多重网格区域分裂分布式计算

罗铁祥

(中南民族学院计算机科学系,湖北武汉 430074)

摘要: 对分布式多重网格计算进行了研究. 其顺序算法描述的是非递归形式,算法并行化是基于区域分裂实现的. 网状拓扑结构组织在多处理机上,并行算法映射到多进程上,在一定程度上显著提高并行化速度和并行化效率.

关键词: 分布式;多重网格计算;网状拓扑结构;并行化.
中图分类号: TP301　　**文献标识码**: A
文章编号: 1000-2383(2001)03-0323-05

作者简介: 罗铁祥(1963),男,讲师,1986 年毕业于武汉测绘科技大学,主要从事并行算法与高性能计算技术、多媒体技术的研究.

地震学、空气动力学、原子物理、核物理和等离子物理等学科的数值模拟涉及相当大的偏微分方程组. 多重网格方法是一类特殊的求偏微分方程数值解迭方法[1],在科学计算领域中应用极为广泛. 随着并行计算的发展,它的并行化引起人们的普遍重视[2,3]. 本文主要研究多重网格方法的分布式计算.

1 椭圆方程离散化与非递归多重网格迭代算法

简单的典型椭圆方程是泊松方程

$$-\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)=f. \tag{1}$$

其中,$f=f(x,y)$不恒为零,主要定解问题是边值问题,即求出未知函数 $u=u(x,y)$在某区域 Ω 内满足公式(1),并且在边界 Ω 上满足给定的条件,第一类边界条件给定函数考虑 $\Omega=\{0<x<1,0<y<1\}$,边值条件 $\Omega: u=0$ 的椭圆方程(1)定解问题. 设定 N 为自然数,取 $h=1/N$,在 Ω 上画纵横线

$$x_i=ih, y_i=jh \quad (i,j=0,\cdots,N)$$

建立等距网格,如图 1,$N=8$.

直线的交点(x_i,y_i)称格点,记为(i,j),其中 $i=1,\cdots,N-1, j=1,\cdots,N-1$. 在 Ω 内,称其为内格点;在 Ω 上的称为边界格点. 格式(i,j)的函数值 $u(x_i,y_i)$记为 $u_{i,j}$,对于内格点,用差商代替微商,得到差分方程组

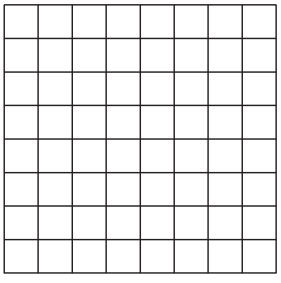

图 1 等距网格

Fig. 1 Uniformly meshed domain

$$4u_{i,j}-(u_{i-1,j}+u_{i+1,j}+u_{i,j-1}+u_{i,j+1})=h^2 f_{i,j}, (i,j=1,\cdots,N-1). \tag{2}$$

排列未知数 $u_{i,j}$,并且相应地排列方程的次序,可得

$$\boldsymbol{AU}=\boldsymbol{F}. \tag{3}$$

其中,系数矩阵 $\boldsymbol{A}$ 为块三对角线矩阵,如

$$\begin{bmatrix} \boldsymbol{S} & \boldsymbol{Q} & & & \\ \boldsymbol{Q} & \boldsymbol{S} & \boldsymbol{Q} & & \\ & \cdots & & & \\ & & \boldsymbol{Q} & \boldsymbol{S} & \boldsymbol{Q} \\ & & & \boldsymbol{Q} & \boldsymbol{S} \end{bmatrix}$$

$\boldsymbol{S}$ 和 $\boldsymbol{Q}$ 都是$(N-1)\times(N-1)$矩阵,且 $\boldsymbol{S}$ 本身又是三对角线矩阵,而 $\boldsymbol{Q}=\boldsymbol{I}/4$,$\boldsymbol{I}$ 为单位矩阵,向量 $\boldsymbol{U}$ 未知.

$$\boldsymbol{U}^{\mathrm{T}}=(u_{1,1},u_{1,2},\cdots,u_{1,N-1},u_{2,1},u_{2,2},\cdots,u_{2,N-1},$$

收稿日期: 2000-05-10
基金项目: 国家高性能计算技术基金(No. 984057).

…, $u_{N-1,1}$, $u_{N-1,2}$, …, $u_{N-1,N-1}$), 向量 F 已知, $F_{(i-1)(N-1)+j}=f_{i,j}$.

用雅可比和高斯－赛德尔型迭代法求解公式(3),其收敛性有时很难令人满意.

假定 N 为 2 的乘方,即 $N=2^L$,取 $h_1=1/2^l$, $l=0,1,\cdots,L$,在 Ω 上面画等距纵横线,得到多重网格

$$\Omega=\{(i,j)\mid 0\leqslant i,j\leqslant 2^l+1\}.$$

当 $l=0$ 时得最粗的网格, $l=\lg N$ 时得最细的网格. 从最粗的网格到最细的网格,自然地形成一个层次.

在细网格上的迭代适合求线性方程组的准确解,而在粗网格上的迭代一般能够加快收敛速度. 适当组合细网格和粗网格上的迭代,可得多重网格算法[1].

取 h_l 作细网格 Ω_l, $h_{l-1}=2h_l$ 作粗网格 Ω_{l-1}. (1)式在 Ω_l 上的离散化方程为

$$A_lU_l=F_l. \tag{4}$$

任取 $U_l^{(0)}$ 作(4)式的初始近似解,按

$$u_{i,j}^{(k)}=\frac{1}{4}(h^2f_{i,j}+u_{i-1,j}^{(k-1)}+u_{i+1,j}^{(k-1)}+u_{i,j-1}^{(k-1)}+u_{i,j+1}^{(k-1)}),(i,j=1,\cdots,N-1).$$

对 $A_lU_l^{(0)}=F_l$ 作 v_1 次平滑 S_l,得近似解 $U_l^{(v_1)}=S_l^{(v_1)}(U_l^{(0)},F_l)$,其误差 $E_l=U_l-U_l^{(v_1)}$. 将 E_l 插入方程 $A_1U_l-F_l=0$,得

$$A_lE_l=A_l(U_l-U_l^{(v_1)})=A_lU_l-A_lU_l^{(v_1)},$$

即 $A_lE_l=F_l-A_lU_l^{(v_1)}$,它是 $U_l^{(v_1)}$ 的缺陷方程. 此方程与方程(4)等价,其解 E_l 为 $U_l^{(v_1)}$ 的校正值,即 $U_l=U_l^{(v_1)}+E_l$.

缺陷方程同方程(4)一样,可以在 Ω_l 上求解. 然而,为了加快收敛速度,将 $F_l-A_lU_l^{(v_1)}$ 从 Ω_l 映射到 Ω_{l-1} 上, $R_l: F_l-A_lU_l^{(v_1)}\to D_{l-1}$,并在 Ω_{l-1} 上构造方程

$$A_{l-1}V_{l-1}=D_{l-1}. \tag{5}$$

其中 A_{l-1} 与 A_l 类似. 方程(5)的解能够在 Ω_{l-1} 上快速求出,而要作为方程(4)的近似解的校正值,它还必须从 Ω_{l-1} 映射到 Ω_l 上, $P_{l-1}: V_{l-1}\to E_l$. 校正后,再作 v_2 次平滑,可得方程(4)的新的近似解 $U_l^{(v_2)}$. 上述过程如图 2 所示.

图 2 简化得图 3a(V 形周期),重复得图 3b(W 形周期)和图 3c(μ 形周期)粗网格校正.

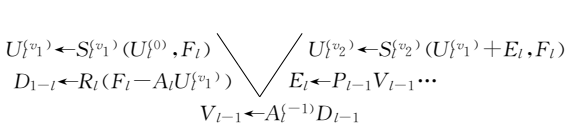

图 2 粗网格校正

Fig. 2 Coarse grid correction

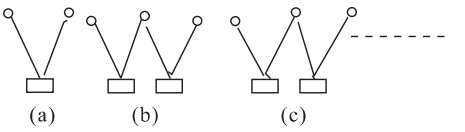

图 3 周期性精网格校正

Fig. 3 Periodic fine grid correction

图中 o. 表示平滑;\. 表示压缩(细－粗网格映射);/. 表示扩充(粗－细网格映射);□. 表示求准确解.

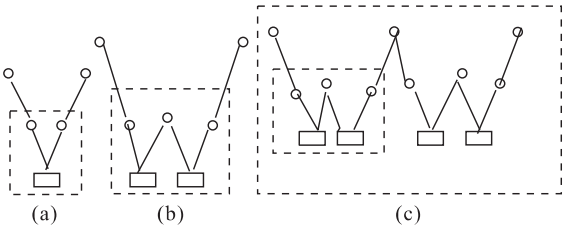

图 4 多重网格迭代

Fig. 4 Multigrid cyclic iteration

使用更粗的网格,递归地进行粗网格校正,可得 V 形周期(图 4a)、W 形周期(图 4b,图 4c)和 μ 形周期多重网格迭代.

一般而言,用 $L=\lg N$ 表示最细的网格层, l 跟踪网格的细－粗、粗－细变化, r 给定在粗网格上的迭代次数, k_l 控制 l 层网格上的迭代,可得多重网格算法[4]如下:

```
procedure  MG(L,U_l,F_l);
begin  l←L;
repeat  U_l←S_l^(v_1)(U_l,F_l);F_{l-1}←R_l(F_l−A_lU_l);
        l←l−1
until  l=0;
while  l<L  do
begin  U_0←A_0^{-1}F_0
while  k_l=γ  do
begin  k_l←0;U_l←U_l+P_{l-1}(U_{l-1},F_{l-1});l←l+1
end;
U_l=S_l^(v_2)(U_l+P_{l-1}(U_{l-1},F_{l-1}),F_l);
if  l<L  then
begin if  k_l=0  then  U_l←0;
F_{l-1}←R_l(S^(v_1)(U_l,F_l),F_l);K_l←K_l+1,l←0
end
end
end
```

此算法行为与 Hackbusch 递归算法类同,但在最精网格上的迭代次数由 $(2\gamma-1)^{(l-1)}$ 减少到 γ^{l-1},且计算复杂度降低. 收敛速度在网格加细时则不然,

是一种“渐近最优”迭代算法.

2　算法并行化与分布式计算

算法并行化与所用的计算系统有关,所用的并行机是由存储器(RAM)、32 位处理机和 4 对通讯链路组成的高性能单片微处理机(图 5). 通讯链路用以同其他并行机互连,一组并行机互连,可以形成一个多处理机系统,并发地操作,通过链路进行通讯.

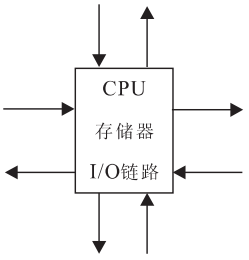

图 5　单并行处理单元

Fig. 5　Single transputer processing element

多个处理机能够配置成流水线型、网状或超立方拓扑结构,以适应算法并行设计与实现的需要. 设计并行算法的一条基本途径是重新排列网格结点的次序. 格点红－黑相间排序方法是最典型的. 若 $i+j=0(\mathrm{mod}\ 2)$,格点(i,j)称为红点,若 $i+j=1(\mathrm{mod}\ 2)$,称为黑点. 按红－黑排序,其并行平滑算法为:(1)并行更新所有的红点;(2)同时将红点的信息传给黑点;(3)并行更新所有的黑点;(4)同时将黑点的信息传送给红点.

在处理机数目不多的情况下,可将区域分裂成若干个子域,以此为基础构造并行算法,并配以与子域数目相同的处理机,实现多重网格计算与通讯的并行.

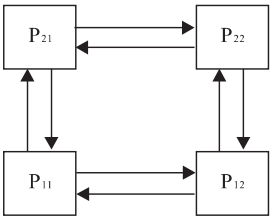

图 6　2×2 处理机网状拓扑

Fig. 6　2×2 processor mesh topology

就 4 个处理机而言,可以把它们配置成网状结构,如图 6.

区域分裂就是分裂建立在该区域基础上的所有网格,以 3 重网格为例,对它们作相同的划分,将区域 Ω 分裂成与处理机数目相同的子域,如图 7.

一个子域的多重网格计算与通讯可由一个进程负责. 程序设计可通过改变多重网格算法中格点、已知函数和未知函数的下标实现,并行程序依次映射到各处理机上,负责相邻子域多重网格计算的进程,通过发送和接收信息实现同步. 平滑使用红－黑点高斯－赛德尔松弛方法,各进程并行更新其子域内的红点,同时将内部边界上的黑点信息发送给相邻的进程,并接收相邻进程发送来的黑点信息,随后并行更新内部边界上的红点. 黑点的更新和与之相伴的信息传递是类似的. 例如,负责子域 $\Omega_{11}=\{(i,j)\mid 0\leqslant i,j\leqslant 2^{l-1}\}$ 多重网格计算和通讯的并行进程 MG_{11} 实现平滑 S 的程序段如下:

```
for  k=0   to 1   do
for  j=1   to   2^(l-1)-2   do in parallel
for  i=(j mod 2)+k   to   2^(l-1)-2   step 2 do
u_{i,j} = (1/4)(h^2 f_{i,j} + u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1})
P12 !  (u_{i,j}), i=2^(l-1)-1, j=2-k, 4-k, ..., 2^(l-1)+k-2
```

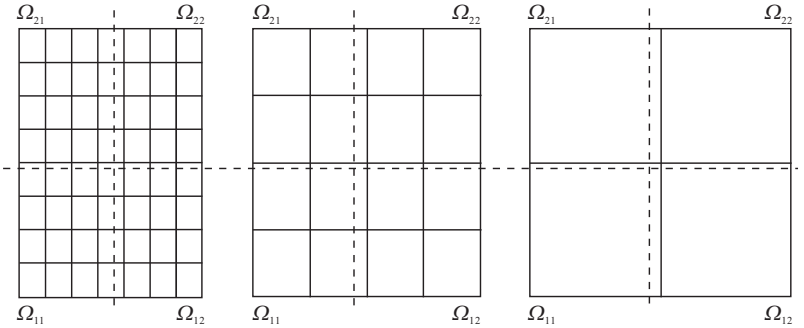

图 7　区域分裂

Fig. 7　Domain decomposition

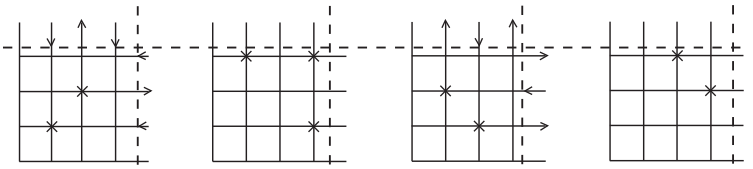

图 8 并行平滑与边界数据交换

Fig. 8 Parallel smoothing and boundary data exchange

P_{12} ? $(u_{i,j})$,i=2^{l-1},j=3−k,5−k,…,2^{l-i}−k−1

P_{21} ! $(u_{i,j})$,i=2−k,4−k,…,2^{l-1}+k−2,j=2^{l-1}−1

P_{21} ? $(u_{i,j})$,i=3−k,5−k,…,2^{l-1}−k−1,j=2^{l-1}

i=2^{l-1}−1

for j=k+1 to 2^{l-1}−k−1 step 2 do

$$u_{i,j}=\frac{1}{4}(h^2 f_{i,j}+u_{i-1,j}+u_{i+1,j}+u_{i,j-1}+u_{i,j+1})$$

J=2^{l-1}−1

for i=k+1 to 2^{l-1}−k−1 step 2 do

$$u_{i,j}=\frac{1}{4}(h^2 f_{i,j}+u_{i-1,j}+u_{i+1,j}+u_{i,j-1}+u_{i,j+1})$$

其中，！表示发送信息，?表示接收信息. 例如，P_{12} ！$(u_{i,j})$表示经连接处理机 P_{12}的输出链路向并行进程 MG_{12}发送向量$(u_{i,j})$，P_{21} ?$(u_{i,j})$表示经连接 P_{21}的输入链路从 MG_{21}那里接收$(u_{i,j})$. 具体执行情况如图 8 所示.

细−粗网格映射采取五点加权平均压缩方法，各进程并行压缩内点，同时交换边界数据(可能是单向发送或接收). 子域 Ω_{11}上的细−粗网格映射如图 9 所示.

粗−细网格映射使用插值方法，如图 9 所示，但映射方向相反. 考虑如下问题：

$$-\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)=4(\Omega=(0,1)^2),$$

$$u=x^2+y^2(\partial\Omega).$$

取 $N=128$，$h_L=1/2^{\lg N}=1/128$，确定最细的网格，其加粗使用标准化方法，处理机配置成 $m\times m(m=1,2,\cdots,5)$网状拓扑结构. 相应地，区域 Ω 分裂成 $m\times m$个矩阵子域，每个处理机上安排一个进程，负责相应子域的多重网格计算，对于每一个进程，迭代周期定为 w，先后平滑次数分别定为 2 和 1，平滑用红−黑点高斯−赛德尔松弛法，网格映射使用加权平均压缩和插值扩充方法，直接求解在最精的网格上进行.

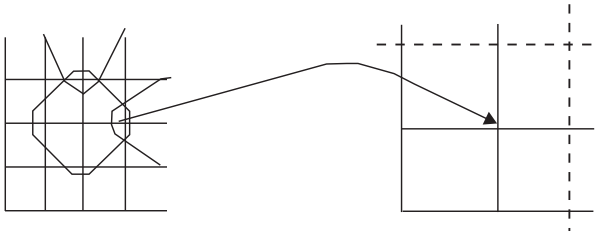

图 9 网格间映射

Fig. 9 Intergrid mapping

实验结果表明，执行(计算和通讯)时间(相对值)(表 1)对于固定精度而言，与网格精细有关，即与离散方程的未知数个数成正比. 单处理机在多重网格上执行时间为 446.36，这个结果比运行 FORTRAN 程序[1]求解同一问题的执行时间快一倍多，与多处理机执行时间相比，加速因子随网格尺寸(一个方向的格点数)大小而变. 随着网格的细化，加速因子接近 1/23，这表明：处理机数越多，格点数越大，加速效果越好，处理机的效率，即多处理机加速与数目之比，随格点数目增加而提高. 格点数目一定，增加处理机数目，效率反而有所下降，幅度与格点数目大抵上成正比. 对于最细的网格，处理机效率可高达 99%左右[3]，一般能够维持在 80%以上.

表 1 执行时间

Table 1 Executive time

网络尺寸	1	4	9	16	25
127	337.36	118.27	47.24	25.29	14.45
63	82.02	36.92	14.80	7.28	4.24
31	20.10	11.90	4.73	2.40	1.48
15	4.71	3.67	1.45	0.73	0.44
7	0.98	1.14	0.48	0.24	0.15
3	0.19	0.23	0.21	0.15	0.13

3 结论

本文叙述的非递归多重网格算法，其思想简单，具有广泛的适用性. 通过区域分裂创建的多重网格并行进程，运行时能够保持负载平衡，减少通讯开销. 在 1 台处理机上的计算速度比递归计算快一倍多，同最多 25 个处理机并行计算比较，顺序/并行加速显著，接近达到线性加速. 并行化效率提高幅度大，一般可达百分之八九十左右.

参考文献:

[1] McCormick S F. Multi-grid methods [M]. SIAM:Philadelphia, 1987.

[2] Hackbusch W. Multigrid algorithms. Introduction, survey and convergence analysis [A]. In: Spedicato E, ed. Computer algorithms for solving linear algebraic equations [C]. The State of the Art: Springer-Verlag, 1990. 133～160.

[3]Xu Z Q. Distributing multi-grid computation on a message passing system [D]. Wuhan: Huazhong University of Science and Technology, 1993.

[4]Xu Z Q, Libert G. Non-recursive multigrid computation on transputers [A]. In: Veronis A M, Paker Y, eds. Transputer research and applications 5 [C]. Amsterdam: IOS Press, 1992. 302～309.

DISTRIBUTED MULTIGRID COMPUTATION WITH DOMAIN DECOMPOSITION

Luo Tiexiang

(*Department of Computer Science, South-Central College for Nationalities, Wuhan* 430074, *China*)

Abstract: This paper presents the distributed multigrid computation. The sequential algorithm is expressed in a non-recursive form. The parallel algorithm is achieved based on domain decomposition. Mesh topology architectures are organized with multiple transputers. Parallel algorithms are mapped onto processes, increasing markedly, to some extent, the speed and efficiency of parallelization.

Key words: distribution; multigrid computation; mesh topology architectures; parallelization.

* *

(**上接** 296 **页**)

outflowing from the mountain to the oasis of the basin. The data of groundwater chemistry and isotope and geologic survey show that this model does not conform with the actual hydrogeologic situation. Based on the result of numerical simulation for generating runoff in high mountain area obtained in foreign countries, an analogue method is suggested to estimate roughly the quantity recharged laterally from mountain to oasis of basin through the basement. The construction of this new concept model may reveal the fact that a considerable quantity of water resources are buried in shallow basement rocks in the oasis of mountain-basin systems in northwestern China, which have never been counted in the reserves before.

Key words: high moutain-basin systems; water resources; recharge laterally; Hexi Corridor, Gansu.