计算机科学 1999Vol. 26№. 4

网络环境下的并行计算模型*)

Parallel Computational Model under Network Environment

吴洁明　　计永昶　陈国良

（广西大学梧州分校计算机系　梧州 543002）（中国科学技术大学计算机系）

Abstract NHBSP parallel computational model was presented to take consideration of the two major characteristics of NOW-Nondedicated and Heterogeneous, and was aimed to reflect the influence of them to the performance of parallel algorithms on NOW. Then the cost of PSRS algorithm and Matrix multiplies algorithm was analyzed with NHBSP model and the analysis was validated by experiment results.

Keywords Networks of workstations, NHBSP parallel computational model, PSRS algorithm, Matrix multiplies algorithm, Message passing interface

1. 引言

所谓网络并行计算就是将一群计算机系统用网络以某种结构互连起来，充分利用各系统资源，统一调度、协调处理，以实现高效的并行处理。它是并行分布式计算领域近年来比较活跃的研究领域。由于网络技术的发展，特别是局域网中高速数据通讯网络的出现，使得利用工作站集群(NOW 或 COW)作为并行计算的平台越来越具有吸引力，同时也出现了许多支持异种机网络计算的软件工具环境，如 MPI[1]、PVM、EXPRESS 等。NOW 与巨型机和 MPP 系统相比，具有很高的性能价格比，可扩展性好，结构灵活。它与以往的并行系统之不同，主要表现在其异质性和非独占性，即组成系统的工作站的计算能力各不相同且工作站上有其他用户的计算任务在执行。原有的并行计算模型[2,3]，如 PRAM、BSP、LogP、C^3 等模型未能体现这两大特点，从而无法为 NOW 上的用户提供可用的并行计算能力，因此我们提出 NHBSP(Nondedicated Heterogeneous BSP)模型。

本文首先介绍了 NHBSP 并行计算模型，然后给出了 PSRS(Parallel Sorting by Regular Sample)算法和矩阵相乘算法在工作站集群与 MPI 环境下的实测结果以验证我们的分析。

*)本文工作获高校博士点基金资助。

2. NHBSP 模型

NHBSP 模型是一个与体系结构无关的并行计算模型，其主要目标是在异质非独占的 NOW 计算环境下，反映用户负载和计算节点的异质性等对算法的影响，它是 BSP 模型[4]在 NOW 环境下所作的扩展。我们之所以选择 BSP 模型作为我们扩展的基础，主要是考虑到 BSP 模型在很多体系结构上都取得了成功，同时它将通信操作看作一个整体能使我们很方便地将用户网络负载的影响考虑进我们的计算模型中去。

NHBSP 的参数如下，p：计算模块的数目；g：网络渗透性能参数；o：为处理器参与每个消息传送和接收的时间，在这段时间处理器不能进行其他操作；l：为一次 Barrier 同步的代价；S_i：各计算模块的计算速度；O_i：其他用户计算任务在第 i 个机器上的平均执行时间；P_i 在给定时间内，其他用户计算任务到来的平均概率。

由 Barrier 同步可将算法分成若干个超级步，每个超级步的代价由计算代价、通信代价和同步代价组成，所以并行计算的总代价为各超级步计算代价和通信代价的和，再加上同步时间。下面就分别对这些代价的计算进行讨论：

①无用户计算的计算代价：假定在一个超级步

时 P_i 执行的并行计算量为 C_i，P_i 的计算速度为 S_i，则计算时间为 $T_{cpui}=C_i/S_i$。

②有用户负载的计算代价：实际的计算时间受用户负载的影响。当一个用户计算进程开始时，并行计算进程被挂起，用户计算进程在工作站上执行 O_i 时间。一旦用户进程完成计算，并行计算启动执行，并保证执行 τ 时间，τ 时间为操作系统的时间片时间。在有用户负载的情况下，计算时间为：

$$T=\max_{processes}(T_{cpu}+n*O_i)$$

其中 n 为用户进程请求的次数。用户进程可以在每次并行计算进程使用完工作站后提出，因此用户计算请求的数目满足二项式分布[5]，所以：

$$T=\max_{processes}(T_{cpu}+\sum_{j=0}^{v}O_i\times j\times\binom{v}{j}P_i^j(1-P_i)^{v-j})$$

其中 $v(=T_{cpu}/\tau)$ 是在并行计算执行时间内最多可能出现的用户计算的数目。

③通信代价：我们沿袭 BSP 模型的观点，将通讯操作看作一个整体。由于 BSP 模型的实现采用延迟通信，不区分 m 个单个消息和一个长为 m 的消息，它的通信时间为 $m*g+2*o$。对于总线型 Ethernet 网络这种分时独占的共享资源，总的通讯时间为 $\sum_{i,j}(M_{i,j}*g+2*o)$，其中 $M_{i,j}$ 为进程 i 发送给 j 的消息长度。

3. 实例分析

我们使用一台 SUN Sparc-5 和一台 SUN Ultra-Sparc 工作站组成 NOW 计算环境，并在上面实现了 MPI 环境。我们测出它在用户负载为 10%情况下的参数如下：

o：103 μs；g：92.42 μs/byte；S_1：27.679Mflop/s；S_2：11.779Mflop/s。

下面以 PSRS 算法和矩阵相乘算法为例，用 NHBSP 模型进行分析，并与实际的运行结果进行比较。

3.1 PSRS 算法描述

PSRS 算法分为四步（设有 n 个数据，p 个处理器，$w=n/p^2$，$r=p/2$）。

（1）每个处理器将自己的数据用串行快速排序（Quicksort）法排好序。然后各自选第 1，$w+1$，$2w+1$，……，$(p-1)w+1$ 个共 p 个数据作为代表元素，送到处理器 M_0 中。

（2）M_0 将上步送来的 p 个有序数据列做 p 路归并，再选择排序后的第 $p+r$，$2p+r$，……，$(p-1)p+r$ 个共 $p-1$ 个主元送到各处理器中。

（3）每个处理器根据上步送来的 $p-1$ 个主元把自己的 n/p 个数据分成 p 段。

（4）p 个处理器间多对多交换数据，从而使得第 i 个处理器含有所有 p 个处理器的第 i 段数据（$1\leqslant i\leqslant p$）。再通过归并排序将每个处理器自己的数据排好序。

3.2 PSRS 算法的代价分析

PSRS 算法的第一步本地快速排序 n/p 个数的代价为 $(n/p)\log(n/p)$，选取代表元素的代价为 p；第二步 p 路归并和选主元的代价为 $p^2+\log p-p$；第三步数据划分的代价为 $p+n/p$；第四步 p 路归并的代价为 $n/p+\log p$。这样，总的计算代价为 $(n/p)\log(n/p)+p^2+3p+2n/p+2\log p$。计算时间 $T_{cpui}=((n/p)\log(n/p)+p^2+3p+2n/p+2\log p)/S_i$，有用户计算的计算时间为 $T=\max_{processes}(T_{cpu}+\sum_{j=0}^{v}O_i\times j\times\binom{v}{j}P_i^j(1-P_i)^{v-j})$，其中，$v=T_{cpui}/\tau$。通信代价可以如下估计，我们使用的实验环境为广播型网络，收集 p^2 个样本的代价为 $(p-1)(g*p+2*o)$，广播 $p-1$ 个主元的时间为 $(p-1)(g*p+2*o)$，数据多对多交换时总通信量为 $n-n/p$，通信代价为 $g*(n-n/p)+2p*(p-1)*o$，所以总的通信代价为：$ng-ng/p+2p^2g-2p^2+2po-4o+2p^2o$。整个算法的代价为总的计算代价＋总的通信代价。

3.3 PSRS 算法的实验结果

我们设 $p=2$，$O_i=2s$，$\tau=1s$，$P_i=1/18$，为方便估计，我们忽略公式中一些无关紧要的部分。表 1 和图 1 给出了 PSRS 算法在工作站集群上的实验结果与预测结果以及它们的对照。因为 PSRS 算法的通

表 1 PSRS 的实测结果与预测结果

规模(n)	实测结果	预测结果	误差度
10000	0.14s	0.08s	43%
100000	0.76s	0.46s	39%
200000	1.88s	0.90s	52%
300000	2.38s	1.36s	43%
400000	3.19s	1.80s	44%
500000	3.91s	2.30s	41%
600000	4.89s	2.81s	43%
700000	5.72s	3.20s	44%
800000	6.34s	3.72s	41%
900000	6.92s	4.10s	41%
1000000	7.32s	4.70s	36%

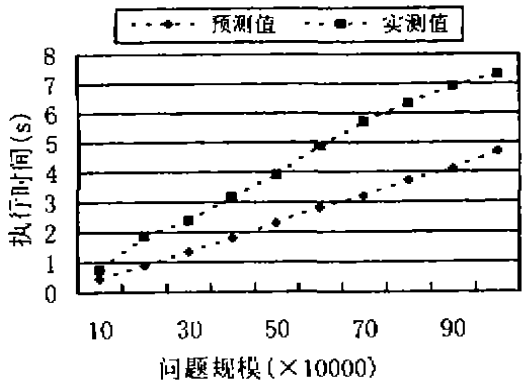

图 1 PSRS 预测值与实测值的比较图

信时间为 $O(n)$，计算时间为 $O(n\log n)$，但通信时间在目前的问题规模下，占了总时间的主要部分，用户负载对算法的性能影响并不明显。随问题规模的增大，总时间成线形增长趋势，与实测的趋势一致。从实验结果来看，预测值与实测值误差约为 40%，基本可以接受。

3.4 矩阵相乘算法的实验结果

按照同样的方法我们对矩阵相乘算法[6]进行分析，并与实际运行结果进行比较，如图 2。我们测量了四个进程在两个处理器上分布的两种情况，其一是计算速度较快的机器承担 3 个进程，另一台机器承担 1 个进程，其二是平均分布，在两台机器上各分布 2 个进程。实验结果表明误差在 20—30%左右，实测结果与预测结果反映的执行时间与问题规模大小的关系基本一致。

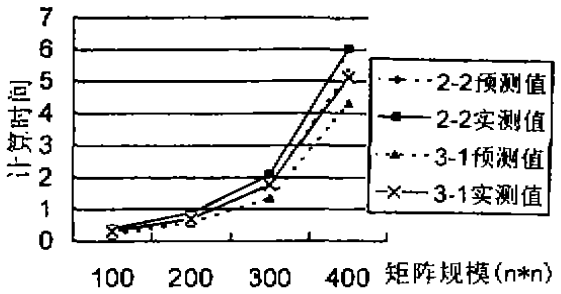

图 2 矩阵乘法实测值与预测值的比较

结束语 可见 NHBSP 并行计算模型考虑了 NOW 环境的两大特点，反映了用户负载和计算节点的异质性等对算法的影响。该模型对 PSRS 算法和矩阵相乘算法的分析结果与实测结果的比较，表明了该模型具有一定的可用性，性能预测也在可接受的准确度之内。

参 考 文 献

1 Message Passing Interface Forum. MPI: A message-passing interface standard. Intl. J. of Supercomputer Applications, 1994, 8(3/4)

2 陈国良 更实际的并行计算模型. 小型微型计算机系统, 1995, 16(2)

3 计永昶, 卜添, 陈国良 并行播送和求和算法在几种实际计算模型上的设计和分析. 中国科学技术大学学报, 1995, 26(2): 95～103

4 Valiant L G. A Bridging Model for Parallel Computation. CACM, 1990, 33(8): 103～111

5 Yan Yong, et al. An effective and practical performance prediction model for parallel computing on nondedicated heterogeneous NOW. Journal of Parallel and Distributed Computing, 1996, 38: 63～80

6 计永昶, 卜添, 陈国良. 网络计算环境下并行算法及其可扩放性分析. 计算机研究与发展, 1997, (11): 844～849

(上接第 25 页)

Cube；允许对操作关系进行选择操作，在满足条件的原始关系的子集合上执行 Gcube 操作；既可以计算对应于 GCUBE BY 属性集合的所有 *cuboid*，也可以选择计算特定的 *cuboid* 集合，而忽略其他的 *cuboid*，以提高效率；既可以在计算所有 *cuboid* 时使用相同的聚集函数集合，也可以使不同的 *cuboid* 使用不同的聚集函数集合。

本文还提出了一个实现 Gcube 操作的简单算法，并研究了提高 Gcube 操作效率的优化技术。我们将进一步深入研究高效率实现 Gcube 操作的算法。

参 考 文 献

1 Agrawal R, et al. Modeling Multidimensional Databases. In: Proc. of the 13th Int'l Conference on Data Engineering. Birmingham, U. K., 1997. 105～116

2 Gray J, et al. Data Cube: A Relational Operator Generalizing Group-By, Cross-Tab and Sub-Totals. In: Proc. of the 12th Int. Conf. on Data Engineering, 1996. 152～159

3 Agarwal S, et al. On the Computation of Multidimensional Aggregates. In: Proc. of the 22nd VLDB Conference. Mumbai, India, 1996. 506～521

4 Harinarayan V, et al. Implementing Data Cubes Efficiently. In: Proc. of ACM SIGMOD Conf., 1996. 205～216