

MPI+OpenMP 混合编程模型在 大规模三对角线性方程组求解中的应用

郑汉垣^{1,2}, 刘智翔¹, 封卫兵¹, 张 武¹

(1 上海大学 计算机工程与科学学院, 上海 200072; 2 龙岩学院 计算机系, 福建 龙岩 364000)

摘 要: 分布式共享存储系统的特点是每个节点内是共享存储的, 而节点间是分布式存储。为了更好地利用这种多级体系结构, 讨论了 MPI+OpenMP 混合编程模型的性能及实现方法, 建立了大规模三对角线性方程组的 MPI+OpenMP 混合并行算法, 并在上海大学高性能计算集群上与单纯 MPI 算法进行了性能方面的比较。结果表明, MPI+OpenMP 混合并行算法具有更好的加速比和扩展性。

关键词: 混合编程模型; 分布共享存储; 三对角线性方程组; MPI

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-7180(2011)08-0158-04

MPI and OpenMP Paradigms and Its Application of Solving Large Scale Tridiagonal Linear Systems

ZHENG Han-yuan^{1,2}, LIU Zhi-xiang¹, FENG Wei-bing¹, ZHANG Wu¹

(1 School of Computer Science, Shanghai University, Shanghai 200072, China;

2 Department of Computer, Longyan University, Lingyan 364000, China)

Abstract: The distributed shared memory system is characterized by shared memory multi-processors on each node and distributed memory among nodes. In order to make use of this hierarchical architecture, this paper discusses the performance of MPI+OpenMP hybrid programming paradigm and different implementations. We design a multi-granularity parallel algorithm for solving larger scale tridiagonal linear systems, and compare its performance with pure MPI algorithm on the high performance computer of Shanghai University. The results indicate that the hybrid algorithm shows better speedup and scalability.

Key words: hybrid paradigm; distributed shared memory; tridiagonal linear systems; MPI

1 引言

目前, 分布共享存储并行机是高性能计算机发展的主要趋势, 其主要特点是: 每个节点内部是一个共享存储的多处理器, 而节点间是分布存储且通过网络(Myrinet, Infiniband, Internet 等)进行连接。为了能够充分利用分布共享存储系统的结构特点, 发挥更好的性能, 需要寻找一种有效的方法来使用这种特殊的多层的体系结构。在分布存储系统上, 使用

MPI 的消息传递已经被证实是理想的程序设计模型。而在共享内存系统上, 基于编译制导的并行标准是 OpenMP, 制导主要处理工作分配。在分布共享存储系统中, 存在着多种并行程序设计方式, 包括使用单一消息传递的纯 MPI 模式, 运行于 SMP 节点的 OpenMP 模式, MPI 和 OpenMP 相结合的混合模式。由于具有共享存储和分布存储这种两级存储结构, 因此节点内通信使用内存访问(OpenMP), 节点间使用消息传递(MPI)的并行模式能更好地利用分

收稿日期: 2011-06-13; 修回日期: 2011-06-30

基金项目: 上海市科委重点项目(10510500600); 教育部 2008 年度高等学校博士学科点专项科研项目(200802800007); 上海市重点学科建设项目资助基金项目(J50103)

布共享存储系统的特点. 混合编程模型已经被应用到许多科学应用中^[1]. 通常, MPI 和 OpenMP 的混合编程模型实现起来较为复杂, 因为程序需要针对两种存储模式进行操作, 使得程序的设计较为复杂, 并且由于线程管理的问题, 可能对某些问题使用混合编程模型并不能得到预期的效果. 但是在一些领域中, MPI 和 OpenMP 结合的混合编程模型的使用已取得成功^[2-3]. 在许多实际问题求解过程中, 往往存在大规模三对角线性方程组的求解^[4], 所以研究 MPI 和 OpenMP 结合的并行算法是有一定意义的.

本文首先介绍了 MPI 和 OpenMP 混合编程模型及其实现方法. 之后建立了求解大规模三对角线性方程组的 MPI+OpenMP 多粒度混合并行算法, 并在上海大学高性能计算集群上与纯 MPI 算法进行了性能比较, 最后给出结论.

2 MPI+OpenMP 混合编程模型

2.1 MPI 编程模型

MPI(Message Passing Interface)是一个消息传递接口的标准, 用于开发基于消息传递的并行程序, 其目的是为用户提供一个实际可用的、可移植的、高效的和灵活的消息传递接口库. 其优点有: 具有很好的可移植性; 具有很好的可扩展性; 有完备的异步通信功能等. 然而 MPI 也存在以下不足: 应用的分解、开发和调试相对困难, 而且通常要求对代码做大量的改变. 通信可能造成大的开销, 为了最小化延迟, 通常需要大的代码粒度.

2.2 OpenMP 编程模型

OpenMP 是基于线程的共享编程模型, 一个共享存储的进程由多个线程组成. OpenMP 定义了一个由编译制导运行时库函数和环境变量构成的集合, 用来描述共享内存的并行机制. 由于 OpenMP 中的通信是隐性的, 这使得 OpenMP 的编程相对简单, 且相对容易实现并行. 同时, OpenMP 更好地利用了共享存储的体系结构, 避免了消息传递的开销, 并提供了细粒度和粗粒度并行机制.

2.3 MPI+OpenMP 混合编程模型

根据分布存储高性能计算机的多层次体系结构的特点, 结合 MPI 和 OpenMP 的各自特点, 可以将上述两种编程模型 MPI 和 OpenMP 相结合, 以提供节点内和节点间的两级并行, 最大限度地利用分布存储高性能计算机的体系结构, 实现 MPI+OpenMP 的混合编程模型. 这种混合编程模型的主要优势在于: MPI 可以解决处理器间的粗粒度通

信, 而 OpenMP 提供轻量级线程可以很好地解决每个多处理器计算机内部各处理器间的交互, 同纯 MPI 模型相比, 混合编程模型使用快速的共享内存访问代替节点内的消息传递, 减少了消息传递的数量, 降低了通信开销, 缩短了执行时间. 且当 MPI 进程数增加时, 通信开销增大, 并行性能通常会下降, 而混合模型的通信是在节点间, 而非处理器间, 因此在解决问题处理器不变的情况下, 进程数大大地降低, 因此混合模型相对于纯 MPI 模型具有更好的可扩展性.

由于 OpenMP 提供了粗、细两种粒度的并行机制, 所以在 MPI 与 OpenMP 组成的混合模型中也对应存在这样的两种并行机制: MPI+粗粒度 OpenMP 混合并行、MPI+细粒度 OpenMP 混合并行. 下面我们简单介绍一下这两种并行机制.

2.3.1 MPI+细粒度 OpenMP 混合并行

OpenMP 的细粒度并行是指利用 OpenMP 只并行求解循环部分的计算, 又称为循环级并行, 这种并行方式是最简单的混合编程方法, 在整个混合编程的过程中只需要对 MPI 进程内的循环使用 OpenMP 多线程共享结构并行即可. 具体的运行过程及伪代码可以参考文献^[5-6].

由于细粒度的混合并行存在着循环级并行优化的问题和并行效率的问题, 在实际使用混合编程方法式往往会选择粗粒度的并行, 以更好地发挥高性能计算机的性能, 得到更好的并行效率.

2.3.2 MPI+粗粒度 OpenMP 混合并行

在粗粒度混合并行中, OpenMP 采用 SPMD 编程模式同 MPI 结合. 粗粒度混合并行完全不同于细粒度混合并行, 它是在 MPI 进程生成和初始化后直接创建线程, 通过库函数获取线程启动数量和线程的编号, 在根据不同的线程编号分配不同的操作, 以实现任务和数据的并行. 由于各线程在执行过程中是共享节点内部的内存, 所以可能会需要数据同步, 这就需要在各线程中加入同步点, 或者采用其他方式以达到同步的效果, 此外由于节点间还需要进行通信操作, 这时候通常采用选择某个线程单独执行节间的通信任务, 在通信期间, 其他线程的任务可以继续执行, 这样就充分发挥了分布存储系统的优势, 最大限度的提高了并行效率. 所以 MPI+粗粒度 OpenMP 混合并行在实际的 MPI+OpenMP 混合编程中广泛使用.

3 大规模三对角线性方程组的求解

大规模三对角线性方程组的求解在科学与工程

计算中具有重要地位. 所以本节主要介绍求解超大规模三对角线性方程组的混合并行算法.

求解三对角线性方程组的最快的串行算法法是 LU 分解法, 而分而治之算法是适合并行计算的一种最有效的方法. 本文所采用的并行算法是基于 LU 分解算法的并行分割算法^[7]. 下面给出算法的实现方法和性能, 这里不涉及算法的具体细节.

3.1 多粒度混合并行算法的实现方法

在混合并行模型中, 每个节点执行一个 MPI 进程, 且这个 MPI 进程包含了多个线程, 为了创建一个高效算法, 必须将算法与分布共享存储系统的多层体系结构相结合. 由于分布共享存储系统在节点内部是共享存储的, 而在节点间是通过快速互联网相连接的一个系统, 为此我们构造了一个多级并行算法, 即节点间使用进程级 (MPI) 并行和节点内使用线程级 (OpenMP) 并行.

进程级 (MPI) 并行主要包括五个过程: (1) 各个进程约化, 得到两个缩减方程; (2) 在各个节点间收集和发送约化后的缩减方程; (3) 求解缩减方程组; (4) 将缩减方程组的解发送给各个节点; (5) 各个进程进行回代, 计算原方程组的解. 节点内的线程级并行主要是通过多个线程完成 (1)、(3)、(5) 中的计算.

在 MPI+OpenMP 混合算法的具体实现中, 过程 (1)、(3)、(5) 由节点级算法通过 OpenMP 进行并行化, 在过程 (3) 中, 采用循环级并行, 即这步采用细粒度的 OpenMP 并行, 而在过程 (1) 和 (5) 中, 主要是约化和回代求解的过程, 在约化过程中, 可以将需要约化的方程组按行分割划分, 划分和约化是同时各个线程中进行的, 之后再行整合以得到节点内的两个缩减方程. 在回代求解过程中, 按照约化过程的划分, 各个线程同时进行回代求解, 即这两个过程是粗粒度的 OpenMP 并行, 所以整个的 MPI+OpenMP 混合并行算法具有多粒度. 除此之外, 为了减少通信的延迟时间, 我们采用全局通信函数 MPI_Gather 和 MPI_Scatter 进行数据的收发操作.

3.2 性能测试

我们分别在两台上海大学高性能计算集群 (IBM、Lenove) 上将 MPI+OpenMP 的多级混合并行算法与纯 MPI 算法作了性能比较.

3.2.1 高性能计算集群

上海大学高性能计算集群 (IBM) 由 14 台 IBM HS21 刀片服务器和 2 台 X3650 服务器组成计算和管理节点, 网络连接采用千兆以太网连接各个节点. 每个节点配置 4 GB 内存, CPU 为 Intel Xeon 5150,

2.66 GHz 主频, 系统计算速度为 4 500 亿次每秒以上. 系统软件环境包括 Red Hat 3.4.5-2 操作系统、支持 OpenMP 制导的 Intel Fortran Compiler 10.1 和 C/C++ 编译器和 MPICH2 并行环境.

上海大学高性能计算集群 (Lenove) 由 10 台 Lenove B5102 刀片服务器和两台 R525 服务器组成计算和管理节点, 网络连接也是采用千兆以太网连接各个节点. 每个节点配置 4 GB 内存, 计算节点内 CPU 为双路六核 2.66 GHz 主频, 系统峰值速度为 1 143 Gflops/s. 系统软件环境包括 Red Hat 4.1.2-48 操作系统、支持 OpenMP 制导的 Intel Fortran Compiler 10.0 和 C/C++ 编译器和 MPICH2 并行环境.

混合 MPI+OpenMP 代码需要使用带有 “-openmp” 选项的编译命令编译, 该选项激活、解释程序中的 OpenMP 编译制导.

3.2.2 实验结果

在 IBM 机器上进行测试的三对角线性方程组的系数矩阵为 92252160 阶的 Topelitz 矩阵 $[-1, 2, -1]$, 在 Lenove 机器上测试的系数矩阵为 69189120 阶的 Topelitz 矩阵 $[-1, 2, -1]$, 右端向量取值为 $[1, 0, \dots, 0, 1]$. 由于两个高性能集群的配置不同, 一个节点内是 4 个线程, 一个节点内是 12 个线程, 所以在进行数值实验时分两种情况分别进行说明.

图 1 给出了在 IBM 机器上纯 MPI 算法和 MPI+OpenMP 混合算法在节点个数分别为 1, 2, $\dots$, 15 时的加速比 (每个节点内含 4 个核). 对 MPI 程序的执行, 我们将每个进程分配到核上, 在四个进程的情况下, 仅被分配到一个节点上, 在 60 个进程的情况下, 进程被均匀地分配到 15 个节点中. 对于 MPI+OpenMP 实现的程序, 每个节点仅创建一个 MPI 进程, 每个进程内有四个线程. 这样当 MPI+OpenMP 混合算法在 N 个节点上运行时, 与其作对比的纯 MPI 算法也在这 N 个节点上的 $4 \times N$ 个核上进行并行计算. 图 1 表明, 从 12 个节点后, MPI+OpenMP 并行算法的加速比明显好于纯 MPI 并行算法的加速比, 并且随着节点个数的增加, 二者加速比的差距也加大了.

图 2 给出了在 Lenove 机器上纯 MPI 算法和 MPI+OpenMP 混合算法在节点个数分别为 1, 2, $\dots$, 11 时的加速比 (每个节点含 12 个核). 对 MPI 程序的执行, 我们将每个进程分配到核上, 在 12 个进程的情况下, 仅被分配到一个节点上, 在 132 个进

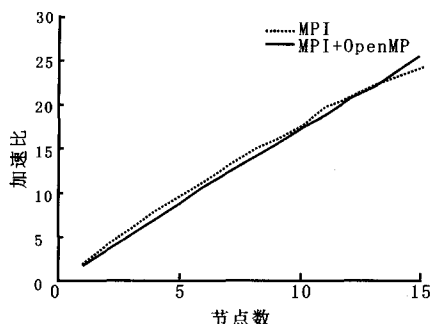


图1 IBM 集群, MPI 和 MPI+OpenMP 加速比比较

程的情况下, 进程被均匀地分配到 11 个节点中. 对于 MPI+OpenMP 实现的程序, 每个节点仅创建一个 MPI 进程, 每个进程内有 12 个线程. 这样当 MPI+OpenMP 混合算法在 N 个节点上运行, 与其作对比的纯 MPI 算法也在这 N 个节点上的 $12 \times N$ 个核上进行并行计算. 图 2 表明, 从 3 个节点开始, MPI+OpenMP 并行算法的加速比明显好于纯 MPI 并行算法的加速比, 并且随着节点个数的增加, 二者加速比的差距也明显加大. 同时随着节点数的增加, MPI 的加速比会下降(主要原因是通信开销过大), 但是 MPI+OpenMP 并行算法的加速比还是不断上升, 所以 MPI+OpenMP 并行算法具有更好的扩展性, 且对比图 1 可知, 当节点内部的核数越多时, 使用 MPI+OpenMP 混合编程模型具有更好的加速比和扩展性.

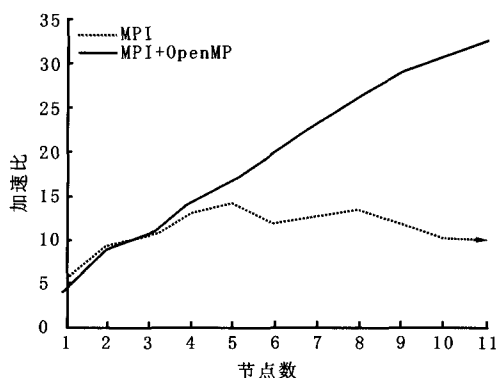


图2 Lenovo 集群, MPI 和 MPI+OpenMP 加速比比较

图 1 中节点数小于 12 和图 2 中节点数小于 3 时, MPI+OpenMP 的加速比 MPI 的加速比小, 主要是由于在节点个数少时, 并行通信开销很小, 同时由于三对角方程组计算的速度较快, 且节点内数据量大容易引起内存总线拥挤等因素导致 MPI+OpenMP 混合并行算法的加速比不如纯 MPI 并行算法的加速比, 但是随着节点数的增加, 通信的开销增加, MPI+OpenMP 的优势就显示出来, 特别是当

OpenMP 的线程数越多时, 其优势越明显, 扩展性越好. 因为 MPI+OpenMP 算法当每个节点内线程越多时, 大量减少了节点间的数据通信开销.

4 结束语

本文分析了混合编程模型的性能和实现方法, 并且使用了多粒度的混合编程模式建立了一个求解大规模三对角线性方程组的并行算法, 结果表明 MPI+OpenMP 的混合编程模型在分布存储系统中能更好地发挥系统体系结构的优势, 而且 MPI+OpenMP 的混合同行算法具有良好的移植性和扩展性, 且随着处理器核数的不断增加, MPI+OpenMP 混合编程模型的扩展性等优势将会更加明显.

参考文献:

- [1] Michael J Quinn. Parallel programming in c with MPI and openMP[M]. New York: McGraw-Hill, 2003.
- [2] Nikolaos Drosionos, Nectarios Koziris. Performance comparison of pure MPI vs hybrid MPI-openMP parallelization models on SMP clusters[C]// Proceedings of IPDPS04. Santa Fe, New Mexico: IEEE, 2004.
- [3] Yan Xiaoyang, Zhang Weiwen. The parallelization of three-dimensional electro-magnetic particle model using both MPI and openMP[C]// Proceedings of IC-CIS2010, Chengdu, China: IEEE, 2010: 17-19.
- [4] Yingchong S, L Lixia. Reducing communication overhead in large eddy simulation of jet engine noise[C]// Proceedings of CLUSTER2010, Crete, Greece: IEEE, 2010: 255-264.
- [5] 潘卫, 陈燎原, 张锦华, 等. 基于 SMP 集群的 MPI+OpenMP 混合编程模型研究[J]. 计算机应用研究, 2009, 12(26): 4592-4594.
- [6] 赵永华, 迟学斌. 基于 SMP 集群的 MPI+OpenMP 混合编程模型及其有效实现[J]. 微电子学与计算机, 2005, 10(22): 7-11.
- [7] 李晓梅, 莫则尧, 胡庆丰, 等. 可扩展并行算法的设计与分析[M]. 北京: 国防工业出版社, 2000.

作者简介:

郑汉垣 男, (1965-), 博士研究生, 副教授. 研究方向为高性能计算方法与应用.

刘智翔 男, (1986-), 博士研究生. 研究方向为高性能计算方法与应用.

封卫兵 男, (1968-), 博士, 副教授, 研究生导师. 研究方向为高性能计算方法与应用.

张武 男, (1957-), 博士, 教授, 博士生导师. 研究方向为高性能计算方法与应用.