计算机科学2002Vol. 29№. 3

基于 NOWs 分布式共享存储系统的并行图像处理数据通讯研究*)

Analysis on Data Communication in Parallel Image Processing System Based on Distributed and Shared Memory System in NOWs

黄国满　何建邦

(中国科学院地理科学与资源研究所　资源与环境信息系统国家重点实验室　北京100101)

Abstract In this paper, we analyze data communication in parallel image processing system based on distributed and shared memory system in PC or workstation group. As our views, image processing should be classified as point, line and area processing and the amount of data communication should be classified as net and real communication. By analyzing the net and real amount of communication, we propose a strategy of data partition in these three kinds of image processing which are based on distributed and shared memory system of NOWs. In order to improve the efficiency of the system we also present an approach of balancing data communication in the whole computing process by adjusting the sequence of computing.

Keywords Distributed and shared memory system, Parallel image processing, Data communication

1. 引言

目前，并行分布式系统的应用正在不断地发展和扩大。将现有的微机、工作站用高速网络连接起来，再配上并行分布式计算的软件环境，便能实现高速的并行计算[1]。这将有利于普及并行计算。在所有类型的并行系统中，基于分布式存储的工作站集群(NOWs)由于具有很高的性价比，越来越受到高性能计算领域的重视[2~4]。

数字图像处理具有内在的并行性，建立基于微机、工作站集群的并行图像处理系统，将有利于实时处理海量数字图像系统的实用化。

分布式系统的并行机制主要有两种，即消息传递和共享存储。但它们不是完全对立的，一个系统可以同时采用这两种机制，不过有所侧重，习惯上依其侧重点的不同分别称为消息传递系统和共享存储系统。在消息传递系统中，通过仔细设计消息传递可以获得高效率，但设计消息传递的过程使得编程较难；而精心设计的共享存储系统的效率跟消息传递系统相差不多，且易于编程[5,6]。因而在非计算机专业人员作为主要编程人员的一些图像处理领域里，分布式共享存储系统更有市场。

在共享存储系统中，数据传递是由系统完成的。要控制数据通讯量，只能通过调节数据划分来实现。本文就是基于这样的目的，对分布式共享存储系统环境下的并行图像处理的数据划分作了初步的研究。

2. 并行图像处理的数据划分

分布式并行图像处理中，图像数据的划分主要有四种：水平条带、竖直条带、矩形块及不规则划分，其中不规则划分用得较少，这里只考虑前三者：水平条带、竖直条带、矩形块。

一般将图像处理分为点处理和域处理两类[7]。作者认为，将图像处理分为这样两类不尽合理，根据输入元素的分布情况，图像处理分为点处理、线处理和域处理三类。点处理是指输入元素只涉及到一个像素的图像处理；线处理是指输入元素集中在一行或一列的图像处理，输入元素集中在一行上的线处理称为水平线处理，输入元素集中在一列上的线处理称为竖直线处理；域处理是指输入元素分布在一个矩形区域内的图像处理。

从以上定义可以看出，线处理是域处理的一种特例。之所以要将线处理从域处理中分离出来，是因为：①线处理有其独特性，这样更有利于分析数据划分；②线处理在图像处理中用得很多，一些域处理常常可以化为行和列两个方向上的线处理，比如二维傅里叶变换就可以转换为两个方向上的一维傅里叶变换，这种转换会有利于这类算法的并行化。

显然，在拥有者计算原则下，点处理不需要用到远程数据，并行化开支最小，数据划分最简单，可根据情况任选一种划分方法；线处理分两种情况，即水平线处理和竖直线处理，最好分别按水平条带和竖直条带划分；而对于域处理，无论采用何种方式划分，都不可避免远程数据调用，究竟采用何种划分方法，要视并行计算支撑环境而定。

选择数据的划分方式，依据是数据通讯量。数据通讯量应当进一步细分为净通讯量和实际通讯量。

定义1(净通讯量)　净通讯量是指并行运算确切需要的数据通讯的量。

定义2(实际通讯量)　实际通讯量是指并行运算过程中并行计算支撑环境实际传送的数据量。

定义3(连带通讯量)　连带通讯量是指实际通讯量与净通讯量的差。

相应地，还有以下定义：

定义4(净通讯域)　净通讯域是指并行运算确切需要传送的数据的分布范围。

定义5(实际通讯域)　实际通讯域是指并行运算过程中并行计算支撑环境实际传送的数据的分布范围。

*)本文得到国家自然科学基金重大项目69896250、中科院"九五"基础性研究重大项目 KJ951-B1-703支持。黄国满　博士研究生，长期在地理信息领域从事计算机图像处理研究与软件系统开发。何建邦　研究员，博士生导师，欧亚科学院院士，长期从事地理信息系统和信息共享研究。

定义6(连带通讯域) 连带通讯域是指实际通讯域与净通讯域的差。

人们通常使用的"通讯量",有时是指"净通讯量",有时又是指"实际通讯量",这实际上是不够准确的。从以下分析可以看出净通讯量和实际通讯量的区别和意义。

3. 数据划分的通讯量分析

为了简化问题,以下的分析是基于这样两个假设的:

① 并行分布式系统是同构的;

② 图像处理的计算量与数据量成正比。

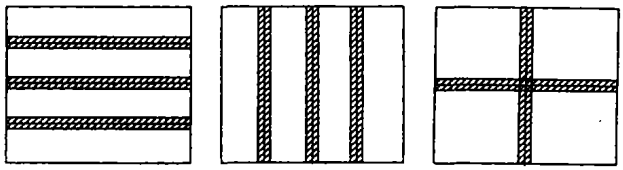
3.1 三种数据划分的净通讯量分析

如果只考虑净通讯量,域处理数据划分的方式应采用接近正方形的矩形块划分。原因是,净通讯量与矩形的周长成比例,而面积一定的矩形中,正方形的周长最短。例如,对于长和宽均为m的一幅图像,要用4台处理机对它作5*5的域处理,我们来比较三种划分方式下的净通讯量。

①*按水平条带划分* 如图1a,阴影部分是净通讯域。净通讯域有六个。净通讯量为2*m*6像素。

②*按竖直条带划分* 如图1b,净通讯域也是六个。净通讯量为2*m*6像素。

③*按矩形块划分* 如图1c,净通讯域是四个。净通讯量为2*m*4像素(仔细分析,中央相交之处应该是三度重叠,还有2*2*4=16像素,但(2*2*4)/(2*m*4)=2/m是微小量,为便于分析,忽略不计)。



a. 水平条带 b. 竖直条带 c. 矩形块

图1 三种划分方式下的净通讯量

显然,按水平条带和竖直条带划分,净通讯量是相同的,而按矩形块划分,净通讯量最小。更一般地,设处理机数为k(k≥4),域处理范围是(2*n+1)*(2*n+1),并假定n、k相对m来说较小(n*k*2<m),则按条带(水平条带和竖直条带)划分时,净通讯量为:

$$n*m*(k-1)*2\text{像素} \quad (1)$$

而按矩形块划分,净通讯量为:

$$n*m*(\mathrm{sqrt}(k)-1)*2*2\text{像素} \quad (2)$$

由(1)、(2)两式可得:按条带划分和按矩形块划分的净通讯量之比为:

$$(k-1)/((\mathrm{sqrt}(k)-1)*2) \quad (3)$$

为了更直观地表示这两种方式的净通讯量之比,按(3)式根据处理机数列表如下:

表1 按条带划分和按矩形块划分的净通讯量之比

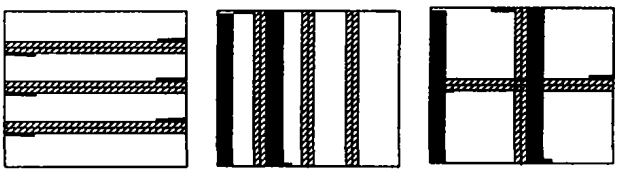
处理机数k	4	9	16	25	36	49
净通讯量之比	1.5	2.0	2.5	3.0	3.5	4.0

可以看出,上述"采用接近正方形的矩形块划分可以使净通讯量最小化"的论断得到了验证。在消息传递系统如PVM环境下,程序员可以严格控制消息传递函数,使之恰好传送所需的远程数据,因而应该优先采用这种按矩形块划分的方式。而对分布式共享存储系统来说,情况就要复杂一些,必须考虑实际通讯量。

3.2 三种数据划分的实际通讯量分析

在分布式共享存储系统(DSM)中,为了减小编程的难度,远程数据传送由DSM系统完成,而不是由应用程序来控制。当应用程序用到的数据是远程数据时,DSM系统能侦测到,并在后台做好传送工作。如果DSM系统发现远程数据调用时只传送被用到的数据,那么用来传送数据的消息就很多,降低了效率。因此,DSM系统一般都以页面为单位传送数据,在UNIX系统中每个页面一般是4k字节。这样,实际通讯量就要大于净通讯量。

令页面大小为p(为方便计算,p化算到以像素为单位),其他条件与上同。如图2,分三种情况(即按水平条带、竖直条带和矩形块划分)来分析实际通讯量。



a. 水平条带 b. 竖直条带 c. 矩形块

图2 三种划分方式下的实际通讯量

①*按水平条带划分* 如图2a,阴影部分是净通讯域,短线条表示连带通讯域。连带通讯量最好的情况下是0,最坏的情况下是(p-1)像素,所以实际通讯量在最坏的情况下为:

$$(n*m+(p-1))*(k-1)*2\text{像素} \quad (4)$$

②*按竖直条带划分* 如图2b,短线条相当于图2a的短线条,和实心矩形一起表示连带通讯域。图中只标出了第一条和第二条之间的远程数据调用情况,其他各条之间是类似的。如果条带的宽度大于p,实心矩形块的宽度为(p-n),考虑最好的情况,即短线条长为0,则实际通讯量不少于:

$$(n+p*(m-1))*(k-1)*2\text{像素} \quad (5)$$

否则,实际通讯量是整个影像的(1+(k-2)/k)倍。

③*按矩形块划分* 如图2c,短线条相当于图2a的短线条,和实心矩形一起表示连带通讯域。与图1c一样,忽略中央相交之处的三度重叠部分。上下矩形块之间的实际通讯量约为$(n*m+\lambda(p-1))*(\mathrm{sqrt}(k)-1)*2$像素,$\lambda\in(0,1)$,考虑最好的情况,其值为$n*m*(\mathrm{sqrt}(k)-1)*2$像素;左右矩形块之间的实际通讯量约为$(n+p*(m-1))*(\mathrm{sqrt}(k)-1)*2$像素,所以总的实际通讯量约为:

$$(n*m+n+p*(m-1))*(\mathrm{sqrt}(k)-1)*2\text{像素} \quad (6)$$

不难看出,按竖直条带划分,实际通讯量最大;按矩形块划分实际通讯量次之;按水平条带划分,实际通讯量最小。

由(4)、(5)两式可得:按竖直条带划分和按水平条带划分的实际通讯量之比不小于:

$$(n+p*(m-1))/(n*m+(p-1)) \quad (7)$$

由(4)、(6)两式可得:按矩形块划分和按水平条带划分的实际通讯量之比不小于:

$$(n*m+n+p*(m-1))*(\mathrm{sqrt}(k)-1)/((n*m+(p-1))*(k-1)) \quad (8)$$

以一景m=10000的三波段点交叉影像为例,对于4K页面,p=4096/3≈1365,设域处理采用5*5的算子,则n=2,(7)式的值大于638;若k=4,(8)式的值大于567。

这里,按矩形块、竖直条带划分的实际通讯量是按水平条

带划分的实际通讯量的五、六百倍，因此，对于分布式共享存储系统，域处理的数据划分应该优先采用按水平条带划分的方式。

4. 运算次序与通讯的时间分布

并行图像处理系统的效率不但与数据通讯量有关，还与数据通讯在时间上的分布有关。

按照自然的运算次序，各处理机都从分配给它数据的起始处开始逐行、逐像素处理。这里仍然假定并行处理系统是同构的，且计算量与数据量成正比。记处理机台数为 hosts，处理机顺序编号为 pid(从0起算)，称处理机(pid+1)为处理机 pid(0≤pid≤hosts-2)的下邻处理机，处理机(pid－1)为处理机 pid(1≤pid≤hosts－1)的上邻处理机。根据以上分析，将待处理数据(大小为 width * height)按水平条带平均划分成 hosts 个条带，每台处理机负责一个条带的运算。那么，各处理机(对应编号为 pid)对它所承担的条带的处理可用 C 语言描述为：

```
int start,end;
start=pid * height/hosts;
end=(pid+1) * height/hosts;
for (int i=start; i<end;i++)
    for (int j=0; j<width;j++)
        process (i,j);
```

这样的思路是很自然的，但容易造成以下局面：就是各处理机要么各忙各的，执行那些不需要远程数据参与的处理，使得网络处于空闲状态；要么同时要求从别的处理机那里获取远程数据参与运算，使得网络处于拥堵状态，致使程序效率不高。具体情况是：当运算刚开始时，除了第0台处理机以外，每台处理机都向它的上邻处理机要求远程数据参与运算，使得网络通讯繁忙；这样处理数行以后，行数超过了域处理的一半的宽度时，所有的处理机都不需要远程数据参与运算，使得网络处于空闲状态；而当运算行将结束，待处理的行数不足域处理的一半的宽度时，除了最后一台处理机以外，每台处理机都向它的下邻处理机要求远程数据参与运算，使得网络通讯再次繁忙。也就是说，数据通讯集中在运算刚开始和行将结束时，其他时间网络均处于空闲状态。

如果适当地调整各处理机的运算次序，就可以缓解这样的情况，使得数据通讯在时间上趋于均匀分布，减少网络拥堵现象，提高程序效率。

方法是将每个条带再按行平均细分为(hosts－1) * 2个细条，依次编号为0到(hosts－1) * 2－1。每个处理机从不同编号的细条开始运算，运算完细条(hosts－1) * 2－1后，再回过头来从细条0起运算，直至最初运算的起始处。除了第一台处理机(pid＝0)外，运算0号细条时将向其上邻处理机取远程数据；除了最后一台处理机(pid＝hosts－1)外，运算(hosts－1) * 2－1号细条时将向其下邻处理机取远程数据。

令 h＝(hosts－1) * 2，各处理机均从细条 pid * 2%h 开始运算(其中"%"表示取模运算)。这样的安排将保证数据通讯被分散在整个运算过程中。以 hosts＝8为例，将每个条带按行平均细分为 h＝(hosts－1) * 2＝14个细条，可以用表2来说明运算过程。

表2 运算及通讯过程

pid	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14
0	0	1	2	3	4	5	6	7	8	9	10	11	12	
1	2	3	4	5	6	7	8	9	10	11	12			1
2	4	5	6	7	8	9	10	11	12			1	2	3
3	6	7	8	9	10	11	12			1	2	3	4	5
4	8	9	10	11	12			1	2	3	4	5	6	7
5	10	11	12			1	2	3	4	5	6	7	8	9
6	12			1	2	3	4	5	6	7	8	9	10	11
7		1	2	3	4	5	6	7	8	9	10	11	12	13

表中，T1～T14为计算的步次，T1对应的列是各处理机运算启动时所处的细条编号，此时处理器7需要从上邻取远程数据；T2对应的列是各处理机第二步计算时处理机运算所处的细条编号，此时处理器6需要从下邻取远程数据；依次类推，可以看到，14个实际通讯域在表中呈阶梯状地平均分配在整个运算期间的14个区间中。

当然，运算次序的安排还有许多种，比如，各处理机均从细条 pid * 2%h＋p(0≤p≤hosts－1)开始运算，也能取得相似效果，只是上表的阶梯被抬高 p 级，并在阶梯爬升到最高处时跌落到底，然后重新爬升。

各处理机(对应编号为 pid)对它所承担的条带的处理可改写为：

```
int start,end,h;
h=(hosts－1) * 2
start=pid * height/hosts＋pid * 2%h * (height/hosts)/h;
end=(pid+1) * height/hosts;
for (int i=start; i<end;i++)
    for (int j=0;j<width;j++)
        process (i,j);
for (i=pid * height/hosts; i<start;i++)
    for (j=0; j<width;j++)
        process (i,j);
```

结论 与并行图像处理数据划分相适应，图像处理应分为点处理、线处理和域处理三类；数据通讯量应当进一步细分为净通讯量和实际通讯量。在基于微机、工作站集群的分布式共享存储系统中，点处理不需要用到远程数据，可根据情况任选一种划分方法；水平线处理和竖直线处理，最好分别按水平条带和竖直条带划分；域处理的数据划分应该优先采用按水平条带划分的方式，以减少实际通讯量。除了实际通讯量以外，影响并行处理效率的还有通讯行为在运算过程中的时间分配，需要适当调整计算的次序，以使通讯行为按时间均匀分布在整个运算过程中，合理利用网络传输。

参 考 文 献

1 方金云，等. 基于机群的地理数据并行处理试验. 见：中国地理信息系统协会第六届年会论文集(2)，2001，158～162
2 孟杰，王小鸽，李三立. 并行计算机性能的分析与预测. 计算机科学，1999，26(2)：14～17
3 胡凯，王强，胡建平. 机群并行计算机中负载共享的关键问题. 计算机科学，2000，27(7)：8～11
4 彭德纯，邱毓兰，林子禹. 分布式并行处理技术导论. 武汉：武汉大学出版社，1996，5
5 Lu H，Dwarkadas S，Cox A，et al. Quantifying the performance differences between PVM and ThreadMarks. Journal of Parallel and Distributed Computing，1997，43(2)：65～78
6 唐志敏，施巍松，胡伟武. 曙光1000A 上消息传递与共享存储的比较. 计算机学报，2000，23(2)，134～140
7 Castleman K R.(朱志刚等译). 数字图像处理. 北京：电子工业出版社，1999