ISSN1000-9825 Journal of Software 软件学报 2000,11(5):628~633

# 大规模问题数据并行性能的分析*

舒继武 郑纬民 沈美明 汪东升

(清华大学计算机科学与技术系 北京 100084)

E-mail: shujw@est4.cs.tsinghua.edu.cn

**摘要** 从应用的角度建立了评价大规模问题数据并行处理性能的模型,分析了区域的不同划分对解整个问题算法的收敛速度有影响时的并行性能,进而就操作重叠、数据规模以及算法选取等几个方面的问题对大规模数据并行性能所产生的影响进行了分析.最后,给出的例子证明了模型的有效性.

**关键词** 数据并行,并行处理,性能分析,加速比.

**中图法分类号** TP311

并行处理一般有功能并行和数据并行两种方式.数据并行是指,将数据划分为若干块,分别映像到不同的处理机上,每一处理机对所分派的数据进行处理,它通常以SPMD(single program multiple data)模式运行.大部分并行处理均采用数据并行处理方式,尤其是对计算复杂性很高的问题,如流体力学计算、油藏模拟、图像处理等,因此,研究大规模数据并行处理的性能是很重要的.大规模问题数据并行的性能究竟如何?文献[1~3]基于不同角度分析了数据并行性能.文献[4,5]虽然从应用的角度研究了数据并行性能,但它们是将问题的划分与整个问题的求解割裂开的,认为解整个问题算法的收敛速度与问题划分无关.在实际的并行计算中,经常由于问题划分情况的不同而影响解整个问题算法的收敛速度.如区域分解法DDM(domain decomposition method)并行计算随着区域划分的增多,收敛速度变慢.因此,在大规模问题的数据并行计算中,除了要考虑处理机间的通信及负载均衡外,还要结合问题的性质来考虑数据划分.本文建立了大规模问题数据并行性能分析模型,着重讨论了区域不同划分对解整个问题算法的收敛速度有影响的情况,并对该情况下的并行性能进行了分析,进而就影响大规模数据并行性能的几个因素进行了讨论.

## 1 数据并行计算性能模型

许多问题均有适合于并行处理的规则几何计算结构,也就是说,这些问题的计算可方便地被划分成若干个子问题的计算,子问题的计算之间通常有一定的关系.设应用问题的大小为$D\times D$,处理机为$P\times P$数目的mesh结构,每个处理机存储并处理$n\times n$个点,$n=D/P$(假设$D$是$P$的倍数),这样,共划分为$D/P\times D/P$块数据域,相邻数据域被分派到相邻处理机上,每块数据域的边缘$e$宽度的区域中各点值的计算需要用到相邻数据域的边缘点值(对不同的应用问题,$e$值通常是不同的).每个处理机的计算开销是正比于所分派的格子块的大小的,而其通信开销则正比于格子块边缘区域的大小.设每个点的计算只用到相邻4个点的值(如图1所示,其中●表示一个格子块内的数据点,○表示与其相邻格子块内的数据点).

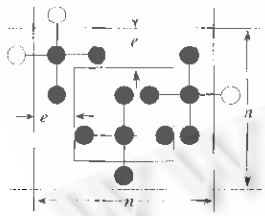


Fig. 1 A subdomain
图1 一个子区域

* 本文研究得到国家自然科学基金(No. 69933020)和国家重点基础研究发展规划基金(No. G1999032702)资助.作者舒继武,1968年生,博士,主要研究领域为并行处理,分布式计算.郑纬民,1946年生,教授,博士生导师,主要研究领域为并行计算机体系结构,并行计算.沈美明,女,1938年生,教授,博士生导师,主要研究领域为并行计算机体系结构,并行计算.汪东升,1966年生,博士后,副教授,主要研究领域为并行计算机体系结构,容错.

本文通讯联系人:舒继武,北京 100084,清华大学计算机科学与技术系

本文1998-12-07收到原稿,1999-06-10收到修改稿

设每个数据点的一次计算时间为 $t_{calc}$,相邻处理机间传输一个数据点的时间为 $t_{comm}$,每次通信所占用的处理机时间为 $t_s$.一个计算步每个处理机需要从相邻处理机获得的数据点数为 $4\times e\times n$,同时,需要发送到相邻处理机的数据点数也为 $4\times e\times n$.若计算与通信无重叠,则每个处理机的总通信时间为 $T_{comm}=8\times e\times n\times t_{comm}+8\times t_s$,每块数据的计算时间为 $T_{calc}=(n-e)^2\times t_{calc}$,则一个计算步的并行计算时间为

$$T_p=T_{comm}+T_{calc}=8\times e\times n\times t_{comm}+8\times t_s+(n-e)^2\times t_{calc}. \tag{1}$$

设当串行解整个问题达到要求精度时,算法迭代次数为 $I_1$,则串行处理时间为

$$T_{seq}=I_1\times D^2\times t_{calc}. \tag{2}$$

设在 $P$ 个处理机上并行计算,解整个问题达到同样精度要求时,算法迭代次数为 $I_p$.采用 SPMD 模式,每一计算步的计算过程是相同的,则每个子块域都计算了 $I_p$ 次.考虑到区域初值影响、各子区域物理性质不同以及均匀性不同等因素而使得解子区域的算法收敛速度不同.设在第 $i$ 次迭代中,解子区域算法的最大迭代次数为 $I_m^i$,则并行处理时间为

$$T_{par}=I_p\times(8en\times t_{comm}+8t_s)+\sum_{i=1}^{I_p}I_m^i(n-e)^2\times t_{calc}=I_p\left(8e\frac{D}{P}t_{comm}+8t_s\right)+\sum_{i=1}^{I_p}I_m^i\frac{D^2}{P^2}t_{calc}, \tag{3}$$

则加速比为

$$S_p=\frac{T_{seq}}{T_{par}}=\frac{I_1D^2t_{calc}}{I_p\left(8e\frac{D}{P}t_{comm}+8t_s\right)+\sum_{i=1}^{I_p}I_m^i\frac{D^2}{P^2}t_{calc}}. \tag{4}$$

令 $\alpha=\frac{t_{comm}}{t_{calc}}$,$\beta=\frac{t_s}{t_{calc}}$,则有

$$S_p=\frac{I_1D^2P^2}{I_p(8eDP\alpha+8P^2\beta)+\sum_{i=1}^{I_p}I_m^iD^2}. \tag{5}$$

由此可见,$S_p$ 与 $t_{calc}$,$t_{comm}$ 和 $t_s$ 的绝对值无关,而与其相对比值 $\alpha$ 和 $\beta$ 有关.此外,$S_p$ 还与 $I_p$ 及 $I_m^i$ 有关.因此,从 $S_p$ 的观点看,并行算法的设计不是着眼于 $t_{comm}$,$t_{calc}$ 和 $t_s$,而是 $I_p$,$I_m^i$,$\alpha$ 和 $\beta$.

## 2 数据并行计算性能分析

(1) 数据的划分与解整个问题算法收敛速度无关的情况.子问题的性质相同,其每一个计算步是相同的.此时,$I_1=I_p$,$I_m^i=1$,每一计算步 $S_p$ 与整体 $S_p$ 相等,加速比为

$$S_p=\frac{I_1D^2P^2}{I_p(8eDP\alpha+8P^2\beta)+\sum_{i=1}^{I_p}I_m^iD^2}=\frac{D^2P^2}{8eDP\alpha+8P^2\beta+D^2}. \tag{6}$$

有关 $S_p$ 与 $D$,$P$,$\alpha$,$\beta$ 和 $e$ 等参数变化时的关系已有相关讨论,此处不再赘述[15].

(2) 数据的划分与解整个问题算法收敛速度有关的情况.此时,$I_1\neq I_p$ 表明,在一个计算步 $S_p$ 与整个 $S_p$ 不等.此时,一般 $I_1<I_p$,由式(5)可得

$$\frac{\partial(S_p)}{\partial P}=\frac{8I_1D^2P^2\left[eD\alpha I_p-(eDP\alpha+P^2\beta)\frac{\partial I_p}{\partial P}\right]+I_1D^2P\left(2\sum_{i=1}^{I_p}I_m^i-P\frac{\partial\left(\sum_{i=1}^{I_p}I_m^i\right)}{\partial P}\right)}{\left[I_p(8eDP\alpha+8P^2\beta)+\sum_{i=1}^{I_p}I_m^iD^2\right]^2}, \tag{7}$$

令

$$\varphi(I_1,D,P,\beta,I_p,I_m^i)=8I_1D^2P^2\left[eD\alpha I_p-(eDP\alpha+P^2\beta)\frac{\partial I_p}{\partial P}\right]+I_1D^2P\left(2\sum_{i=1}^{I_p}I_m^i-P\frac{\partial\left(\sum_{i=1}^{I_p}I_m^i\right)}{\partial P}\right),$$

则有

$$\frac{\partial(S_p)}{\partial P}=\frac{\varphi(I_1,D,P,\beta,I_p,I_m^i)}{\left[I_p(8eDP\alpha-8P^2\beta)+\sum_{i=1}^{I_p}I_m^iD^2\right]^2}.$$

当 $\varphi(I_1,D,P,\beta,I_p,I_m)>0$ 时，$S_p$ 单调递增. 当 $\varphi(I_1,D,P,\beta,I_p,I_m)=0$ 时，$S_p$ 达到最大. 当 $\varphi(I_1,D,P,\beta,I_p,I_m)<0$ 时，$S_p$ 单调递减. 有关 $S_p$ 与 $I_p$，$I_m$，$P$，$D$，$\alpha$，$\beta$ 和 $e$ 等参数变化时的关系的讨论见表 1.

**Table 1**

**表 1**

| The factors of influence performance① | | | $S_p$ | |
|---|---|---|---|---|
| Variables② | Vary③ | Boundary④ | Vary | The limit value⑤ |
| $I_p$ | ↑ | $(1,D^2)_{\text{integer}}$ * | ↓ | $I_1P^2/\left(8eDP\alpha+8P^2\beta+\sum_{i=1}^{I_p}I_m\right)$ |
| $I_m$ | ↑ | $(1,D^2/P^2)_{\text{integer}}$** | ↓ | $I_1D^2P^4/I_p(8eDP^3\alpha+8P^4\beta+D^4)$ |
| $D$ | ↑ | $(0,+\infty)_{\text{integer}}$ | Undefined⑥ | $I_1P^2/\sum_{i=1}^{I_p}I_m$ |
| $P$ | ↑ | $(0,D)_{\text{integer}}$ | Undefined | $I_1D^2/\left(I_p(8e\alpha+8\beta)+\sum_{i=1}^{I_p}I_m\right)$ |
| $e$ | ↓ | $(1,D^2)_{\text{integer}}$ | ↑ | $I_1D^2P^2/\left(I_p(8DP\alpha+8P^2\beta)+\sum_{i=1}^{I_p}I_mD^2\right)$ |
| $\alpha$ | ↓ | $(0,+\infty)_{\text{real}}$ | ↑ | $I_1D^2P^2/\left(8I_pP^2\beta-\sum_{i=1}^{I_p}I_mD^2\right)$ |
| $\beta$ | ↓ | $(0,-\infty)_{\text{real}}$ | ↑ | $I_1DP^2/\left(8I_peP\alpha-\sum_{i=1}^{I_p}I_mD\right)$ |

①影响并行性能的因素，②参数，③变化，④值域，⑤极限值，⑥不定.

由于 $D$ 或 $P$ 的变化会影响 $I_1$，$I_p$，$I_m$ 的变化，因此，当 $D$ 或 $P$ 增大时，其 $S_p$ 的变化是不定的. 一般在开始时，$S_p$ 随 $D$ 或 $P$ 的增大而增大，当 $D$ 或 $P$ 增大到一定值时，$S_p$ 便随 $D$ 或 $P$ 的增大而减小，故此时的极限值不一定是最值. 对一个确定的问题来说，在应用 DDM 进行并行计算时，$I_p$ 与区域的划分数有关，$I_m$ 与子区域求解时所选取的算法有关，$\alpha$，$\beta$ 与通信性能和计算速度的比值有关，$e$ 与子区域的大小及子区域间相邻边界区域的大小有关. 因此，在实际应用中，必须全面考虑. 下面，主要就区域划分对解整个问题算法收敛速度有影响的情况来讨论操作重叠、问题规模、算法选取等对并行计算性能的影响.

### 2.1 操作重叠对并行计算性能的影响

有 3 种操作重叠. 第 1 种是一个处理机的所有通信链路并行重叠地进行数据的传输；第 2 种是一个处理机的运算操作与链路的通信以及各链路的通信均并行重叠地进行；第 3 种是每一处理机都包括一个独立的通信管理硬件，使得计算与通信可操作重叠.

对于第 1 种，有

$$T_{\text{par}}^1=I_p\left(e\frac{D}{P}t_{\text{comm}}+8t_s\right)+\sum_{i=1}^{I_p}I_m^i\frac{D^2}{P^2}t_{\text{calc}}.$$

对于第 2 种，一般大规模问题每块数据域的非边缘数据子域的计算时间大于边缘数据的通信时间，这样，在计算完非边缘数据域时立即进入边缘部分各点的计算，则有

$$T_{\text{par}}^2=I_p\left(8t_s+\max\left\{e\frac{D}{P}t_{\text{comm}},I_m^i\left(\frac{D}{P}-2e\right)^2t_{\text{calc}}\right\}\right)-\sum_{i=1}^{I_p}I_m^i\left(\frac{D^2}{P^2}-\left(\frac{D}{P}-2e\right)^2\right)t_{\text{calc}}$$

$$=I_p8t_s+\sum_{i=1}^{I_p}I_m^i\frac{D^2}{P^2}t_{\text{calc}}.$$

对于第 3 种，同样是大规模问题，每块数据域的非边缘数据子域的计算时间一般大于边缘数据点的通信时

* 问题求解的迭代次数与问题的规模有关，在选择合适的迭代算法后，其收敛次数一般不大于问题规模. 因此对于子区域大小为 $D\times D$ 的问题，$I_P$ 最多为 $D^2$.

** 同 *，对子区域大小为 $D/P\times D/P$ 的问题，$I_m$ 最多为 $D^2/P^2$.

间,此时,处理机无等待地并行计算,故有

$$T_{par}^3=\sum_{i=1}^{I_p}I_m^i\frac{D^2}{P^2}t_{calc},$$

显然有 $$T_{par}^1>T_{par}^2>T_{par}^3,\quad S_p^1<S_p^2<S_p^3.$$

在实际应用中,当通信和计算重叠时,由于通信发送和接收数据仍需软件支持,且当消息不够大时,软件延迟所占比例很大,所以在对非边缘结点进行迭代计算时,通信进程也会分时占用CPU时间,这样,通信和计算重叠虽然减少了通信时间,却使计算时间增加了,这样反而有可能造成$S_p$的下降.因此,要使计算和通信重叠能够确实提高并行计算性能,应在硬件上提供对操作重叠的支持,即设计专门的硬件来完成消息发送和接收中的软件工作,使其不占用CPU时间,亦即第3种操作重叠情况.

### 2.2 计算问题规模对并行计算性能的影响

由式(5)得

$$S_p=\frac{I_1D^2P^2}{I_p(8eD)P\alpha+8P^2\beta)+\sum_{i=1}^{I_p}I_m^iD^2}=\frac{I_1}{I_p\left(8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta\right)+\sum_{i=1}^{I_p}I_m^i}P^2.\tag{8}$$

因为$I_p\left(8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta\right)>0$,一般对大规模问题,有$D\gg P$,则

$$S_p\to\frac{I_1}{\sum_{i=1}^{I_p}I_m^i}P^2,$$

此时,$S_p$的极限值不是最大值.随着问题规模的增大,在$P,\alpha,\beta$都一定时,$I_1,I_p$和$I_m^i$都增大,而当$8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta$减小时,$S_p$不一定增大,$S_p$的变化与问题规模有关.在算法选取后,当$P$一定时,即$I_1,I_p,I_m^i$和$P$一定,由式(8)可知,要使加速比达到最大值,所解问题必须达到某一规模.当问题超过该规模时,加速比趋于下降.

如设每次迭代使解子区域算法迭代相同次数后强制退出,故可设$I_m^i$为一常数$k$,则有

$$S_p=\frac{I_1}{I_p\left(8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta+k\right)}P^2,$$

显然有$\left(8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta+k\right)>0,I_p>0$,则要使$S_p$最大,必须有

$$\left(8e\frac{P}{D}\alpha+8\frac{P^2}{D^2}\beta+k\right)=I_p,$$

即有 $$(I_p-k)D^2-8ePaD-8P^2\beta=0,$$

令 $$y=(I_p-k)D^2-8ePaD-8P^2\beta,$$

一般地,$I_p-k>0$,则当

$$D=\frac{8eP\alpha}{2(I_p-k)},$$

$$y_{min}=-\frac{32(I_p-k)P^2\beta+64e^2P^2\alpha^2}{4(I_p-k)}<0,$$

此时

$$S_{p_{max}}=\frac{I_1}{2I_p}P^2.$$

### 2.3 算法选取对并行计算性能的影响

对确定规模的问题,在硬件环境确定或计算与通信重叠方式确定时,$D,\alpha,\beta$都已确定,此时,$S_p$与$I_p$和$I_m^i$有关.因此,应选取收敛速度快的算法求解各子区域,以减少$I_m^i$.解各子区域的算法,一般随具体问题和求解要求的精度而定.同时,应恰当地选取划分的子区域数,使得解整个区域算法的迭代收敛速度较快,从而减少$I_p$.因

此,算法的选取对并行性能是有影响的,一般在尽可能减少 $I_p$ 和减少 $I_m$ 之间进行折衷.

## 3 计算结果与分析

取一个三维黑油油藏模型,模型纵向分为5层,$N_x$,$N_y$,$N_z$ 分别为170,50,5,节点规模为42 500.以该模型为例,采用并行Schwarz型DDM并行计算,主要就区域的不同划分对并行计算产生的影响进行了测试,结果见表2和表3.

**Table 2** The influence of overlapping domain on convergence speed and computation time, running phase $N=20$, 20, overlapping domain between $\Omega_i$ and $\Omega_j$ is $\Omega$

**表2** 重叠区域大小对收敛速度和计算时间的影响,运行时间段 $N=20$,20,$\Omega$ 和 $\Omega_j$ 重叠于 $\Omega$

| | | | | |
|---|---|---|---|---|
| Domain division① | $\Omega_1$: 170×25×5<br>$\Omega_2$: 170×26×5<br>$\Omega$: 170×1×5 | $\Omega_1$: 170×26×5<br>$\Omega_2$: 170×26×5<br>$\Omega$: 170×2×5 | $\Omega_1$: 170×27×5<br>$\Omega_2$: 170×27×5<br>$\Omega$: 170×4×5 | $\Omega_1$: 170×29×5<br>$\Omega_2$: 170×29×5<br>$\Omega$: 170×8×5 |
| The convergence speed② | Slower③ | Slow④ | Fast⑤ | Faster⑥ |
| Computation time⑦(s) | 30.23 | 24.29 | 27.16 | 30.66 |

①区域划分,②收敛速度,③慢,④较快,⑤较慢,⑥快,⑦计算时间(s).

**Table 3** The influence of varying the number of sub-domain on computation time, $N=20$, 20, overlapping domain $\Omega$ between $\Omega_i$ and $\Omega_j$ is 170×4×5

**表3** 取不同子区域数对计算时间的影响,运行时间段 $N=20$,20,$\Omega_i$ 于 $\Omega_j$ 重叠 $\Omega$:170×4×5

| | | | | | | |
|---|---|---|---|---|---|---|
| Domain division① | $\Omega_1=\Omega_2$:<br>170×27×5 | $\Omega_1=\Omega_2$:<br>170×19×5<br>$\Omega_3$:<br>170×20×5 | $\Omega_1=\Omega_4$:<br>170×15×5<br>$\Omega_2=\Omega_3$:<br>170×16×5 | $\Omega_1\sim\Omega_4$:<br>170×13×5<br>$\Omega_5$:<br>170×14×5 | $\Omega_1=\Omega_6$:<br>170×11×5<br>$\Omega_2\sim\Omega_5$:<br>170×12×5 | $\Omega_1=\Omega_6=\Omega_7$:<br>170×10×5<br>$\Omega_2\sim\Omega_5$:<br>170×11×5 |
| The number of processor② | 2 | 3 | 4 | 5 | 6 | 7 |
| Pressure computation time③(s) | 27.16 | 20.08 | 15.50 | 11.71 | 12.79 | 17.04 |
| $S_p$ | 1.32 | 1.79 | 2.32 | 3.07 | 2.81 | 2.11 |

①区域划分,②处理器数,③压力计算时间(s).

在并行Schwarz型DDM中,区域的不同划分对运行速度和计算时间会产生影响.若区域重叠越多,算法收敛越快,但随之计算量会增加;若区域重叠越少,算法收敛越慢,但随之计算量会减少.该结论与表2相一致,如表2中的第2种划分,也说明存在一个较优的重叠区域大小,使得整个问题的计算时间较少.对于该模型来说,当划分的子区域数为5时,加速比最大,随着子区域数的继续增加,加速比呈下降趋势,这说明在实际并行计算中要使加速比达到最大,需考虑对子区域的恰当的划分,见表3.

## 4 小 结

本文分析了区域划分对解整个问题算法收敛速度有影响时的并行性能,从中可得出如下结论:

(1) 操作重叠可以提高并行处理性能,尤其是设计专门的硬件来完成消息发送和接收中的软件工作,使其不占用CPU的时间,能更进一步地提高加速比.

(2) 加速比 $S_p$ 与 $t_{calc}$,$t_{comm}$ 和 $t_s$ 的绝对值无关,而与其相对比值 $t_{comm}/t_{calc}$,$t_s/t_{calc}$ 有关.

(3) 大规模问题数据并行处理,在处理机数目一定且问题达到某一规模时,$S_p$ 才能达到最大值.当问题的规模足够大时,在合适的数据划分、通信管理情况下,必须选择合适的处理机数目,并行处理的效果才不致于下降.

(4) 大规模问题数据并行性能常与解整个问题算法及解子问题算法的收敛速度有关,需考虑选择恰当的子区域数和高效的迭代算法,才能提高并行性能.

## 参考文献


1 Gustafson J L. Reevaluation Amdahl's law. Communications of the ACM, 1988,31(5):532~533

2 Sun X H, Gustafson J. Toward a better parallel performance metric. Parallel Computing, 1991,17(10,11):1093~1109

3 Barton M L, Withers G R. Computing performance as a function of the speed, quantity, and cost of the processors. In: Proceedings of the Supercomputing'98. New York: The Association for Computing Machinery Press, 1989. 759~764

4 Alain J M, Sophie L B, Zhen L. An analytical approach to the performance evaluation of master-slave computational model. Parallel Computing, 1998,24(5,6):841~862

5 刘德才,王鼎兴,沈美明等.数据并行的性能分析.软件学报,1994,5(5):8~15
(Liu De-cai, Wang Ding-xing, Shen Mei-ming *et al*. Potential in data parallelism. Journal of Software, 1994,5(5):8~15)


## Performance Analysis for Massive Problem Data Parallel Computing


SHU Ji-wu ZHENG Wei-min SHEN Mei-ming WANG Dong-sheng

(*Department of Computer Science and Technology Tsinghua University Beijing* 100084)



**Abstract** In this paper, from application viewpoint, a model for enhancing parallel computing performance is established. The performance of parallel computing is analyzed when different domain divisions affect convergence speed in solving the whole problem. In addition, the major factors affecting the performance of massive data parallel computing are analyzed, such as overlapping operation, the problem data size and selective algorithm. Finally some computational examples provide evidence for the usefulness of the model.



**Key words** Data parallelism, parallel processing, performance analysis, speedup.