithmetic complexities for computing generalized inverse A^+, computing the minimum-norm least-squares solution of $Ax=b$, computing order $m+n-r$ determinants and finding the characteristic polynomials of order $m+n-r$ matrices are shown to have the same grawth rate. Algorithms are given that compute A^+ and A^+_{MN} in $O(\log r\cdot\log n+\log m)$ and $O(\log^2 n+\log m)$ steps using a number of processors which is a ploynomial in m, n and r ($A\in R_r^{m\times n}$, $r=\text{rank } A$).

§ 1. Introduction

Let $I(n)$, $E(n)$, $D(n)$, $P(n)$ denote the parallel arithmetic complexities of inverting order n matrices, solving a system of n linear equations in n unknowns, computing order n determinants and finding the characteristic polynomials of order n matrices respectively. Then Csanky gave an important theoretical result [1]:

Theorem 1. $I(n)=O(f(n))\Leftrightarrow E(n)=O(f(n))\Leftrightarrow D(n)=O(f(n))\Leftrightarrow P(n)=O(f(n))$.

He also gives algorithms that compute these problems in $O(\log^2 n)$ steps using a number of processors which is a polynomial in n (n is the order of the matrix of the problem).

Let $A\in R_r^{m\times n}$, $r=\text{rank } A$. In this paper, we give two parallel algorithms for computing A^+ and A^+_{MN} respectively. The one for A^+ is based on Decell's method in [2], and the one for A^+_{MN} is a generalization of Decell's method in [3].

The parallel arithmetic complexities for computing the generalized inverse A^+, computing the minimum-norm least-squares solution of $Ax=b$, computing order $m+n-r$ determinants and finding the characteristic polynomials of order $m+n-r$ matrices are shown to have the same growth rate.

§ 2. The Parallel Algorithm for Computing A^+

Let $A\in R_r^{m\times n}$. Then there is a unique matrix $X\in R_r^{n\times m}$ satisfying

$$AXA=A,\ XAX=X,\ (AX)^T=AX,\ (XA)^T=XA.$$

This X is called the M-P inverse of A and is denoted by $X=A^+$.

In [2], Decell gave a finite algorithm for computing A^+. We rewrite it as follows:

* Received March 21, 1987.

Algorithm 1. (1) Parallelly compute $B=A^TA$.

(2) Parallelly compute $B^k=(b_{ij}^{(k)})$, $k=1, 2, \cdots, r$.

(3) Let $\lambda_1, \lambda_2, \cdots, \lambda_n$ denote the roots of the characteristic polynomial $f(\lambda)$ of B. Let

$$s_k=\sum_{i=1}^{n}\lambda_i^k, \quad k=1, 2, \cdots, r.$$

Parallelly compute

$$s_k=\mathrm{tr}(B^k)=\sum_{i=1}^{n}b_{ii}^{(k)}, \quad k=1, 2, \cdots, r.$$

(4) Let the characteristic polynomial $f(\lambda)$ of B be

$$f(\lambda)=\det(\lambda I-B)=\lambda^n+c_1\lambda^{n-1}+\cdots+c_n.$$

From the Newton formula

$$s_k+c_1s_{k-1}+c_2s_{k-2}+\cdots+c_{k-1}s_1+kc_k=0, \quad k\leqslant n$$

we have

$$\begin{bmatrix} 1 & & & & \\ s_1 & 2 & & & \\ s_2 & s_1 & 3 & & \\ \vdots & \vdots & & \ddots & \\ s_{r-1} & s_{r-2} & \cdots & s_1 & r \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_r \end{bmatrix}=-\begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_r \end{bmatrix}.$$

Parallelly compute the solution of the above triangular system.

(5) Parallelly compute

$$A^+=-((A^TA)^{r-1}+c_1(A^TA)^{r-2}+\cdots+c_{r-1}I)A^T/c_r. \tag{2.1}$$

Theorem 2. *Let $A\in R_r^{m\times n}$, and $GI(m, n)$ denote the parallel arithmetic complexity for computing the* M-P *inverse A^+. Then*

$$GI(m, n)=\log r(\log n+7/2)+(1/2)\log^2 r+2\log n+\log m+4=0(f(m, n, r))$$

and the number of processors used in the algorithm is

$$cp=\begin{cases} n^3r/2, & m<nr/2, \\ mn^2, & m\geqslant nr/2. \end{cases}$$

Proof. (1) Parallel computation of $B=A^TA$ takes $T_1=\log m+1$ steps and $cp_1=mn^2$ processors.

(2) Parallel computation of B^k $(k=1, 2, \cdots, r)$ takes $T_2=\log r(\log n+1)$ steps and $cp_2=n^3r/2$ processors.

(3) Parallel computation of s_k $(k=1, 2, \cdots, r)$ takes $T_3=\log n$ steps and $cp_3=rn/2$ processors.

(4) Parallel computation of $c_k(k=1, 2, \cdots, r)$ takes $T_4=(1/2)\log^2 r+(3/2)\log r$ steps and $cp_4=O(r^3)$ processors.

(5) Since $B^2, \cdots, B^{r-1}$ are already available, parallel computation of A^+ takes $T_5=\log r+\log n+3$ steps and $cp_5=n^2m$ processors.

Thus

$$cp=\max_{1\leqslant i\leqslant 5} cp_i=\begin{cases} n^3r/2, & m<nr/2, \\ mn^2, & m\geqslant nr/2. \end{cases}$$