文章编号:1000-0887(2013)03-0240-12

ⓒ 应用数学和力学编委会,ISSN 1000-0887

一类特殊矩阵方程的并行预处理变形共轭梯度算法*

曹方颖[1], 吕全义[1], 谢公南[2]

(1. 西北工业大学 应用数学系,西安 710129;
2. 西北工业大学 机电学院,工程仿真与宇航计算技术联合实验室,西安 710072)

(我刊编委谢公南来稿)

摘要: 研究了求解一类矩阵方程 $\boldsymbol{AXB}=\boldsymbol{C}$,提出了一种并行预处理变形共轭梯度法.该方法给出一种迭代法的预处理模式.首先给出的预处理矩阵是严格对角占优矩阵,构造并行迭代求解预处理矩阵方程的迭代格式,进而使用变形共轭梯度法并行求解.通过数值试验,预处理变形共轭梯度法与直接使用变形共轭梯度法相比较,该算法不仅有效提高了收敛速度,而且具有很高的并行性.

关 键 词: 矩阵方程; 变形共轭梯度法; 预处理矩阵; 并行性
中图分类号: O246 **文献标志码:** A
DOI: 10.3879/j.issn.1000-0887.2013.03.004

引 言

矩阵方程在诸如计算数学、控制论、工程力学等领域中都经常会遇到,因此研究线性矩阵方程的求解很重要.例如,1955 年 Penrose[1] 得到了 $\boldsymbol{AXB}=\boldsymbol{C}$ 有解的充要条件和通解的表达式;1970 年 Lancaster[2] 利用 Kornecker 乘积和拉直映射也得到了它的一般解的条件和显式解.1984 年,Mitra[3] 利用空间有关理论及秩的相关不等式,给出了矩阵方程组 $\boldsymbol{AX}=\boldsymbol{C}$, $\boldsymbol{XB}=\boldsymbol{D}$ 的极小秩及其它定秩的通解;Uhlig[4] 于 1987 年给出了矩阵方程 $\boldsymbol{AX}=\boldsymbol{B}$ 的可能秩的解;Xiao 等[5] 于 2009 年研究了矩阵方程 $\boldsymbol{AX}=\boldsymbol{B}$ 的对称最小秩解和最佳逼近解.

一般求解线性矩阵方程的直接法[6] 主要有 LU 分解法,Cholesky 分解法,QR 分解法等;迭代法[6] 有 Jacobi 迭代法,Gauss-Seidel 迭代法,JGS(Jacob Gauss-Seidel)迭代法,SOR(successive over relaxation)迭代法,SSOR 迭代法,SAOR 迭代法,参数迭代法等.对于求解大型线性矩阵方程以共轭梯度法和变形共轭梯度法为主,其具有存储量少,计算量少和适合并行计算等优点.文献[6]研究了关于求解矩阵方程极小范数解或极小范数最小二乘解的变形共轭梯度法(modified conjugate gradient,简称 MCG 法).MCG 法作为最基本的 Krylov 子空间方法,易于并行化.MCG 法的收敛速度与系数矩阵的条件数紧密相关,条件数愈小,收敛性愈好,该算法可

* **收稿日期:** 2013-01-29
基金项目: 国家自然科学基金资助项目(11202164);陕西省自然科学基金资助项目(2009JM1008)
作者简介: 曹方颖(1986—),女,西安人,硕士生(E-mail:caofangying1987@163.com);
吕全义(1963—),女,沈阳人,副教授(通讯作者. E-mail:luquan@nwpu.edu.cn).

以在很少的几步就会获得高精度的近似解.但当系数矩阵的条件数很大时,收敛速度就很慢.于是出现了预处理变形共轭梯度法[6-7](predisposed modified conjugate gradient,简称 PMCG 法),它是通过适当的预处理方法引入预处理矩阵 $\boldsymbol{M}$,使矩阵的特征值分布更为集中,降低矩阵条件数,以达到提高收敛速度的目的.

本文针对矩阵方程极小范数解或极小范数最小二乘解问题提出了一种并行解决方案.本文的方法是为了降低求解预条件矩阵方程 $\boldsymbol{AM}^{-1}=\tilde{\boldsymbol{A}}$, $\boldsymbol{BM}^{-1}=\tilde{\boldsymbol{B}}$ 的计算复杂度,对预处理矩阵方程采取若干次迭代方法来求解,而且在求解过程中只在相邻处理机间有通信.此方法不仅有效地改变了原矩阵方程的条件数,又具有很好的并行性.最后在联想深腾 1800 集群上进行了数值试验,并与传统的变形共轭梯度法的计算结果进行了比较.文中用 $\boldsymbol{A}\otimes\boldsymbol{B}$ 表示矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 的 Kronecker 积, $\overline{\mathrm{vec}}(\boldsymbol{X})$ 表示将矩阵 $\boldsymbol{X}$ 按行拉直构成的列向量.定义同型矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 的内积为 $[\boldsymbol{A},\boldsymbol{B}]=\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{B})$,由此导出矩阵的 Frobenius 范数 $\|\boldsymbol{A}\|=\sqrt{[\boldsymbol{A},\boldsymbol{A}]}$.

1　MCG 算法

设线性矩阵方程

$$\boldsymbol{AXB}=\boldsymbol{C}, \tag{1}$$

其中, $\boldsymbol{A}$ 为 $m\times n$ 矩阵, $\boldsymbol{X}$ 为 $n\times q$ 矩阵, $\boldsymbol{B}$ 为 $q\times l$ 矩阵, $\boldsymbol{C}$ 为 $m\times l$ 矩阵.

记 $\overline{\mathrm{vec}}(\boldsymbol{X})=\boldsymbol{x}$, $\overline{\mathrm{vec}}(\boldsymbol{C})=\boldsymbol{c}$,将矩阵方程 $\boldsymbol{AXB}=\boldsymbol{C}$ 通过按行拉直转化为线性方程组

$$(\boldsymbol{A}\otimes\boldsymbol{B}^{\mathrm{T}})\boldsymbol{x}=\boldsymbol{c}. \tag{2}$$

当矩阵方程(1)有解时,建立求解线性方程组(2)的极小范数解的 MCG 算法[6]如下:

第一步　任意给定初始向量 $\boldsymbol{x}^{(0)}$,置 $k:=1$, 计算

$$r^{(0)}=\boldsymbol{c}-(\boldsymbol{A}\otimes\boldsymbol{B}^{\mathrm{T}})\boldsymbol{x}^{(0)},\ z^{(0)}=\boldsymbol{c}-(\boldsymbol{A}\otimes\boldsymbol{B}^{\mathrm{T}})^{\mathrm{T}}\boldsymbol{r}^{(0)}.$$

第二步　若 $\boldsymbol{r}^{(k)}=\boldsymbol{0}$,或者 $\boldsymbol{r}^{(k)}\neq\boldsymbol{0}$ 而 $z^{(k)}=\boldsymbol{0}$, 停止;否则,计算

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}+\frac{\|\boldsymbol{r}^{(k)}\|^2}{\|\boldsymbol{z}^{(k)}\|^2}\boldsymbol{z}^{(k)}.$$

第三步　计算

$$\boldsymbol{r}^{(k+1)}=\boldsymbol{c}-(\boldsymbol{A}\otimes\boldsymbol{B}^{\mathrm{T}})\boldsymbol{x}^{(k+1)},\ z^{(k+1)}=(\boldsymbol{A}\otimes\boldsymbol{B}^{\mathrm{T}})^{\mathrm{T}}\boldsymbol{r}^{(k+1)}+\frac{\|\boldsymbol{r}^{(k+1)}\|^2}{\|\boldsymbol{r}^{(k)}\|^2}z^{(k)}.$$

第四步　置 $k:=k+1$, 转第二步.

记

$$\boldsymbol{x}^{(k)}=\overline{\mathrm{vec}}(\boldsymbol{X}^{(k)}),\ \boldsymbol{r}^{(k)}=\overline{\mathrm{vec}}(\boldsymbol{R}^{(k)}),\ z^{(k)}=\overline{\mathrm{vec}}(\boldsymbol{Z}^{(k)}),$$

将以上算法写为下面的矩阵形式.

第一步　任给初始矩阵 $\boldsymbol{X}^{(0)}$,置 $k:=0$, 计算

$$\boldsymbol{R}^{(0)}=\boldsymbol{C}-\boldsymbol{AX}^{(0)}\boldsymbol{B},\ \boldsymbol{Q}^{(0)}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{R}^{(0)}\boldsymbol{B}^{\mathrm{T}},\ \boldsymbol{Z}^{(0)}=\boldsymbol{Q}^{(0)}.$$

第二步　若 $\boldsymbol{R}^{(k)}=\boldsymbol{O}$ 或者 $\boldsymbol{R}^{(k)}\neq\boldsymbol{O}$ 而 $[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]=0$, 停止计算;否则,计算

$$\boldsymbol{X}^{(k+1)}=\boldsymbol{X}^{(k)}+\frac{\|\boldsymbol{R}^{(k)}\|^2}{\|\boldsymbol{Z}^{(k)}\|^2}\boldsymbol{Z}^{(k)}.$$

第三步　计算 $\boldsymbol{R}^{(k+1)}=\boldsymbol{C}-\boldsymbol{AX}^{(k+1)}\boldsymbol{B}$, $\boldsymbol{Q}^{(k+1)}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{R}^{(k+1)}\boldsymbol{B}^{\mathrm{T}}$,

$$\boldsymbol{Z}^{(k+1)}=\boldsymbol{Q}^{(k+1)}+\frac{\|\boldsymbol{R}^{(k+1)}\|^2}{\|\boldsymbol{R}^{(k)}\|^2}\boldsymbol{Z}^{(k)}.$$

第四步　置 $k:=k+1$, 转第二步.

当线性矩阵方程(1)无解时,我们求方程的极小范数最小二乘解,即求解方程(1)的正规矩阵方程 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{AXBB}^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{CB}^{\mathrm{T}}$ 的极小范数解,MCG 算法的矩阵形式如下:

第一步　任给初始矩阵 $\boldsymbol{X}^{(0)}$,置 $k:=0$, 计算

$$\boldsymbol{R}^{(0)}=\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{C}-\boldsymbol{AX}^{(0)}\boldsymbol{B})\boldsymbol{B}^{\mathrm{T}},\ \boldsymbol{Q}^{(0)}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{AR}^{(0)}\boldsymbol{BB}^{\mathrm{T}},\ \boldsymbol{Z}^{(0)}=\boldsymbol{Q}^{(0)}.$$

第二步　若 $\boldsymbol{R}^{(k)}=\boldsymbol{O}$ 或者 $\boldsymbol{R}^{(k)}\neq\boldsymbol{O}$ 而 $[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]=0$, 停止计算;否则,计算

$$\boldsymbol{X}^{(k+1)}=\boldsymbol{X}^{(k)}+\frac{\|\boldsymbol{R}^{(k)}\|^2}{\|\boldsymbol{Z}^{(k)}\|^2}\boldsymbol{Z}^{(k)}.$$

第三步　计算 $\boldsymbol{R}^{(k+1)}=\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{C}-\boldsymbol{AX}^{(k+1)}\boldsymbol{B})\boldsymbol{B}^{\mathrm{T}},\boldsymbol{Q}^{(k+1)}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{AR}^{(k+1)}\boldsymbol{BB}^{\mathrm{T}}$,

$$\boldsymbol{Z}^{(k+1)}=\boldsymbol{Q}^{(k+1)}+\frac{\|\boldsymbol{R}^{(k+1)}\|^2}{\|\boldsymbol{R}^{(k)}\|^2}\boldsymbol{Z}^{(k)}.$$

第四步　置 $k:=k+1$, 转第二步.

2　PMCG 算法

选取预条件矩阵 $\boldsymbol{M}_1$ 使得 $\boldsymbol{AM}_1^{-1}$ 尽量接近矩阵 $\boldsymbol{A}$ 的等价标准形,预条件矩阵 $\boldsymbol{M}_2$ 使得 $\boldsymbol{BM}_2^{-1}$ 尽量接近矩阵 $\boldsymbol{B}$ 的等价标准形,如果不考虑 $\boldsymbol{AM}_1^{-1},\boldsymbol{BM}_2^{-1}$ 的精确解,采用迭代算法计算 $\boldsymbol{AM}_1^{-1}$, $\boldsymbol{BM}_2^{-1}$ 的近似矩阵,只要迭代法适合并行计算,那么对于 $\boldsymbol{M}_1,\boldsymbol{M}_2$ 的选取就简单得多;由于 Jacobi 迭代算法适合并行计算,所以采用该算法,又 Jacobi 迭代算法收敛的充分条件为系数矩阵为严格对角占优,故若 $m<n$,选取预处理矩阵 $\boldsymbol{M}_1$ 为 n 阶可逆且含矩阵 $\boldsymbol{A}$ 的带状部分,若 $q<l$,选取预处理矩阵 $\boldsymbol{M}_2$ 为 l 阶可逆且含矩阵 $\boldsymbol{B}$ 的带状部分,即

$$\boldsymbol{M}_1=\begin{pmatrix}
a_{11} & \cdots & a_{1r_1} & & & & & & \\
& \ddots & & \ddots & & & & & \\
a_{r_s1} & & \ddots & & \ddots & & & & \\
& \ddots & & \ddots & & \ddots & & & \\
& & a_{m,m-r_m} & \cdots & a_{mm} & \cdots & a_{m,m+r_m} & & \\
& & & & & 1 & & & \\
& & & & & & \ddots & & \\
& & & & & & & & 1
\end{pmatrix},$$

$$\boldsymbol{M}_2=\begin{pmatrix}
b_{11} & \cdots & b_{1r_1} & & & & & & \\
& \ddots & & \ddots & & & & & \\
b_{r_s1} & & \ddots & & \ddots & & & & \\
& \ddots & & \ddots & & \ddots & & & \\
& & b_{q,q-r_q} & \cdots & b_{qq} & \cdots & b_{q,q+r_q} & & \\
& & & & & 1 & & & \\
& & & & & & \ddots & & \\
& & & & & & & & 1
\end{pmatrix}.$$

每行的半带宽 r_i 的选取要求矩阵 $\boldsymbol{M}_1,\boldsymbol{M}_2$ 是严格对角占优矩阵,若矩阵 $\boldsymbol{A},\boldsymbol{B}$ 主对角元有零元,

在矩阵 $\boldsymbol{M}_1,\boldsymbol{M}_2$ 的相应位置中取 1.

在求解 $\boldsymbol{A}\boldsymbol{M}_1^{-1}$ 的近似值时,设 $\tilde{\boldsymbol{A}}=\boldsymbol{A}\boldsymbol{M}_1^{-1}$ 采用的迭代格式是

任选 $\tilde{\boldsymbol{A}}^{(0)}$

for　$h=0,\cdots,l-1$

$$\tilde{\boldsymbol{A}}^{(h+1)}=(\tilde{\boldsymbol{A}}^{(h)}(\boldsymbol{L}_1+\boldsymbol{U}_1)+\boldsymbol{A})\boldsymbol{D}_1^{-1}$$

end

$$\tilde{\boldsymbol{A}}=\tilde{\boldsymbol{A}}^{(l)}$$

其中, $\boldsymbol{M}_1=\boldsymbol{D}_1-\boldsymbol{L}_1-\boldsymbol{U}_1,\boldsymbol{D}_1,-\boldsymbol{L}_1,-\boldsymbol{U}_1$ 分别是矩阵 $\boldsymbol{M}_1$ 的对角矩阵、严格下三角矩阵与严格上三角矩阵.

在求解 $\boldsymbol{B}\boldsymbol{M}_2^{-1}$ 的近似值时,设 $\tilde{\boldsymbol{B}}=\boldsymbol{B}\boldsymbol{M}_2^{-1}$ 采用的迭代格式为

任选 $\tilde{\boldsymbol{B}}^{(0)}$

for　$h=0,\cdots,l-1$

$$\tilde{\boldsymbol{B}}^{(h+1)}=(\tilde{\boldsymbol{B}}^{(h)}(\boldsymbol{L}_2+\boldsymbol{U}_2)+\boldsymbol{B})\boldsymbol{D}_2^{-1}$$

end

$$\tilde{\boldsymbol{B}}=\tilde{\boldsymbol{B}}^{(l)}$$

其中, $\boldsymbol{M}_2=\boldsymbol{D}_2-\boldsymbol{L}_2-\boldsymbol{U}_2,\boldsymbol{D}_2,-\boldsymbol{L}_2,-\boldsymbol{U}_2$ 分别是矩阵 $\boldsymbol{M}_2$ 的对角矩阵、严格下三角矩阵与严格上三角矩阵.

当线性矩阵方程 $\boldsymbol{A}\boldsymbol{X}\boldsymbol{B}=\boldsymbol{C}$ 转化为求解

$$\boldsymbol{A}\boldsymbol{M}_1^{-1}(\boldsymbol{M}_1\boldsymbol{X})\boldsymbol{B}\boldsymbol{M}_2^{-1}=\boldsymbol{C}\boldsymbol{M}_2^{-1}. \tag{3}$$

设 $\boldsymbol{M}_1\boldsymbol{X}=\boldsymbol{Y}$, 则式(3)可转化为求解

$$\tilde{\boldsymbol{A}}\boldsymbol{Y}\tilde{\boldsymbol{B}}=\tilde{\boldsymbol{C}}. \tag{4}$$

分别针对矩阵方程(4)有解与无解两种情况,采用上述变形共轭梯度算法,求 $\boldsymbol{Y}$,然后采用与预处理过程相同的算法计算 $\boldsymbol{M}_1^{-1}\boldsymbol{Y}$ 的近似矩阵 $\boldsymbol{X}$, 便得到矩阵方程的极小范数解或极小范数最小二乘解.

由于这样选取 $\boldsymbol{M}_1,\boldsymbol{M}_2$ 可以保证在求解预处理矩阵方程过程中只需要相邻处理机进行通讯. 在本文中,我们不需要准确计算 $\boldsymbol{A}\boldsymbol{M}_1^{-1},\boldsymbol{B}\boldsymbol{M}_2^{-1}$,而是利用上面的迭代算法迭代 l 次,这样既降低了矩阵的条件数,又减少了计算量.

3　算法的并行实现

为叙述方便,设处理机台数为 P,P 整除 m 且 $m=Pv$,P 整除 n 且 $n=Ps$,P 整除 q 且 $q=Pd$, P 整除 l 且 $l=Pg$,$P_i(i=1,2,\cdots,P)$ 表示第 i 台处理机,myid 表示当前处理机. 记

$$\boldsymbol{A}=(\boldsymbol{A}_1,\boldsymbol{A}_2,\cdots,\boldsymbol{A}_P),\ \tilde{\boldsymbol{A}}=(\tilde{\boldsymbol{A}}_1,\tilde{\boldsymbol{A}}_2,\cdots,\tilde{\boldsymbol{A}}_P),\ \tilde{\boldsymbol{A}}^{\mathrm{T}}=(\tilde{\boldsymbol{A}}_1^{\mathrm{T}},\tilde{\boldsymbol{A}}_2^{\mathrm{T}},\cdots,\tilde{\boldsymbol{A}}_P^{\mathrm{T}})^{\mathrm{T}},$$

$$\boldsymbol{B}=(\boldsymbol{B}_1,\boldsymbol{B}_2,\cdots,\boldsymbol{B}_P),\ \tilde{\boldsymbol{B}}=(\tilde{\boldsymbol{B}}_1,\tilde{\boldsymbol{B}}_2,\cdots,\tilde{\boldsymbol{B}}_P),\ \tilde{\boldsymbol{B}}^{\mathrm{T}}=(\tilde{\boldsymbol{B}}_1^{\mathrm{T}},\tilde{\boldsymbol{B}}_2^{\mathrm{T}},\cdots,\tilde{\boldsymbol{B}}_P^{\mathrm{T}})^{\mathrm{T}},$$

$$\boldsymbol{C}=(\boldsymbol{C}_1,\boldsymbol{C}_2,\cdots,\boldsymbol{C}_P),\ \tilde{\boldsymbol{C}}=(\tilde{\boldsymbol{C}}_1,\tilde{\boldsymbol{C}}_2,\cdots,\tilde{\boldsymbol{C}}_P),\ \boldsymbol{Y}^{(k)}=(\boldsymbol{Y}_1^{(k)},\boldsymbol{Y}_2^{(k)},\cdots,\boldsymbol{Y}_P^{(k)}),$$

$$\boldsymbol{X}^{(k)}=(\boldsymbol{X}_1^{(k)},\boldsymbol{X}_2^{(k)},\cdots,\boldsymbol{X}_P^{(k)}),\ \boldsymbol{R}^{(k)}=(\boldsymbol{R}_1^{(k)},\boldsymbol{R}_2^{(k)},\cdots,\boldsymbol{R}_P^{(k)}),$$

$$\boldsymbol{Z}^{(k)}=(\boldsymbol{Z}_1^{(k)},\boldsymbol{Z}_2^{(k)},\cdots,\boldsymbol{Z}_P^{(k)}),$$

其中，$\boldsymbol{A}_i,\boldsymbol{B}_i$ 分别是 $v\times n,d\times l$ 行块子矩阵，$\tilde{\boldsymbol{A}}_i^{\mathrm{T}},\tilde{\boldsymbol{B}}_i^{\mathrm{T}}$ 是 $n\times v,l\times d$ 列块子矩阵，$\boldsymbol{X}_i^{(k)}$ 是 $n\times d$ 列块子矩阵. 由于矩阵的存储比较复杂，所以在本文中采用了 4 种矩阵相乘的并行算法，分别是[8-9]

方法 1　行列划分矩阵相乘；

方法 2　列行划分矩阵相乘；

方法 3　列列划分矩阵相乘；

方法 4　行行划分矩阵相乘.

1）存储方式. 将 $\boldsymbol{A}_i,\tilde{\boldsymbol{A}}_i^{\mathrm{T}},\boldsymbol{X}_i^{(k)},\boldsymbol{B}_i,\tilde{\boldsymbol{B}}_i^{\mathrm{T}},\boldsymbol{R}_i^{(k)},\boldsymbol{Z}_i^{(k)}$ 均存储于第 $P_i(i=1,2,\cdots,P)$ 台处理机中.

2）预处理过程. 任选初始矩阵 $\tilde{\boldsymbol{A}}^{(0)}$，

for　$h=0,\cdots,l-1$

$$\tilde{\boldsymbol{A}}^{(h+1)}=(\tilde{\boldsymbol{A}}^{(h)}(\boldsymbol{L}_1+\boldsymbol{U}_1)+\boldsymbol{A})\boldsymbol{D}_1^{-1}$$

end

$$\tilde{\boldsymbol{A}}=\tilde{\boldsymbol{A}}^{(l)}$$

其中用方法 1 计算 $\tilde{\boldsymbol{A}}^{(h)}(\boldsymbol{L}_1+\boldsymbol{U}_1)$，再计算 $\tilde{\boldsymbol{A}}^{(h)}(\boldsymbol{L}_1+\boldsymbol{U}_1)+\boldsymbol{A}$，最后用方法 1 计算 $\tilde{\boldsymbol{A}}^{(h+1)}=(\tilde{\boldsymbol{A}}^{(h)}(\boldsymbol{L}_1+\boldsymbol{U}_1)+\boldsymbol{A})\boldsymbol{D}_1^{-1}$.

$$\tilde{\boldsymbol{B}}^{(h+1)}=(\tilde{\boldsymbol{B}}^{(h)}(\boldsymbol{L}_2+\boldsymbol{U}_2)+\boldsymbol{B})\boldsymbol{D}_2^{-1},\ \tilde{\boldsymbol{C}}^{(h+1)}=(\tilde{\boldsymbol{C}}^{(h)}(\boldsymbol{L}_2+\boldsymbol{U}_2)+\boldsymbol{C})\boldsymbol{D}_2^{-1}$$

的计算与 $\tilde{\boldsymbol{A}}^{(h+1)}$ 的计算类似.

矩阵方程(4)有解时算法如下：

用方法 1 计算矩阵乘积 $\boldsymbol{D}=\tilde{\boldsymbol{A}}\boldsymbol{Y}^{(0)}$，方法 4 计算矩阵乘积 $\boldsymbol{D}\tilde{\boldsymbol{B}}$；然后计算 $\boldsymbol{R}_i^{(0)}=\boldsymbol{C}_i-\boldsymbol{D}_i\tilde{\boldsymbol{B}}$. 用方法 2 和方法 3 计算 $\boldsymbol{Q}^{(0)}=\tilde{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{R}^{(0)}\tilde{\boldsymbol{B}}^{\mathrm{T}}$. 置 $\boldsymbol{Z}^{(0)}=\boldsymbol{Q}^{(0)}$.

3）循环过程.

(i) 各进程计算 $[\boldsymbol{R}_i^{(k)},\boldsymbol{R}_i^{(k)}]$ 与 $[\boldsymbol{Z}_i^{(k)},\boldsymbol{Z}_i^{(k)}]$，全规约后得到 $[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]$ 与 $[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]$. 若 $\|\boldsymbol{R}^{(k)}\|<\varepsilon$，$\boldsymbol{Y}=\boldsymbol{Y}^{(k)}$ 转到 4)；否则，继续计算；

(ii) 各进程计算 $\alpha_k=[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]/[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]$；

(iii) 各进程计算 $\boldsymbol{Y}_i^{(k+1)}=\boldsymbol{Y}_i^{(k)}+\alpha_k\boldsymbol{Z}_i^{(k)}$；

(iv) 方法 1 计算矩阵乘积 $\boldsymbol{D}=\tilde{\boldsymbol{A}}\boldsymbol{Y}^{(k+1)}$，方法 4 计算矩阵乘积 $\boldsymbol{D}\tilde{\boldsymbol{B}}$；然后各进程计算 $\boldsymbol{R}_i^{(k+1)}=\boldsymbol{C}_i-\boldsymbol{D}_i\tilde{\boldsymbol{B}}$；各进程计算 $[\boldsymbol{R}_i^{(k+1)},\boldsymbol{R}_i^{(k+1)}]$，全规约后得到 $[\boldsymbol{R}^{(k+1)},\boldsymbol{R}^{(k+1)}]$；

(v) 用方法 2 和方法 3 计算 $\boldsymbol{Q}^{(k+1)}=\tilde{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{R}^{(k+1)}\tilde{\boldsymbol{B}}^{\mathrm{T}}$，各进程计算 $\beta_k=[\boldsymbol{R}^{(k+1)},\boldsymbol{R}^{(k+1)}]/[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]$，$\boldsymbol{Z}_i^{(k+1)}=\boldsymbol{Q}^{(k+1)}+\beta_k\boldsymbol{Z}_i^{(k)}$；

(vi) 置 $k:=k+1$，返回到(i).

4）求解 $\boldsymbol{X}$ 过程.

选初始矩阵 $\boldsymbol{X}^{(0)}=\boldsymbol{O}$，迭代计算

for　$i=0$ to $l-1$

$$\boldsymbol{X}^{(i+1)}=\boldsymbol{D}_1^{-1}((\boldsymbol{L}_1+\boldsymbol{U}_1)\boldsymbol{X}^{(i)}+\boldsymbol{Y}^{(k)})$$

end

$$\boldsymbol{X}=\boldsymbol{X}^{(l)}$$

其中用方法3计算 $\boldsymbol{T}=(\boldsymbol{L}_1+\boldsymbol{U}_1)\boldsymbol{X}^{(i)}$,各进程计算 $\boldsymbol{W}_i=\boldsymbol{T}_i+\boldsymbol{Y}_i$,再用方法3计算 $\boldsymbol{X}^{(i+1)}=\boldsymbol{D}_1^{-1}\boldsymbol{W}$.

矩阵方程(4)无解时算法如下:

用方法1和方法4计算矩阵乘积 $\boldsymbol{D}=\tilde{\boldsymbol{A}}\boldsymbol{Y}^{(0)}\tilde{\boldsymbol{B}}$,然后计算 $\boldsymbol{L}_i=\tilde{\boldsymbol{C}}_i-\boldsymbol{D}_i$;方法2和方法1计算 $\boldsymbol{R}_i^{(0)}=\tilde{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{L}\tilde{\boldsymbol{B}}^{\mathrm{T}}$.方法2计算 $\tilde{\boldsymbol{A}}^{(0)}=\tilde{\boldsymbol{A}}^{\mathrm{T}}\tilde{\boldsymbol{A}}$,方法1计算 $\tilde{\boldsymbol{B}}^{(0)}=\tilde{\boldsymbol{B}}\tilde{\boldsymbol{B}}^{\mathrm{T}}$;方法2计算 $\boldsymbol{Q}^{(0)}=\tilde{\boldsymbol{A}}^{(0)}\boldsymbol{R}^{(0)}\tilde{\boldsymbol{B}}^{(0)}$.置 $\boldsymbol{Z}^{(0)}=\boldsymbol{Q}^{(0)}$.

5)循环过程.

(i)各进程计算 $[\boldsymbol{R}_i^{(k)},\boldsymbol{R}_i^{(k)}]$ 与 $[\boldsymbol{Z}_i^{(k)},\boldsymbol{Z}_i^{(k)}]$,全规约后得到 $[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]$ 与 $[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]$.若 $\|\boldsymbol{R}^{(k)}\|<\varepsilon$,$\boldsymbol{Y}=\boldsymbol{Y}^{(k)}$ 转到4);否则,继续计算;

(ii)各进程计算 $\alpha_k=[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]/[\boldsymbol{Z}^{(k)},\boldsymbol{Z}^{(k)}]$;

(iii)各进程计算 $\boldsymbol{Y}_i^{(k+1)}=\boldsymbol{Y}_i^{(k)}+\alpha_k\boldsymbol{Z}_i^{(k)}$;

(iv)用方法1方法4计算矩阵乘积 $\boldsymbol{D}=\tilde{\boldsymbol{A}}\boldsymbol{Y}^{(k+1)}\tilde{\boldsymbol{B}}$,然后计算 $\boldsymbol{L}_i=\boldsymbol{C}_i-\boldsymbol{D}_i$;方法2和方法1计算 $\boldsymbol{R}_i^{(k+1)}=\tilde{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{L}\tilde{\boldsymbol{B}}^{\mathrm{T}}$.各进程计算 $[\boldsymbol{R}_i^{(k+1)},\boldsymbol{R}_i^{(k+1)}]$,全规约后得到 $[\boldsymbol{R}^{(k+1)},\boldsymbol{R}^{(k+1)}]$;

(v)用方法2方法3计算 $\boldsymbol{Q}^{(k+1)}=\tilde{\boldsymbol{A}}^{(0)}\boldsymbol{R}^{(k+1)}\tilde{\boldsymbol{B}}^{(0)}$,各进程计算 $\beta_k=[\boldsymbol{R}^{(k+1)},\boldsymbol{R}^{(k+1)}]/[\boldsymbol{R}^{(k)},\boldsymbol{R}^{(k)}]$,$\boldsymbol{Z}_i^{(k+1)}=\boldsymbol{Q}^{(k+1)}+\beta_k\boldsymbol{Z}_i^{(k)}$;

(vi)置 $k:=k+1$,返回到(i).

6)求解 $\boldsymbol{X}$ 过程.

选初始矩阵 $\boldsymbol{X}^{(0)}=\boldsymbol{O}$,迭代计算

for $i=0$ to $l-1$

$$\boldsymbol{X}^{(i+1)}=\boldsymbol{D}_1^{-1}((\boldsymbol{L}_1+\boldsymbol{U}_1)\boldsymbol{X}^{(i)}+\boldsymbol{Y}^{(k)})$$

end

$$\boldsymbol{X}=\boldsymbol{X}^{(l)}$$

其中用方法3计算 $\boldsymbol{T}=(\boldsymbol{L}_1+\boldsymbol{U}_1)\boldsymbol{X}^{(i)}$,各进程计算 $\boldsymbol{W}_i=\boldsymbol{T}_i+\boldsymbol{Y}_i$,再用方法3计算 $\boldsymbol{X}^{(i+1)}=\boldsymbol{D}_1^{-1}\boldsymbol{W}$.

4 数值算例与结果分析

在并行机上分别采用MCG算法与本文给出的PMCG算法计算了2个算例进行了比较;在迭代过程中,初始矩阵 $\boldsymbol{Y}^{(0)}$,$\boldsymbol{X}^{(0)}$ 为零矩阵,终止条件 $\varepsilon=10^{-10}$.

例1 求矩阵方程

$$\boldsymbol{AXB}=\boldsymbol{C},\ \boldsymbol{A}=(a_{ij})_{m\times n},\ \boldsymbol{B}=(b_{ij})_{q\times l},\ \boldsymbol{C}=(c_{ij})_{m\times l}.$$

其中

$$\boldsymbol{A}=\begin{bmatrix}4&1&&&\\1&4&1&&\\&\ddots&\ddots&\ddots&\\&&1&4&1\\&&&1&4\end{bmatrix},\ \boldsymbol{B}=\begin{bmatrix}4&1&&&\\1&4&1&&\\&\ddots&\ddots&\ddots&\\&&1&4&1\\&&&1&4\end{bmatrix},\ \boldsymbol{C}=\begin{bmatrix}1&&&&\\&1&&&\\&&\ddots&&\\&&&1&\\&&&&1\end{bmatrix}.$$

求 $\boldsymbol{X}$.这里取 $m=2\,000$,$n=2\,000$,$q=2\,000$,$l=2\,000$.由于矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 为对称正定矩阵,所以可采用共轭梯度法求解,故这里另外给出用共轭梯度法所得的求解结果进行比较,结果见

表 1.

表 1 例 1 的计算结果

Table 1 Numerical results of example 1

	P	10	12	16	20
conjugate gradient method	T/s	1 000.806 4	844.478 7	642.198 7	520.808 4
	K	40	40	40	40
	S		11.851 2	15.584 1	19.216 4
	E		0.99	0.97	0.96
	$\bar{E}$	0.43	0.42	0.41	0.41
	Δ	6.13E-11	6.13E-11	6.13E-11	6.13E-11
deformation conjugate gradient method	T/s	2 072.825 0	2 097.461 9	1 310.749 2	1 103.048 0
	K	127	127	127	127
	S		9.882 5	15.814 0	18.791 8
	E		0.82	0.98	0.94
	$\bar{E}$	0.21	0.17	0.20	0.19
	Δ	8.88E-11	8.88E-11	8.88E-11	8.88E-11
algorithm of this paper $l = 1$	T/s	1 875.597 1	1 932.601 4	1 263.948 4	1 031.566 3
	K	118	118	118	118
	S		9.705 0	14.839 2	18.181 7
	E		0.81	0.93	0.91
	$\bar{E}$	0.22	0.19	0.20	0.20
	Δ	8.00E-11	8.00E-11	8.00E-11	8.00E-11
algorithm of this paper $l = 2$	T/s	428.515 0	437.728 2	289.476 0	224.419 5
	K	21	21	21	21
	S		9.789 5	14.803 1	19.094 4
	E		0.82	0.93	0.95
	$\bar{E}$	1	0.82	0.93	0.95
	Δ	4.26E-11	4.26E-11	4.26E-11	4.26E-11
algorithm of this paper $l = 3$	T/s	438.780 4	453.500 1	290.625 0	232.822 6
	K	19	19	19	19
	S		9.675 4	15.097 8	18.846 1
	E		8.10	0.94	0.94
	$\bar{E}$	0.97	0.79	0.91	0.91
	Δ	4.56E-11	4.56E-11	4.56E-11	4.56E-11

例 2 求矩阵方程

$$\boldsymbol{AXB} = \boldsymbol{C},\ \boldsymbol{A} = (a_{ij})_{m\times n},\ \boldsymbol{B} = (b_{ij})_{q\times l},\ \boldsymbol{C} = (c_{ij})_{m\times l}.$$

其中,(c_{ij}) 是单位矩阵,a_{ij}, b_{ij} 分别是

$$a_{ij} = \begin{cases} 12, & i = j, \\ -3, & i - j = 1, \\ -3, & j - i = 1, \\ 2, & i - j = 0.5n, \\ 2, & j - i = 0.5n. \end{cases}$$

$$b_{ij}=\begin{cases}12, & i=j,\\ -3, & i-j=1,\\ -3, & j-i=1,\\ 2, & i-j=0.5n,\\ 2, & j-i=0.5n.\end{cases}$$

求 $\boldsymbol{X}$. 这里取 $m=2\,000$, $n=2\,000$, $q=2\,000$, $l=2\,000$. 与例 1 类似,另外给出了用共轭梯度法计算的结果,结果见表 2.

表 2　例 2 的计算结果

Table 2　Numerical results of example 2

	P	10	12	16	20
conjugate gradient method	T/s	1 609.863 1	1 360.242 5	1 023.925 4	830.637 1
	K	67	67	67	67
	S		11.835 1	15.722 5	19.381 1
	E		0.99	0.98	0.97
	$\bar{E}$	0.51	0.51	0.50	0.49
	Δ	9.95E−11	9.95E−11	9.95E−11	9.95E−11
deformation conjugate gradient method	T/s	6 006.577 6	5 229.797 9	4 531.145 7	4 528.352 3
	K	345	345	345	345
	S		11.485 3	13.256 2	13.264 4
	E		0.96	0.83	0.66
	$\bar{E}$	0.14	0.13	0.12	0.09
	Δ	9.76E−11	9.76E−11	9.76E−11	9.76E−11
algorithm of this paper $l=1$	T/s	5 908.176 1	4 956.525 5	3 715.029 2	3 096.320 9
	K	314	314	314	314
	S		11.920 0	15.903 4	19.081 3
	E		0.99	0.99	0.95
	$\bar{E}$	0.14	0.14	0.14	0.13
	Δ	9.75E−11	9.75E−11	9.75E−11	9.75E−11
algorithm of this paper $l=2$	T/s	824.440 5	689.632 4	528.293 8	424.928
	K	39	39	39	39
	S		11.954 8	15.605 7	19.401 9
	E		0.99	0.98	0.97
	$\bar{E}$	1	0.99	0.98	0.97
	Δ	8.62E−11	8.62E−11	8.62E−11	8.62E−11
algorithm of this paper $l=3$	T/s	900.259 8	755.605 1	585.873 5	468.998 6
	K	41	41	41	41
	S		11.914 4	15.366 1	19.195 4
	E		0.99	0.96	0.96
	$\bar{E}$	0.92	0.91	0.88	0.88
	Δ	9.39E−11	9.39E−11	9.39E−11	9.39E−11

例 3　求矩阵方程

$$\boldsymbol{AXB}=\boldsymbol{C},\ \boldsymbol{A}=(a_{ij})_{m\times n},\ \boldsymbol{B}=(b_{ij})_{q\times l},\ \boldsymbol{C}=(c_{ij})_{m\times l}.$$

其中

$$a_{ij}=\begin{cases}(i+1)\times n, & i=j,\\ 1, & i\neq j.\end{cases}$$

$$b_{ij}=\begin{cases}(i+1)\times l, & i=j,\\ 1, & i\neq j.\end{cases}$$

$$c_{ij}=\begin{cases}1, & i=j,\\ 0, & i\neq j.\end{cases}$$

求 $\boldsymbol{X}$. 这里取 $m=1\ 200$, $n=1\ 200$, $q=1\ 200$, $l=1\ 200$. 求解结果见表 3 和表 4.

表 3 例 3 $m=1\ 200$, $n=1\ 200$, $q=1\ 200$, $l=1\ 200$ 的计算结果

Table 3 Numerical results of example 3 about $m=1\ 200$, $n=1\ 200$, $q=1\ 200$, $l=1\ 200$

	P	10	12	16	20
conjugate gradient method	T/s	3 823.503 0	3 335.253 7	2 819.507 9	2 477.774 2
	K	1 000	1 000	1 000	1 000
	S		11.463 9	13.560 9	15.431 2
	E		0.96	0.85	0.77
	$\bar{E}$	0.006	0.006	0.005	0.005
	Δ	2.17E02	2.17E02	2.17E02	2.17E02
algorithm of this paper $l=1$	T/s	29.609 3	25.598 2	20.983 3	18.347 8
	K	5	5	5	5
	S		11.566 9	14.110 8	16.137 8
	E		0.96	0.88	0.81
	$\bar{E}$	0.80	0.77	0.70	0.65
	Δ	3.21E−11	3.21E−11	3.21E−11	3.21E−11
algorithm of this paper $l=2$	T/s	23.709 3	20.544 5	16.349 7	14.179 8
	K	2	2	2	2
	S		11.540 5	14.501 4	16.720 5
	E		0.96	0.91	0.84
	$\bar{E}$	1	0.96	0.91	0.84
	Δ	6.82E−11	6.82E−11	6.82E−11	6.82E−11
algorithm of this paper $l=3$	T/s	29.561 3	25.188 8	20.500 9	20.292 0
	K	2	2	2	2
	S		11.735 9	14.419 5	14.568 0
	E		0.98	0.90	0.73
	$\bar{E}$	0.80	0.78	0.72	0.58
	Δ	2.70E−15	2.70E−15	2.70E−15	2.70E−15

表 4 例 3 $m=1\ 000$, $n=1\ 200$, $q=1\ 000$, $l=1\ 200$ 的计算结果

Table 4 Numerical results of example 3 about $m=1\ 000$, $n=1\ 200$, $q=1\ 000$, $l=1\ 200$

	P	10	12	16	20
conjugate gradient method	T/s	−	−	−	−
	K	−	−	−	−
	S	−	−	−	−
	E	−	−	−	−
	$\bar{E}$	−	−	−	−
	Δ	−	−	−	−

续表 4

	P	10	12	16	20
algorithm of this paper $l=1$	T/s	19.907 7	16.872 2	15.720 0	13.202 0
	K	34	34	37	36
	S		9.439 3	10.131 1	12.063 4
	E		0.94	0.84	0.75
	$\bar{E}$	0.26	0.24	0.22	0.20
	Δ	5.48E-11	5.18E-11	5.18E-11	5.18E-11
algorithm of this paper $l=2$	T/s	6.452 0	5.586 7	4.884 8	4.282 0
	K	7	7	7	7
	S		9.239 1	10.566 7	12.054 2
	E		0.92	0.88	0.75
	$\bar{E}$	0.79	0.73	0.70	0.59
	Δ	4.29E-11	4.29E-11	4.29E-11	4.29E-11
algorithm of this paper $l=3$	T/s	5.125 8	4.514 1	4.044 9	3.441 7
	K	3	3	3	3
	S		9.084 1	10.137 8	11.914 6
	E		0.91	0.84	0.74
	$\bar{E}$	1	0.91	0.84	0.74
	Δ	6.77E-11	6.77E-11	6.77E-11	6.77E-11

注 1　表 1 ~ 4 中，P 表示处理机台数，T 表示时间，K 表示迭代次数，S 表示加速比，E 表示相对并行效率，$\bar{E}$ 表示绝对并行效率，Δ 表示误差.

注 2　由于一台处理机的计算时间较长，因此采用多台计算机的计算时间，并且使用多台处理机中最小的计算时间作为基准来计算加速比和并行效率.

注 3　例 3 中表 3 当迭代次数 $K = 1\ 000$ 时，退出循环. 表 4 中当迭代次数 $K = 3\ 000$ 时，退出循环.

4　结果分析

1）本章算法明显提高了迭代速度，提高了变形共轭梯度法的效率，具有良好的并行性.

2）例 1 和例 2 说明，当矩阵是对称正定时，使用共轭梯度法的迭代次数比变形共轭梯度法的迭代次数少，但本章算法内迭代次数 $l=2$ 时的迭代次数比共轭梯度法少. $l=1$ 的迭代次数没有明显减少，说明原线性矩阵方程的条件数没有明显变化，预处理算法基本失效，这是因为本文算法的预处理矩阵是数量矩阵. 在本章算法中，当内迭代次数 $l=2$ 时效果最好，迭代速度已经达到很快并且并行效率很高，优于 $l=1$ 的情况，即优于传统的 Jacobi 预处理方法.

3）例 3 的表 3 结果可以看出，当迭代次数到达 1 000 时变形共轭梯度法也没有达到收敛精度，而使用本章算法，迭代次数明显减少，大大地提高了收敛速度. 在本章算法中，当内迭代次数 $l=2$ 时效果最好，迭代速度已经达到很快并且并行效率很高.

4）例 3 的表 4 结果可以看出，原矩阵方程无解时，求矩阵方程的极小范数最小二乘解. 当迭代次数到达 3 000 时变形共轭梯度法也没有达到收敛精度，而使用本章算法，当内迭代次数 $l=1$ 时，在有限步后就达到收敛，大大地提高了收敛速度.

5 结束语

本文主要内容是变形共轭梯度法求解一类特殊的矩阵方程 $\boldsymbol{AXB}=\boldsymbol{C}$. 首先通过构造预处理矩阵 $\boldsymbol{M}_1,\boldsymbol{M}_2$ 来降低求解矩阵方程的复杂度. 本文给出了一种预处理方法. 预处理矩阵 $\boldsymbol{M}_1$, $\boldsymbol{M}_2$ 是严格对角占优矩阵,该方法提出了迭代法的预处理模式,构造并行迭代求解预处理矩阵方程的迭代格式,进而使用变形共轭梯度法并行求解. 最后通过本文的 3 个算例结果与直接使用变形共轭梯度法求解矩阵方程相比较,我们得出结论,本文算法不仅有效地提高了并行效率,而且具有良好的并行性.

致谢 感谢西北工业大学高性能计算研究与发展中心的大力支持!

参考文献(References):

[1] Penrose R. A generalized inverse for matrices[J]. *Mathematical Proceedings of the Cambridge Philosophical Society*, 1955, **51**(3): 406-413.

[2] Lancaster P. Explicit solutions of linear matrix equations[J]. *SIAM Review*, 1970, **72**(4): 544-566.

[3] Mitra S K. The matrix equation $\boldsymbol{AX}=\boldsymbol{C},\boldsymbol{XB}=\boldsymbol{D}$ [J]. *Linear Algebre and Its Application*, 1984, **59**: 171-181.

[4] Uhlig F. On the matrix equation $\boldsymbol{AX}=\boldsymbol{B}$ with applications to the generators of controllability matrix[J]. *Linear Algebre and Its Application*, 1987, **85**: 203-209.

[5] XIAO Qing-feng, HU Xi-yan, ZHANG Lei. The symmetric minimal rank solution of the matrix equation $\boldsymbol{AX}=\boldsymbol{B}$ and the optimal approximation[J]. *Electronic Journal of Linear Algebra*, 2009, **18**: 264-273.

[6] 张凯院, 徐仲. 数值代数[M]. 第 2 版修订本. 北京: 科学出版社, 2010. (ZHANG Kai-yuan, XU Zhong. *Numerical Algebra*[M]. Revised 2nd Ed. Beijing: Science Press, 2010. (in Chinese))

[7] 胡家赣. 解线性代数方程组的迭代解法[M]. 北京:科学出版社, 1999: 173-201. (HU Jia-gan. *Iterative Solution of Linear Algebraic Equations*[M]. Beijing: Science Press, 1999: 173-201. (in Chinese))

[8] 陈国良, 安虹, 陈俊, 郑启龙, 单九龙. 并行算法实践[M]. 北京:高等教育出版社, 2004. (CHEN Guo-liang, AN Hong, CHEN Jun, ZHENG Qi-long, SHAN Jiu-long. *The Parallel Algorithm*[M]. Beijing: Higher Education Press, 2004. (in Chinese))

[9] 李晓梅, 吴建平. 数值并行算法与软件[M]. 北京: 科学出版社, 2007. (LI Xiao-mei, WU Jianping. *Numerical Parallel Algorithm and Software*[M]. Beijing: Science Press, 2007. (in Chinese))

A Parallel Preconditioned Modified Conjugate Gradient Method for a Kind of Matrix Equation

CAO Fang-ying[1], LÜ Quan-yi[1], XIE Gong-nan[2]

(1. *Department of Applied Mathematics, Northwestern Polytechnical University, Xi'an* 710129, *P. R. China*;

2. *Engineering Simulation and Aerospace Computing, School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an* 710072, *P. R. China*)

Abstract: In view of a parallel algorithm of preconditioned modified conjugate gradient method for solving a kind of matrix equation $\boldsymbol{AXB} = \boldsymbol{C}$, a preconditioned model was proposed. Based on this thought, firstly the preconditioned matrix was constructed, which was strictly diagonally dominant matrix, secondly the parallel algorithm for preprocessing matrix equation iterative format was formed, and finally the modified conjugate gradient method was used for parallel solving the preconditioned matrix equation. Through numerical experiments, comparing our algorithm with the modified conjugate gradient method, ours has higher parallel efficiency.

Key words: parallel algorithm; preconditioned modified conjugate gradient method; preconditioned matrix; parallelism