Pure Mathematics 理论数学, 2017, 7(2), 89-98
Published Online March 2017 in Hans. http://www.hanspub.org/journal/pm
https://doi.org/10.12677/pm.2017.72013

Hans汉斯

Parallel Methods for Parabolic Equations Based on MPI Implementation

Yuyang Gao, Haiming Gu

Qingdao University of Science and Technology, Qingdao Shandong
Email: crazy3435@163.com, ghm@qust.edu.cn

Received: Mar. 11th, 2017; accepted: Mar. 28th, 2017; published: Mar. 31st, 2017

Abstract

Many applications in mathematics and engineering involve numerical solutions of partial differential equations (PDEs). The demands of large-scale computing are quickly increasing in modern science and technology, and parallel computing has received more and more attention. In this paper, the main idea is that classical Group Explicit method (GEM) for parabolic equations, the group explicit method is established briefly and the stability analysis of the method is indicated simply. Then we focus on how to calculate the format in MPI parallel environment. Two parallel MPI algorithms are established and compared with non-parallel algorithm based on GEM. They are MPI block communication (wait communication) and non-blocking communication (no-wait communication). These two MPI schemas both better than one single process to calculate numerical solutions use group explicit method. Also, the non-blocking communication program has higher computational efficiency than blocking communication program.

Keywords

Finite Difference Method, Group Explicit Method, Parabolic Equations, MPI (Message Passing Interface)

求解抛物方程的MPI并行方法

高玉羊，顾海明

青岛科技大学，山东 青岛
Email: crazy3435@163.com, ghm@qust.edu.cn

收稿日期：2017年3月11日；录用日期：2017年3月28日；发布日期：2017年3月31日

摘 要

数值求解偏微分方程广泛应用于数学与工程领域。大规模数值计算在当今科学技术运用中得到飞速发展，

文章引用: 高玉羊, 顾海明. 求解抛物方程的 MPI 并行方法[J]. 理论数学, 2017, 7(2): 89-98.
https://doi.org/10.12677/pm.2017.72013

其中可并行的有限差分格式受到越来越多的重视。在本文中，主要阐述了经典的分组显示方法求解抛物方程，并简单扼要的分析了该格式的建立以及稳定性。随后本文着重介绍了如何在MPI并行环境下对该格式进行数值计算，构建了两种不同的并行计算模型，即阻塞通信(等待模式)和非阻塞通信(即非等待模式)模式。并与非并行状态下的差分格式做出比较，结果表明，相对于一个进程求解偏微分方程，两种模式都表现出较好的效果，而且非阻塞通信相较于阻塞通信模式亦表现出较好的并行效率。

关键词

有限差分法，分组显式格式，抛物方程，MPI (Message Passing Interface)

1. 引言

现代科学技术的发展很大程度上依赖于物理学、化学、生物学的成就与进展，而这些学科自身的精确化又是他们取得进展的重要保证。学科的精确化往往是通过建立数学模型来实现的，而数学模型的大都可以归纳为偏微分方程的形式。现代科学技术对大规模科学与工程计算的需求日益增长，在科学与工程计算领域提出了许多大规模计算和超大规模计算问题，解决这些问题需要在高性能并行计算机([1] [2])上进行，而并行计算机系统的研制和发展推动了偏微分方程并行算法的研究与发展。

分组显式方法(文献[3])是 Evans 等人上世纪 80 年代设计的可以并行计算的经典有限差分算法。分组显式格式是使用不同类型的 Saul'yev 非对称格式([4])进行恰当的组合，在不同的时间层上连续交替的使用不同的非对称格式，这样，可带来截断误差的明显改善，从而使算法的精度得以提高。基于分组显式的思想，有限并行差分格式得到广泛的发展并应用于求解其他偏微分方程([5] [6] [7])中。

MPI 是目前最重要的并行编程工具([8])，它具有移植性好、功能强大、效率高等多种优点，MPI 其实是一个“库”，共有上百个函数调用接口，在 FORTRAN 和 C 语言中可以直接对这些函数进行调用。MPI 为并行算法提供了多种通信模式，一般的阻塞通讯模式即可满足大多数并行算法，但非阻塞通讯模式由于相对减少了通讯时间，从而使并行模式的效率得到较好的提升。将 MPI 并行技术与有限并行差分格式结合需要构造相匹配的消息传递模型([9])，因此，构造合适的并行消息传递模型是并行数值计算偏微分方程的重点。

2. 有限并行差分格式

2.1. 分组显式方法

在区域 $0 \le x \le 1, 0 < t < T$ 上，考虑如下附有初始条件和边界条件的抛物方程。

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad 0 \le x \le 1, \ t \ge 0. \tag{1}$$

$$u(x,0) = f(x) \tag{2}$$

$$\begin{cases} u(0,t) = g_1(x) \\ u(1,t) = g_2(t) \end{cases} \tag{3}$$

将区域$[0,1]\times[0,T)$进行剖分，空间步长$\Delta x=1/m$，m为正整数，时间步长为Δt，$x_i=i\Delta x\,(i=0,1,\cdots,m)$，$t_j=j\Delta t\,(j=0,1,2,\cdots)$。为简单计，将节点$(x_i,t_j)$记为$(i,j)$。构造分组显式格式需要用到 Saul'yev 非对称格式。首先，可以得到以下很明显的关系式，

$$\frac{\partial U_i^j}{\partial t}=\frac{U_i^{j+1}-U_i^j}{\Delta t}+O(\Delta t) \tag{4}$$

$$\frac{\partial^2 U_i^j}{\partial x^2}=\frac{1}{\Delta x}\left(\frac{\partial U_{i+1/2}^j}{\partial x}-\frac{\partial U_{i-1/2}^j}{\partial x}\right)+O(\Delta x^2) \tag{5}$$

应用如下等式

$$\frac{\partial U_{i-1/2}^j}{\partial x}=\frac{\partial U_{i-1/2}^{j+1}}{\partial x}-\Delta t\frac{\partial^2 U_{i-1/2}^{t_j+\theta\Delta t}}{\partial x\partial t},\ (0\le\theta\le 1)$$

将(4)，(5)代入方程(1)得到

$$\frac{U_i^{j+1}-U_i^j}{\Delta t}=\frac{1}{\Delta x}\left(\frac{\partial U_{i+1/2}^j}{\partial x}-\frac{\partial U_{i-1/2}^{j+1}}{\partial x}\right)+\frac{\Delta t}{\Delta x}\frac{\partial^2 U_{i-1/2}^{t_j+\theta\Delta t}}{\partial x\partial t}+O(\Delta t+\Delta x^2) \tag{7}$$

最后，应用如下显然的表达式

$$\frac{\partial U_{i-1/2}^{j+1}}{\partial x}=\frac{U_i^{j+1}-U_{i-1}^{j+1}}{\Delta x}+O(\Delta x^2) \tag{8}$$

$$\frac{\partial U_{i+1/2}^j}{\partial x}=\frac{U_{i+1}^j-U_i^j}{\Delta x}+O(\Delta x^2) \tag{9}$$

便可以将(7)写成如下形式，

$$\frac{U_i^{j+1}-U_i^j}{\Delta t}=\frac{1}{\Delta x^2}\left(U_{i-1}^{j+1}-U_i^{j+1}-U_i^j+U_{i+1}^j\right)+R_i^j \tag{10}$$

其中，

$$R_i^j=O\left(\frac{\Delta t}{\Delta x}+\Delta x^2+\Delta t\right) \tag{11}$$

令$r=\Delta t/\Delta x^2$，并舍弃无穷小项$\Delta t R_i^j$，可以得到如下网格方程，

$$(1+r)u_i^{j+1}-ru_{i-1}^{j+1}=ru_{i+1}^j+(1-r)u_i^j \tag{12}$$

完全类似的可以推导出公式

$$-ru_{i+1}^{j+1}+(1+r)u_i^{j+1}=ru_{i-1}^j+(1-r)u_i^j \tag{13}$$

将方程(12)、(13)写成矩阵的形式，

$$\begin{bmatrix}1+r & -r\\ -r & 1+r\end{bmatrix}\begin{bmatrix}u_i^{j+1}\\ u_{i+1}^{j+1}\end{bmatrix}=\begin{bmatrix}1-r & 0\\ 0 & 1-r\end{bmatrix}\begin{bmatrix}u_i^j\\ u_{i+1}^j\end{bmatrix}+\begin{bmatrix}ru_{i-1}^j\\ ru_{i+2}^j\end{bmatrix} \tag{14}$$

将矩阵移向右侧，(14)可以写成

$$\begin{bmatrix}u_i^{j+1}\\ u_{i+1}^{j+1}\end{bmatrix}=\frac{1}{|A|}\begin{bmatrix}1+r & r\\ r & 1+r\end{bmatrix}\left\{\begin{bmatrix}1-r & 0\\ 0 & 1-r\end{bmatrix}\begin{bmatrix}u_i^j\\ u_{i+1}^j\end{bmatrix}+\begin{bmatrix}ru_{i-1}^j\\ ru_{i+2}^j\end{bmatrix}\right\} \tag{15}$$

其中$|A|=1+2r$，因此方程(15)可以写成如下矩阵的形式，

$$\begin{bmatrix} u_i^{j+1} \\ u_{i+1}^{j+1} \end{bmatrix} = \frac{1}{|A|} \times \begin{bmatrix} r(1+r)u_{i-1}^j + (1-r^2)u_i^j + r(1-r)u_{i+1}^j + r^2 u_{i+2}^j \\ r^2 u_{i-1}^j + r(1-r)u_i^j + (1-r^2)u_{i+1}^j + r(1+r)u_{i+2}^j \end{bmatrix} \tag{16}$$

在靠近求解区域左边界和右边界的点可以分别使用方程(12)、(13)进行求解。

$$u_1^{j+1} = \frac{1}{1+r}\left(ru_0^{j+1} + ru_2^j + (1-r)u_1^j\right) \tag{17}$$

$$u_{m-1}^{j+1} = \frac{1}{1+r}\left(ru_m^{j+1} + ru_{m-2}^j + (1-r)u_{m-1}^j\right) \tag{18}$$

以上即为分组显式格式的基本思想，为了适配下文中的 MPI 消息传递模型，将会选择适当的内部节点数进行网格剖分。考虑区域$(0,1)$上的划分间隔为偶数，既m为偶数，则在每一时间层上，内部节点数为$(m-1)$，为奇数。这样，就会存在一个单独的点无法使用方程(14)求解。为了更好的处理边界节点的值，在下文中给出了具体的方法。

2.2. 交替分组方法

在每一时间层，除初始条件外，所有内部节点是未知的。内部节点的最后一个点使用格式(18)，其余诸点，从内部节点的第一个点开始到第$(m-1)$个内部节点使用方程(16)，共进行$\frac{1}{2}(m-2)$次计算，将该格式称为右端分组显式(GER)格式。写成矩阵的形式如下，

$$\begin{bmatrix} 1+r & -r & & & & & & & \\ -r & 1+r & & & & & & & \\ & & 1+r & -r & & & & & \\ & & -r & 1+r & & & & & \\ & & & & \ddots & & & & \\ & & & & & 1+r & -r & & \\ & & & & & -r & 1+r & & \\ & & & & & & & 1+r & \end{bmatrix} \begin{bmatrix} u_1^{j+1} \\ u_2^{j+1} \\ u_3^{j+1} \\ \vdots \\ \vdots \\ u_{m-3}^{j+1} \\ u_{m-2}^{j+1} \\ u_{m-1}^{j+1} \end{bmatrix}$$

$$= \begin{bmatrix} 1-r & & & & & & & & \\ & 1-r & r & & & & & & \\ & r & 1-r & & & & & & \\ & & & 1-r & r & & & & \\ & & & r & 1-r & & & & \\ & & & & & \ddots & & & \\ & & & & & & 1-r & r & \\ & & & & & & r & 1-r & \end{bmatrix} \begin{bmatrix} u_1^j \\ u_2^j \\ u_3^j \\ \vdots \\ \vdots \\ u_{m-3}^j \\ u_{m-2}^j \\ u_{m-1}^j \end{bmatrix} + b_1 \tag{19}$$

$$b_1^{\mathrm{T}} = \left[ru_0^j, 0, 0, \cdots, 0, ru_m^{j+1}\right]$$

方程(19)写成矩阵的形式为

$$(I + rG_1)u_{j+1} = (I - rG_2)u_j + b_1 \tag{20}$$

其中

$$G_1=\begin{bmatrix}1&-1&&&&\\-1&1&&&&\\&&\ddots&&&\\&&&1&-1&\\&&&-1&1&\\&&&&&1\end{bmatrix}=\begin{bmatrix}G^{(1)}&&&&&\\&G^{(2)}&&&&\\&&\ddots&&&\\&&&\ddots&&\\&&&&G^{(1/2)(m-2)}&\\&&&&&1\end{bmatrix}$$

$$G_2=\begin{bmatrix}1&&&&&\\&1&-1&&&\\&-1&1&&&\\&&&\ddots&&\\&&&&1&-1\\&&&&-1&1\end{bmatrix}=\begin{bmatrix}1&&&&&\\&G^{(1)}&&&&\\&&G^{(2)}&&&\\&&&\ddots&&\\&&&&\ddots&\\&&&&&G^{\frac{1}{2}(m-2)}\end{bmatrix}$$

$$G^{(i)}=\begin{bmatrix}1&-1\\-1&1\end{bmatrix},\quad i=1,2,\cdots,\frac{1}{2}(m-2),\quad j=0,1,\cdots.$$

$$u_j=\left[u_1^j,u_2^j,u_3^j,\cdots,u_{m-1}^j\right]$$

运用相似的方法，在每一时间层，内部节点的第一个节点使用方程(18)，其余内部节点使用方程(16)，可以得到左端分组显式(GEL)格式。

$$\left(I+rG_2\right)u_{j+1}=\left(I-rG_1\right)u_j+b_2 \tag{21}$$

$$b_2^{\mathrm{T}}=\left[ru_0^{j+1},0,0,\cdots,0,ru_m^j\right]$$

在相邻的时间层上交替使用方程(20)、(21)，即可得到交替分组方法(AGE)。其矩阵形式如下，

$$\begin{cases}\left(I+rG_1\right)u_{j+1}=\left(I-rG_2\right)u_j+b_1\\\left(I+rG_2\right)u_{j+2}=\left(I-rG_1\right)u_{j+1}+b_2\end{cases} \tag{22}$$

方程(16)的稳定分析在文献[3]中已经得到充分的说明，稳定性条件为$r\le 1$时误差的增长得到有效的控制，同样，使用文献[10]中的 Kellogg 引理，可以得到方程(20)，(21)，(22)的稳定性条件为$r\le 1$。

3. MPI 并行算法

在本文中，所有并行算法都是针对上文中给出的有限差分格式进行并行化计算。将求解区域$[0,1]$划分为若干大致相等的区域，比如说，若将方程(1)进行 4 进程并行计算，就将求解区域化为 4 条带状的区域，每个进程负责一条带状计算区域，在时间层上逐次计算，为了实现高效计算，比较适宜的做法是为每一条进程分配大约相等的任务量，即保证在空间方向上分割的带状区域大小几乎是一致的。每个相邻进程进行数据通信，当通信与计算完毕，继续进行下一时间层的计算。下文中，将会给出两种不同的通讯模式，即阻塞通信与非阻塞通信，对交替分组格式进行并行计算。表 1 和表 2 分别展示了 MPI 阻塞通信与非阻塞通信的步骤流。

考虑划分区域内部间隔为偶数m，内部节点数$m-1$(同上文，划分区域第一个节点标号为 0，最后一个节点标号为m)，内点数为奇数。不失一般性，考虑以上差分格式在 4 条进程下完成并行计算，不妨假设已知时间层为n上的数据，计算时间层为$n+1$上的值时，将内部节点大约的分为 4 个部分，由于节点

Table 1. The steps of blocking communication mode
表 1. 阻塞通信模式下并行详细步骤

	进程 0	其它进程
步骤 1	计算本进程的所负责的计算区间	同 0 进程
步骤 2	读取本进程的计算数据	同 0 进程
步骤 3	发送数据到其他进程或从其他进程接收数据	同 0 进程
步骤 4	将数据带入差分格式并进行计算	同 0 进程
步骤 5	进入下一时间层的计算，转到步骤 3	同 0 进程

Table 2. The steps of non-blocking communication mode
表 2. 非阻塞通信模式下并行详细步骤

	进程 0	其他进程
步骤 1	计算本进程的所负责的计算区间，并读取本进程的计算数据	同 0 进程
步骤 2	发送数据到其他进程或从其他进程接收数据	同 0 进程
步骤 3	非通信区域的数据计算	同 0 进程
步骤 4	等待通信完成，计算通信区域的数据	同 0 进程
步骤 5	进入下一时间层的计算，转到步骤 2	同 0 进程

数是 $m-1$ 为奇数，则必然有一个点无法使用方程(16)分组计算。在使用 GEL 格式计算的情况下，在 0 号进程中，划分区域内部节点的第 1 个点由方程(17)计算，进程 0 的其他节点和其他进程的点使用方程(16)计算。由于与 0 号进程相邻的 1 号进程之间需要进行数据通信，既计算第 $n+1$ 层中 0 号进程的最右端的两个点需要第 n 层中的 1 号进程的第一个点的数据，计算第 $n+1$ 层中 1 号进程最左端两个点需要第 n 层中 0 号进程最右端一个点的数据，其他进程如 1 号进程与 2 号进程、2 号进程与 3 号进程之间的通讯也是如此。在下文中，以 0，1 号进程之间的通讯传递为例，给出 MPI 通信模型，GEL 通讯简图见图 1。

在使用 GER 格式计算的情况下，与 GEL 格式类似，单独的内部节点的计算由最后一个进程执行，既 $n+1$ 时间层中 3 号进程的最右端内部节点由方程(18)计算，其他节点两两成组由方程(16)计算，与 GEL 格式一致，相邻进程的节点进行数据通信，GER 通讯简图见图 2。

在使用交替分组格式计算的情况下，与 GER，GEL 格式有些不同之处，首先考虑在第 n 层上使用 GEL 格式，在第 $n+1$ 层上使用 GER 格式，第 n 层上的通讯与计算与普通的 GEL 格式一样，在进行 $n+1$ 层上的计算时，相邻两个进程的最后两个数据与最初的两个数据分别向后一个进程和前一个进程发送。通讯简图如图 3。

在 MPI 并行环境下，阻塞通信模式下用到的计算与通讯过程中的命令，如下，

MPI_Send(void* buf, int count, MPI_Datatype, int destination, int tag, MPI_Comm comm)

MPI_Recv(void* buf, int count, MPI_Datatype, int source, int tag, MPI_Comm comm, MPI_Status*status)

非阻塞通信模式下的通信和计算与阻塞通信模式下的基本一致，差异之处在于各个进程之间的通讯可能尚未结束时，就已经在进行数据的计算，这样往往可以使 CPU 在各个核心在通讯的过程中，仍然满载运行，从而减少程序运行的时间，达到并行计算效率的提升。

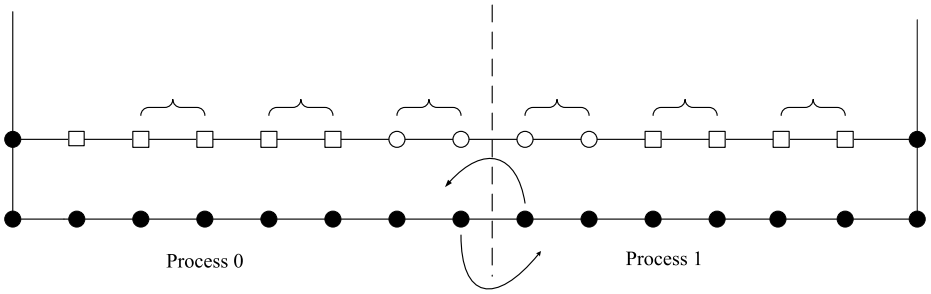

Figure 1. Communication mode of GEL
图 1. GEL 格式通讯简图

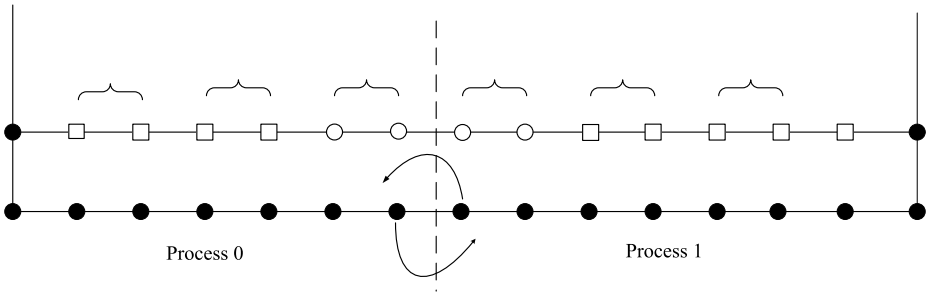

Figure 2. Communication mode of GER
图 2. GER 格式通讯简图

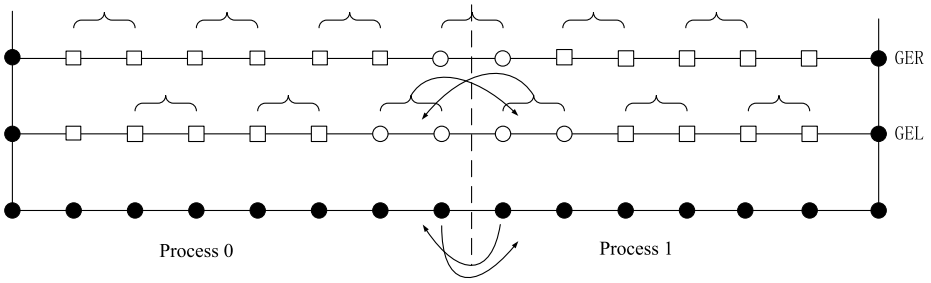

Figure 3. Communication mode of AGE
图 3. AGE 格式通讯简图

在 MPI 并行环境下，非阻塞通信模式下用到的计算与通讯过程中的命令，如下

MPI_Isend(void* buf, int count, MPI_Datatype, int destination, int tag, MPI_Comm comm, MPI_Request*request)

MPI_Irecv(void* buf, int count, MPI_Datatype, int source, int tag, MPI_Comm comm, MPI_Request*request)

MPI_Waitall(int count, MPI_Request*request, MPI_Status*status)

4. 数值并行计算

为了验证 MPI 并行环境下交替分组格式的并行效率的提升情况。作如下数值运算。对方程(1)，考虑如下初始条件和边界条件，

$$u(x,0)=4x(1-x),\quad 0\le x\le 1$$

$$u(0,t)=u(1,t)=0,\quad t\ge 0$$

下文给出的数值模拟计算的结果是在双核计算机上进行 4 进程并行计算得到的运算结果。在上述三种并行有限差分格式下，其中，$r=0.1$，$n=3000$，$\Delta t=1\times10^{-11}$。由于 MPI 并行消息传递模型并未改变交替分组差分格式的逻辑结构，此处不再赘述方程(1)的数值解的误差分析，而且在文献[3]中，亦给出了该类型抛物方程的误差分析状况，在下文中，针对不同的通信模式，得到了两种 MPI 并行模型在不同进程下的加速比及计算效率。表 3~5 分别展示了 GEL，GER，AGE 格式下的 MPI 阻塞与非阻塞通信模型的程序运算时间，图 4 显示了运用交替分组格式求解抛物方程，在非并行环境与 MPI 并行环境下的程序运算时间柱状图。图 5 给出了 4 进程并行计算下阻塞通信与非阻塞通信的时间柱状图。表 6 给出了 4 进程通信下的并行效率及加速比。

Table 3. The executive time of 4 processes in blocking and non-blocking communication mode for GEL (unit/sec.)
表 3. GEL 格式下 4 进程阻塞通信与非阻塞通信模式的程序运算时间比较(单位/秒)

	进程 0	进程 1	进程 2	进程 3	单个进程
GEL 格式阻塞通信计算时间	5.504456	5.541956	5.534419	5.27101	
GEL 格式阻塞通信总时间	6.941848	6.941303	6.941578	6.941439	
GEL 单进程运算执行时间					25.39734
GEL 格式非阻塞通信计算时间	5.189739	5.414192	5.513848	5.287586	
GEL 格式非阻塞通信总时间	6.307096	6.308636	6.309482	6.308666	
GEL 格式非阻塞通信等待时间	0.580585	0.201888	0.063008	0.493103	

Table 4. The executive time of 4 processes in blocking and non-blocking communication mode for GER (unit/sec.)
表 4. GER 格式下 4 进程阻塞通信与非阻塞通信模式的程序运算时间比较(单位/秒)

	进程 0	进程 1	进程 2	进程 3	单个进程
GER 格式阻塞通信计算时间	5.56917	5.632037	5.605388	5.385093	
GER 格式阻塞通信总时间	6.804605	6.805075	6.805131	6.805165	
GER 单进程运算执行时间					25.372994
GER 格式非阻塞通信计算时间	5.214437	5.44246	5.518787	5.217341	
GER 格式非阻塞通信总时间	6.332524	6.333553	6.335751	6.332815	
GER 格式非阻塞通信等待时间	0.576401	0.207211	0.089865	0.577639	

Table 5. The executive time of 4 processes in blocking and non-blocking communication mode for AGE (unit/sec.)
表 5. 交替分组格式下 4 进程阻塞通信与非阻塞通信模式的程序运算时间比较(单位/秒)

	进程 0	进程 1	进程 2	进程 3	单个进程
AGE 格式阻塞通信计算时间	5.58503	5.613905	5.533567	5.387905	
AGE 格式阻塞通信总时间	6.708155	6.708058	6.707898	6.70738	
AGE 单进程运算执行时间					25.099077
AGE 格式非阻塞通信计算时间	5.449957	5.648745	5.650483	5.468564	
AGE 格式非阻塞通信总时间	6.367587	6.368233	6.367122	6.364805	
AGE 格式非阻塞通信等待时间	0.559427	0.155265	0.188809	0.513443	

Table 6. Comparison of the parallel efficiency in 4 processes
表 6. 不同格式下的 4 进程并行效率比较

	总时间(单位/秒)	加速比	并行效率
非并行时	25.099	-----	-----
GEL 格式 4 进程阻塞通信	6.9418	3.62	0.905
GEL 格式 4 进程非阻塞通信	6.3070	3.97	0.993
GER 格式 4 进程阻塞通信	6.8052	3.68	0.92
GER 格式 4 进程非阻塞通信	6.3357	3.96	0.99
AGE 格式 4 进程阻塞通信	6.7081	3.74	0.93
AGE 格式 4 进程非阻塞通信	6.3675	3.94	0.985

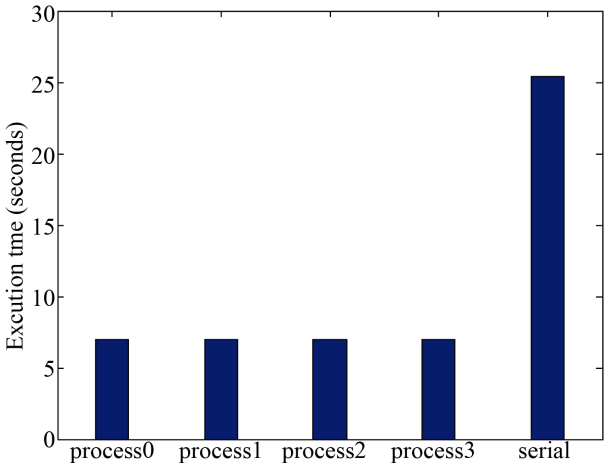

Figure 4. The executive time of single process and 4 processes of AGE blocking communication program
图 4. AGE 阻塞通信模式下四进程与单进程计算的程序运行时间

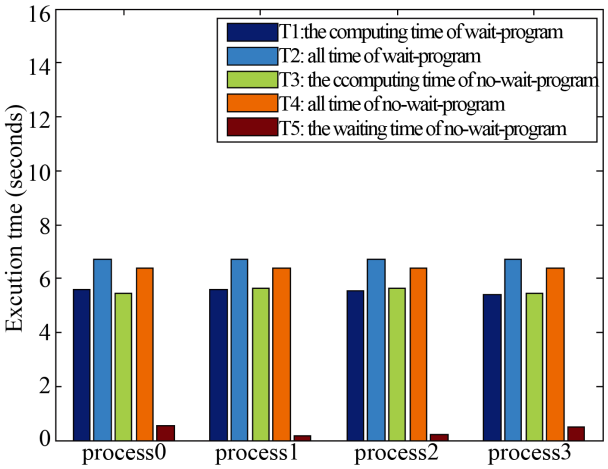

Figure 5. The executive time of AGE blocking and non-blocking communication program in 4 processes
图 5. 四进程下 AGE 阻塞通信模式与非阻塞通信模式下的程序运行时间

5. 总结

在上述数值模拟计算的数据表格中，在图 4 中可以看到，在 MPI 并行环境中，多条进程运用上述差分格式数值计算抛物方程要比单个进程计算的效率更高。多次试验数据表明，4 进程并行计算的时间约是单进程计算的 1/4，而且三种差分格式并未出现较大的差别，因为可以在上文中知道，三种差分格式的计算任务量是大致相等的，而 MPI 并行化处理并未改变计算逻辑，而只是将计算任务分摊，从而达到计算效率的提升。在表 3~5 与图 5 中，可以看到 MPI 阻塞通信对于非阻塞通信的优势，其中，对某一进程来说，非阻塞与阻塞通信的程序计算时间是大致相等的，而非阻塞通信程序的总运算时间是相对较小的，因为非阻塞通信减少了各进程之间的消息传递时间，即通信等待时间，在表 3~5 中并未给出阻塞通信的通信等待时间，是因为在阻塞通信模型中该时间是无法显示的，不过在实际运算中，阻塞通信的通信等待时间是要高于非阻塞通信的。因此非阻塞通信相对于阻塞通信来说表现出更大的优势。但是，非阻塞通信模型的建立要比阻塞通信更加复杂，这在程序的编码实现上有较多的问题需要解决。在表 6 还可以看到不同差分格式下的 4 进程并行效率的比较，可以进一步表明非阻塞通信相对于阻塞通信的优势。即在相同的进程数目下，无论是加速比还是并行效率，非阻塞通信都要高于阻塞通信。

参考文献 (References)

[1] 都志辉, 李三立, 陈渔, 刘鹏. 高性能计算之并行编程技术——MPI 并行程序设计[M]. 北京: 清华大学出版社, 2001.

[2] Quinn, M.J. (2004) Parallel Programming in C with MPI and Open MP. McGraw-Hill, New York.

[3] Evans, D.J. and Abdullah, A.R. (1983) Group Explicit Methods for Parabolic Equations. *International Journal Computer Mathematics*, **14**, 73-105. https://doi.org/10.1080/00207168308803377

[4] Saul'yev, V.K. (1965) Integration of Equations of Parabolic Type by the Method of Nets. *Proceedings of the Edinburgh Mathematical Society*, **14**, 247-248. https://doi.org/10.1017/S0013091500008890

[5] Mohd, A., Norhashidah, H. and Khoo, K.T. (2010) Numerical Performance of Parallel Group Explicit Solvers for the Solution of Fourth Order Elliptic Equations. *Applied Mathematics and Computation*, **217**, 2737-2749.

[6] Vabishchevich, P.N. (2015) Explicit Schemes for Parabolic and Hyperbolic Equations. *Applied Mathematics and Computation*, **250**, 424-431.

[7] 张宝琳. 求解扩散方程的交替分段显-隐式方法[J]. 数值计算与计算机应用, 1991, 4: 245-253.

[8] 张武生, 薛巍, 李建江, 郑纬民. MPI 并行程序设计实例教程[M]. 北京: 清华大学出版社, 2009.

[9] Gorobets, A.V., Trias, F.X. and Oliva, A. (2013) A Parallel MPI + Open MP + Open CL Algorithm for Hybridsuper Computations of Incompressible Flows. *Computers & Fluids*, **88**, 764-772.

[10] Kellogg, R.B. (1964) An Alternating Direction Method for Operator Equations. *Journal of the Society of Industrial and Applied Mathematics*, **12**, 7. https://doi.org/10.1137/0112072