大规模结构有限元分析程序在多核集群计算环境中的性能分析和优化

吕 海 邸瑞华 龚 华
(北京工业大学计算机学院 北京 100124)

摘 要 通过对基于MPI编程模型实现的开源有限元计算分析软件在多核集群计算平台中的程序性能的分析,找出程序瓶颈及其原因,实现了基于MPI编程模型的并行程序在多核计算环境中的性能优化。根据程序性能瓶颈的分析,提出了基于MPI/OpenMP混合并行编程模型的大规模线性/非线性方程组求解和多线程多进程同时进行消息通信的两种程序性能优化方案。不同计算规模的实验结果表明,在多核集群计算平台中,MPI/OpenMP混合编程模型实现的大规模非线性方程组求解器相对于单纯基于MPI编程模型实现的并行程序,其性能有2倍到3倍的提升;多线程多进程同时消息传递的优化方案虽然对程序能够起到性能优化作用,但是对解决程序消息通信瓶颈的问题不是最好的方法。两个方案总体性能分析结果表明,基于MPI/OpenMP混合编程模型实现的并行程序,在多核集群计算平台中能够更好地发挥硬件系统的计算能力。

关键词 MPI/OpenMP,OpenSeesSP,多核,非线性方程组求解
中图法分类号 TP302.7 **文献标识码** A

Performance Analysis and Tuning of Large-scale Finite Element Analysis Program on Multi-core Cluster Platform

LV Hai DI Rui-hua GONG Hua
(College of Computer Science, Beijing University of Technology, Beijing 100124, China)

Abstract Through the performance analysis of an open source finite element software based on MPI program model on multi-core cluster platform, some performance bottlenecks were founded. Based on the performance bottleneck analysis, two optimization plans based on MPI/OpenMP hybrid parallel program model were proposed, one of them resolves the inefficiency in solving linear or nonlinear system equations, and the other one elevates processes communication performance. Experiment results show that hybrid parallel solver can efficiently promote the pure MPI based parallel program performance, as up to 3 times. The multi-thread multi-process communication plan can do some optimization, but is not the best solution in this case. The overall optimized performance analysis indicates that on multi-core cluster computing platform, MPI/OpenMP parallel program model can more efficiently utilize hardware system computation resource.

Keywords MPI/OpenMP, OpenSeesSP, Multi-core, Nonlinear equations

1 引言

有限元法[1-3]从创立之初到现在已经经过了几十年的时间,是目前最为流行的数值计算求解方法之一。简单地说,这种方法将问题域中的实体通过由结点联系在一起的有限多个单元来表示,然后利用相应的问题域中的物理或其它约束条件,建立方程再进行求解。

在结构工程中求解大规模结构有限元动力分析问题时,由于有限元模型中单元数目巨大,动力分析计算时间长等特点,使得利用串行程序计算变得不可行。从20世纪70年代初期并行计算机的问世,不少计算数学家及结构分析专家致力于各种并行算法的研究,结构有限元分析算法的研究也迈出了一个新的台阶。目前结构并行分析方法的研究主要集中在区域分解和对系统方程组并行求解两个方面。而几乎所有的有限元分析软件都采用MPI(Message-Passing Interface)并行编程模型[4]来实现并行机制。

随着计算机硬件体系结构的不断发展,多核集群逐渐成为当今高性能计算机的主流架构。在2010年11月的全世界500强排名中,82.8%的超级计算系统使用集群体系结构,并且绝大多数的系统使用多核处理器[5]。MPI作为基于消息传递的并行编程模型事实上的标准,具有可移植性好、功能强大、效率高等多种优点,而且有多种不同免费、高效、实用的实现版本,非常适用于集群这种分布式存储结构的并行计算环境。但是,多核集群是混合式的存储体系结构,具有节点间分

到稿日期:2011-03-01 返修日期:2011-05-24
吕 海(1982-),男,博士生,主要研究方向为高性能计算、并行计算、分布式计算,E-mail:lvhai@emails.bjut.edu.cn;邸瑞华(1947-),女,博士生导师,主要研究方向为网络应用及网络分布计算环境、分布式系统的体系结构、软件工程等;龚 华(1986-),女,硕士生,主要研究方向为并行计算、分布式计算。

布式存储和节点内共享存储的特点。传统的基于 MPI 的程序与多核集群架构的匹配度较差，无法充分发挥硬件体系结构的特点，使得 MPI 程序在多核集群环境下无法达到最佳的计算性能。

在本文结合当前在多核芯片结构的计算平台上进行高性能计算的许多研究成果，对广泛应用于结构工程领域中的一款开源有限元分析软件 OpenSees 的基于 MPI 的并行版本 OpenSeesSP 进行全面性能分析，提出了两种性能优化方案，并基于 MPI/OpenMP 混合并行编程模式[6]实现了这两种并行优化方案。通过对各种不同实验结果的分析表明，我们提出的优化方案有效地提高了 OpenSeesSP 在多核集群计算平台中的性能。

2 背景

2.1 有限元分析计算流程

有限元分析是用较简单的问题代替复杂问题后再求解。它将求解域看成由许多称为有限元的小的互连子域组成，对每一单元假定一个合适的(较简单的)近似解，然后推导求解这个域总的满足条件(如结构的平衡条件)，从而得到问题的解。这个解不是准确解，而是近似解，因为实际问题被较简单的问题所代替。由于大多数实际问题难以得到准确解，而有限元不仅计算精度高，而且能适应各种复杂形状，因而成为行之有效的工程分析手段。

有限元是那些集合在一起能够表示实际连续域的离散单元。有限元的概念早在几个世纪前就已产生并得到了应用，例如用多边形(有限个直线单元)逼近圆来求得圆的周长，但其作为一种方法而被提出，则是最近的事。有限元法最初被称为矩阵近似方法，应用于航空器的结构强度计算，并由于其方便性、实用性和有效性而引起从事力学研究的科学家的浓厚兴趣。经过短短数十年的努力，随着计算机技术的快速发展和普及，有限元方法迅速从结构工程强度分析计算扩展到几乎所有的科学技术领域，成为一种丰富多彩、应用广泛并且实用高效的数值分析方法。

在结构工程领域中，有限元分析方法的流程如图 1 所示。

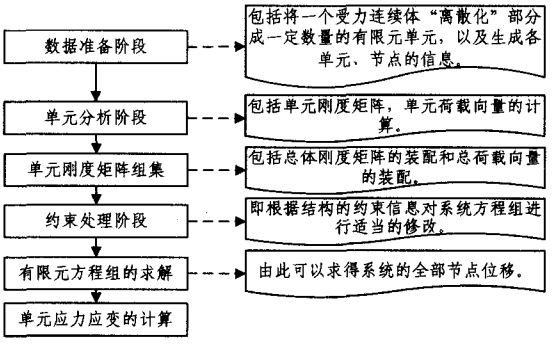

图 1 有限元分析方法流程

2.2 OpenSees 简介

OpenSees 的全称是 Open System for Earthquake Engineering Simulation[7](地震工程模拟的开放体系)。它是由美国国家自然科学基金(NSF)资助、西部大学联盟“太平洋地震工程研究中心”(Pacific Earthquake Engineering Research Center，简称 PEER)主导、加州大学伯克利分校为主研发而成的、用于结构和岩土方面地震反应模拟的一个较为全面且不断发展的开放的程序软件体系。OpenSees 程序自 1999 年正式推出以来，已广泛用于太平洋地震工程研究中心和美国其它一些大学和科研机构的科研项目中，较好地模拟了包括钢筋混凝土结构、桥梁、岩土工程在内众多的实际工程和振动台试验项目，证明其具有较好的非线性数值模拟精度。该软件不断进行升级和提高，加入了许多新的材料和单元，引入了许多业界已成熟的 Fortran 库文件为己所用(如 FEAP、FEDEAS 材料)，更新了高效实用的运算法则和判敛准则，允许多点输入地震波纪录，并不断提高运算中的内存管理水平和计算效率，允许用户在脚本层面上对分析进行更多控制。

OpenSees 由于其开放体系结构设计，使得结构工程领域科研人员能够非常容易地根据自己科研的需要扩展如单元类型、材料、系统方程求解器等等模块。在国内越来越多的科研院所和高校都在使用 OpenSees 作为结构分析、模拟实验等科研工作的工具。

2.3 MPI/OpenMP 混合并行编程模型

在多核计算环境下，使用 MPI/OpenMP 混合编程模型，即在原有的基于 MPI 编写的并行程序中加入 OpenMP 多线程求解，是一种很好的对 MPI 程序进行性能优化的方法[8-15]。由于多核集群体系结构同时具备节点间分布式存储和节点内共享内存的层次结构，支持节点间消息传递和节点内共享内存的多级混合并行编程模型，因此这种混合编程模型与多核集群的体系结构匹配度最高，能充分发挥硬件体系结构的特点，使原有的 MPI 程序获得更好的并行性能。MPI/OpenMP 混合编程模型的设计流程如图 2 所示。

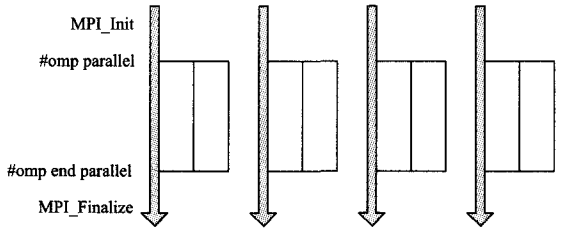

图 2 MPI/OpenMP 混合编程模型

各个节点上运行的进程有主进程和从进程之分，主进程和从进程都可以进行一些局部的计算；之后主进程接收从进程的处理结果，并进行汇总处理；然后主进程可以把汇总结果分发到各个从进程，再由从进程进行局部计算；如此往复，直至程序结束。

2.4 性能分析工具

由美国莱斯大学开发的 HPCTOOLKIT[16]是一套开源的用于分析并展示并行程序性能的软件工具。这套工具支持利用基于硬件性能计数器的 PAPI(Performance counter API)[17]进行并行程序性能数据采样。PAPI 是一组用于采集 CPU 片上硬件性能计数器数据的编程接口，利用这个编程接口可以实时地采集到程序运行时计算机硬件系统所有与性能相关的性能数据。

本文将利用 HPCToolkit 作为程序性能分析的工具。

3 基于 MPI 编程模型的 OpenSeesSP 程序性能分析

OpenSees 对于结构和岩土进行非线性的静/动力有限元分析有着很好的效果。但是随着分析结构规模的扩大和分析问题精度的提高，传统的串行 OpenSees 程序很难快速而有效地解决。在有限元分析中引入并行计算技术，可以增大结构的规模和复杂度，减少问题的求解时间，使 OpenSees 可以用来求解更加大型的结构工程问题。OpenSees 共有两个并行版本[18]，即 OpenSeesSP 和 OpenSeesMP，这两个版本都是基

于消息传递的并行编程模型 MPI 开发的。不同之处在于：

(1)用户不需要对并行计算技术有任何了解就可以像使用串行 OpenSees 一样使用 OpenSeesSP 并行版本，而 OpenSeesMP 需要用户对基于 MPI 的并行有一定的了解来指导并行程序的运行；

(2)用户对原有的输入脚本几乎不用做任何的改变即可将脚本应用于 OpenSeesSP 版本，而 OpenSeesMP 版本需要用户对输入脚本进行适量的修改。

基于 OpenSeesSP 版本对于分析大型结构工程问题的适用性以及用户使用的简便程度，本文选取 OpenSeesSP 版本进行并行优化。图 3 给出了 OpenSeesSP 并行有限元分析的计算流程。

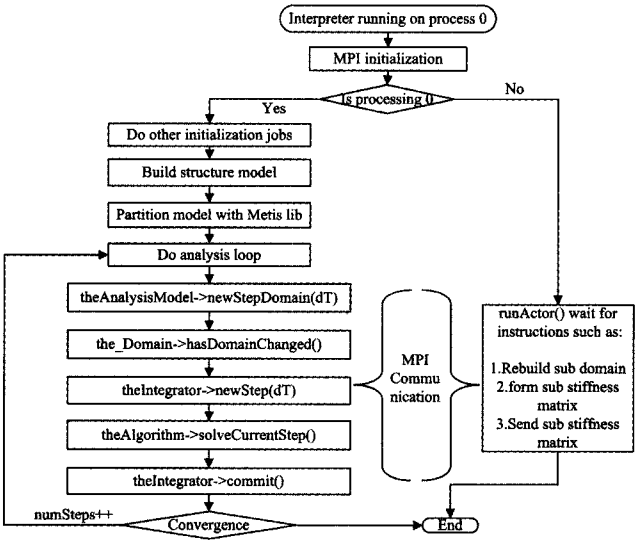

图 3　OpenSeesSP 并行计算流程

问题的规模，即指方程系数矩阵的阶数。影响线性方程组系数矩阵的阶数的因素有很多，如结构模型的规模大小、复杂程度、单元划分粒度、模型的维度、自由度的数量和约束的处理情况等，此处不是本文的重点，不作详细的讨论。

在多核集群计算平台上，使用不同规模的有限元模型，利用 HPCToolKit 并行程序性能分析工具，可以分析出 OpenSeesSP 在整个并行求解过程中的计算时间分配，进而可以找出 OpenSeesSP 潜在的性能瓶颈。图 4 和图 5 给出了 OpenSeesSP 在不同规模有限元模型下的性能分析。

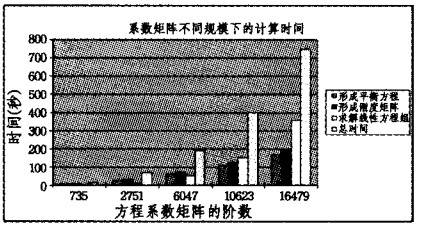

图 4　系数矩阵不同规模下的计算时间

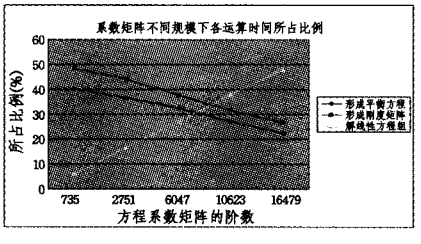

图 5　系数矩阵不同规模下各运算时间所占比例

由图 4 和图 5 可见，问题的规模越大，方程系数矩阵的阶数越高，求解线性方程组所用的计算时间占程序运行的总时间的比重就越大，与之相反的是形成平衡方程和形成刚度矩阵所用的计算时间占程序运行的总时间的比重就越小。因此，在大规模的结构有限元模型分析时，求解方程组成为整个程序的一个性能瓶颈；另外，图 4 和图 5 没有反应出的进程间通信的时间开销，可在实验中通过并行程序性能分析工具抓取到的性能数据得出，在迭代求解系统方程组的过程中，进程间消息通信占据了很大一部分的时间开销，这也成为了程序的另外一个性能瓶颈。在第 4 节中将根据以上性能分析结果，给出针对这个两个瓶颈的程序性能优化方案。

4　优化方案及其实现

针对 OpenSeesSP 程序在多核集群计算平台下出现结构规模限制及硬件等资源利用率低的问题，对程序源代码分析发现主进程因其承担了所有有限元分析中的求解方程组的重担及因 MPI 粗粒度进程无法利用同一 CPU 中的多核，从而易造成主计算机成为整个程序性能的瓶颈点。另外通过一些高性能应用程序性能分析工具辅助进行程序性能分析，以证实该程序的主要计算量集中在求解有限元方程组过程中。此外，进一步发现程序也存在通信开销大的问题，这主要集中在求解有限元方程组之前，由主从进程形成待求解方程组系数矩阵 A(结构总刚度矩阵)和总荷载向量 B 过程中主进程与各从进程间的通信。因此，我们的优化方案主要实现两个方面目标：充分利用多核资源，降低主计算节点计算时间；缩短通信时间。

4.1　方案 1　混合并行系统方程组求解

在原 OpenSeesSP 中，求解线性方程组是通过采用直接解法中的三角分解方法即 UTDU 法[19]实现的。它主要分两个步骤：三角分解和回代。其中三角分解占据求解过程绝大部分的时间，因此并行化三角分解过程可以明显地减少线性方程组的求解时间，也是研究方案要实现的关键部分。下面分析原 OpenSeesSP 中求解线性方程组的解析求解过程。

用有限元法进行结构静/动力分析，最后都归结为求解一个 n 阶线性方程组：

$$\begin{cases} a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n=b_1 \\ a_{12}x_1+a_{22}x_2+\cdots\cdots+a_{2n}x_n=b_2 \\ \cdots\cdots \\ a_{1n}x_1+a_{2n}x_2+\cdots+a_{m}x_n=b_n \end{cases} \tag{1}$$

用矩阵表示为：

$$[A]\{X\}=\{B\} \tag{2}$$

应用三角分解法[14]求解线性方程组的过程分为两步：第一步三角分解，第二步回代。由于有限元线性方程组的系数矩阵 A 就是结构的(有效)刚度矩阵，因此系数矩阵 A 具有正定、对称的特点，从而 A 必然可以分解为如下形式：

$$[A]=[U]^{\mathrm{T}}[D][U] \tag{3}$$

展开上式右边矩阵的乘积并与矩阵 A 的对应元素相等可得：

$$u_{ij}=(a_{ij}-\sum_{k=1}^{i-1}u_{ki}d_ku_{kj})/d_i\begin{pmatrix} j=2,3,\cdots,n \\ i=1,2,\cdots,j-1 \end{pmatrix} \tag{4}$$

$$d_i=a_{ii}-\sum_{k=1}^{i-1}u_{ki}{}^2d_k(j=1,2,\cdots,n) \tag{5}$$

式(3)和式(4)就是求矩阵 U 和矩阵 D 的递推公式。交

替使用这两个公式，即可逐行或逐列地求出 D 和 U 中的各元素。

将式(3)代入式(2)得到：

$$[U]^T[D][U]\{X\}=\{B\} \quad (6)$$

另 $$[U]\{X\}=\{Y\} \quad (7)$$

则 $$[U]^T[D]\{Y\}=\{B\} \quad (8)$$

另 $$[D]\{Y\}=\{Z\} \quad (9)$$

则 $$[U]^T\{Z\}=\{B\} \quad (10)$$

依次求解 Z、Y、X 的值即可求得线性方程组的解 X。

由结构工程理论研究成果可知，结构有限元方程的系数矩阵(即刚度矩阵)具有大型、对称、稀疏、带状分布以及正定、主元占优势等特点。因此，在 OpenSeesSP 程序中的 UTDU 分解的串行算法的核心就是要求出矩阵 D、矩阵 U。在原 OpenSeesSP 中求解矩阵 D、矩阵 U 采取了按列分解的方法。

从计算过程可以看到，在算出 u_{ij} 后，a_{ij} 就再也用不到了，因此可以把原来存放 a_{ij} 的单元改为存放 u_{ij} 之用。同理，在算出 d_i 之后，a_{ii} 就再也用不到了，同时矩阵 U 的对角元素全为1，不必存储，原来存放 a_{ii} 的单元可用来存放对角阵 D 的相应元素 d_i。因此，矩阵 A、U 和 D 可以先后使用同一个数组进行存储。

OpenSeesSP 中 UTDU 列分解的算法描述如图 6 所示。

```
for j=2 to n
    a1j←aj1/a11
        for i=2 to j
        S=0.0
        for k=1 to i-1
            S←S+aki * akk * akj
        end for
        if (i==j)
            ajj←ajj-S
                if |aij|≤τ then 退出
        else
                aij←(aij-S)/aii
    end for
end for
```

图 6　OpenSeesSP 中 UTDU 分解算法

分析 UTDU 求解过程发现，如果按行进行计算则每一行的元素只依赖于首元素及已经求解得到的上一行的元素，因此可以按照行进行并行求解。

这样可以设计基于多线程的并行 UTDU 分解算法。假设在拥有多核处理器的共享存储的并行环境下，有 p 个处理单元(PE)，且矩阵 A 的阶数 n 远远大于 p。把矩阵 A 按列分成 b 块($b=n/p$)，p 个处理单元并行完成第 i 行中每块的 p 列元素 u_{ij} 的计算。例如，对于处理单元 $PE1$ 来说，将完成第 1、$p+1$、$2p+1$、……、$bp+1$ 列的计算。

UTDU 按行分解的并行算法描述如图 7 所示。

至此，我们设计了一个适用于多核及集群计算平台的混合并行有限元系统方程组求解器，并在 OpenSeesSP 中实现。这样针对第 3 节中分析发现的求解系统方程组的程序性能瓶颈，设计了第一个性能优化方案：混合并行系统方程组求解方案。结合 OpenSeesSP 整体并行有限元分析计算过程，我们提出的基于 MPI/OpenMP 混合并行编程模型的求解方案流程如图 8 所示。

```
for i=1 to n
  for k=1 to i-1
        aii←aii=aki^2 * akk
  end for   if |aii|≤ε then 退出
    parallel
        startcol=PEm+i+1(1≤m≤p)
      for j=startcol to n,j+p
          for k=1 to i-1
            aij←aij-aki * akk * akj
            aij←aij/aii
          end for
    end for
  end parallel
end for
```

图 7　基于多线程的并行 UTDU 分解算法

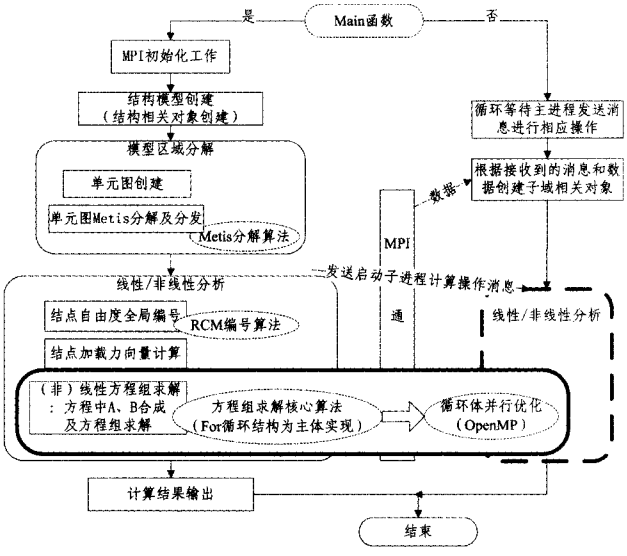

图 8　混合并行系统方程组求解流程

4.2　方案 2　基于 OpenMP 的多线程并行通信优化

在原 OpenSeesSP 中通信开销主要集中在求解有限元方程组之前，由从进程形成待求解方程组系数矩阵 A(结构总刚度矩阵)和总荷载向量 B 的过程中主进程与各从进程间的通信。主进程在进行系统方程求解之前，需要从各个从进程收取子域系数矩阵和子域荷载向量；在一次迭代步中求解完系统方程组后，还需要再将求出的位移等场向量发送回子进程。这个过程是通过轮询的方法依次从各个从进程中收取信息。

在 MPI 标准 2 中已经明确地支持线程对 MPI 通信原语的调用。因此为加快从各个从进程收取、发送中间信息的速度，可以在主进程中利用多线程同时去接收和发送中间信息。

基于多线程并行通信的程序性能优化方案如图 9 所示。

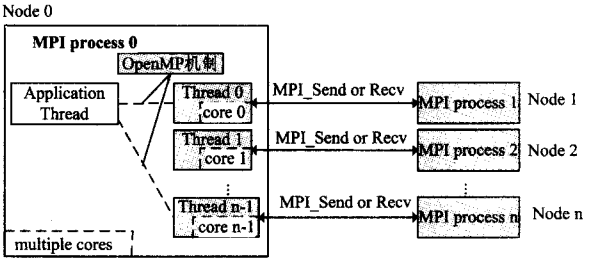

图 9　多线程并行通信的程序性能优化方案

在实现时为保证程序计算的正确性，利用 OpenMP 的原子操作原语对并行收到的数据进行处理。

5 实验及优化后的程序性能分析

本节将对提出的优化 OpenSeesSP 进行整体性能分析和测试进而评价优化效果。性能测试方案包括：

• 对采用两种优化方案优化后的 OpenSeesSP 进行性能分析测试，以确定整体优化效果；

• 对只采用两种优化方案之一优化后的 OpenSeesSP 进行性能分析和测试，以确定两种优化方案对于整体性能提升各自的贡献；

• 对采用两种优化方案优化后的 OpenSeesSP 进行可扩展性分析，主要分析不同线程数目情况下，程序的性能变化。

本文中性能测试分析的实验环境是北京工业大学网格中心的 IBM 高性能 HS21 刀片服务器集群计算平台，实验环境具体硬件性能参数及软件配置如下：

使用 8 个 HS21 刀片服务器组成高性能计算集群；每个刀片服务器有两个 Intel(R) Xeon(R) CPU E5440 Quad core，即 8 个计算核心；每个刀片服务器配置 16G 内存；每个刀片服务器配置 146G 本地硬盘；每个刀片服务器配有两块网卡，分别是 1000M 和 10000M 以太网网卡；8 个刀片服务器通过 1000M/10000M 网络互联。

5.1 测试用例一

这个测试用例将分析采用第 3 节提出的两种性能优化方案优化后的 OpenSeesSP 的整体性能。实验用的结构计算模型相同，通过增加模型中的单元数目来增大计算量，我们使用的是二维结构模型，所以模型单元的数量表示为两个方向单元数目的积。实验测试结果如图 10 和图 11 所示。

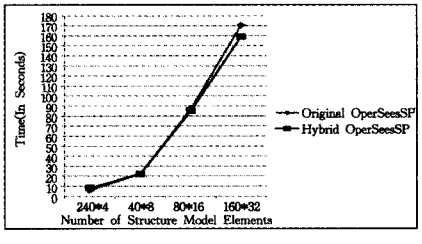

图 10 实验结果一

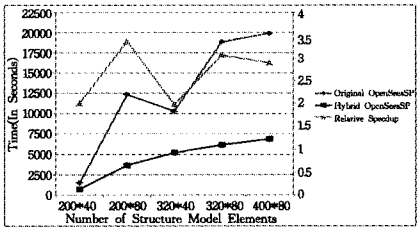

图 11 实验结果二

在这个实验中，每次测试使用不同进程数目，但每次主进程都有 8 个线程同时做方程求解，然后综合几次测试的实验结果，取平均值。

从图 10 可以看出，优化后的 OpenSeesSP 与没有优化的 OpenSeesSP 在性能上差距不是很大，甚至有时性能会比没有优化的 OpenSeesSP 还要差。从图 11 可以看出优化后的 OpenSeesSP 明显比优化前的 OpenSeesSP 性能要好很多，有时优化后的 OpenSeesSP 计算速度是优化前 OpenSeesSP 的 3 倍。在图 11 中同时给出了相对加速比。

这样的结果是由于：对于小规模的结构模型(结构的几何形状相同，但单元数目较少)，由于单元数目较少，在每次迭代求解之前形成的总体刚度矩阵维数较小，这时采用 OpenMP 多线程循环求解平衡方程，8 个线程中有的线程并没有得到实际的计算任务，只是在消耗 CPU cycle。另外每次创建线程等来管理，带来了额外工作负担，使得优化后 OpenSeesSP 有时在性能上比没有优化的 OpenSeesSP 还要差；当模型足够大时(结构的几何形状相同，但单元数目较多)，在每次迭代求解之前形成的总体刚度矩阵维数非常大，这时采用 OpenMP 多线程循环求解平衡方程，8 个线程都能够充分地发挥多核 CPU 的计算能力，使得总体性能得到很大的提升。

5.2 测试用例二

在这个测试用例中，将分析只实现多线程求解平衡方程这个优化方案后的 OpenSeesSP 的性能，实验中使用的结构模型与 5.1 节相同，实验结果如图 12 所示。

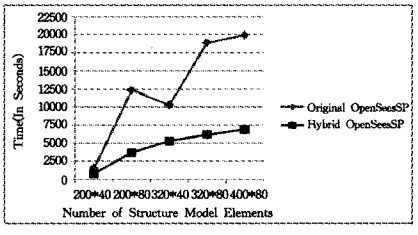

图 12 实验结果三

图 11 与图 12 很相似。实验结果表明，这种利用多线程进行平衡方程求解的 OpenSeesSP 的性能要比没有优化的 OpenSeesSP 好很多，也说明了 MPI 和 OpenMP 混合编程模型能够较为充分地发挥多核集群的计算能力。

5.3 测试用例三

在这个测试用例中，将分析只实现多线程与多进程同时通信这个优化方案后的 OpenSeesSP 的性能，实验中使用的结构模型与5.1节相同，实验结果如图 13 所示。

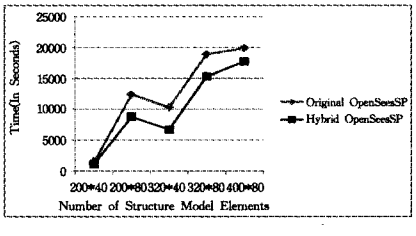

图 13 实验结果四

由图 13 可以看出，优化方案二对于 OpenSeesSP 在大规模的结构有限元计算分析中是优化效果的，但是优化效果与 5.1节中给出的优化方案一相比显的很差。这样的结果是因为：当结构模型中单元数目比较大时，程序的主要性能瓶颈在于主进程中求解平衡方程上。利用多线程与多个子进程同时通信这种优化方案虽然有优化效果，但是对提高程序整体性

能的作用很小。另外，这也说明优化方案二中利用多线程与多进程同时通信的方法，在解决通信密集型应用里由通信造成的性能瓶颈不是一个有效的解决方案。在本次竞赛中由于时间的限制没能对这个问题找到更好的解决方法。

5.4 测试用例四

这个测试用例是分析实现两种优化方案的 OpenSeesSP 程序性能与线程数目的关系。实验中用到的结构模型为 5.2 节实验中单元数目最大的结构模型。实验里对主进程启用不同数目的线程来求解平衡方程。实验结果如图 14 所示。

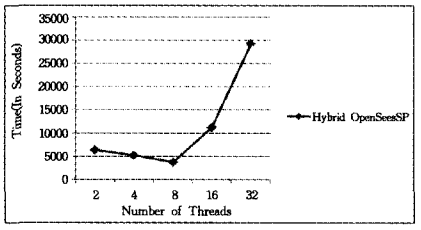

图 14 实验结果五

由图 14 可以看出，在线程数目小于 8 时优化 OpenSeesSP 的计算时间一直在减少，在超过了 8 后，计算时间陡然上升。这是由于本次参加竞赛所使用的计算平台是 8 核的 IBM HS21 刀片服务器，因此在主进程中，参与求解平衡方程的线程数目一旦超过了 8，线程切换等线程管理工作带来的额外负担会影响到整个程序的性能。但是在线程数目小于 8 时，程序的性能近似于线性的提升，这说明优化方案一中给出的多线程求解平衡方程的算法的可扩展性是比较好的。

结束语 我们分析优化了广泛应用于结构工程领域研究中的一款开源有限元分析计算软件 OpenSees。在此过程中结合了当前多核计算平台中高性能计算研究的许多成果，采用了比较成熟的 MPI 和 OpenMP 混合编程模型，我们设计实现了针对 OpenSees 并行版本 OpenSeesSP 的两个优化方案，并行进行了多方面的性能分析测试。

在多核计算平台中，MPI 和 OpenMP 混合编程模型能够很大地提升程序性能。我们设计并实现的混合 MPI、OpenMP 的并行非线性方程组求解器，在多核集群计算平台下有很高的计算效率和很好的可扩展性。

利用多线程与多进程同时通信的方法，在解决通信密集型应用里由通信造成的性能瓶颈不是一个有效的解决方案。OpenSeesSP 这种同时具有计算密集型、通信密集型的应用，由于其整体并行框架采用 MPI，通信瓶颈的解决还是要依赖于 MPI 自身通信的优化。我们设计实现的通信优化方案未能达到预期的优化效果。

参 考 文 献

[1] Bathe K J. Finite Element Procedures[M]. Prentice hall of india New Delhi, 1997

[2] Bathe K-J. Finite Element Procedures[M]. Upper Saddle River, NJ: Prentice Hall, 1996

[3] Fung Y C. Foundations of Solid Mechanics [M]. Englewood Cliffs, NJ: Prentice-Hall, 1965

[4] Adhianto L, Chapman B. Performance modeling of communication and computation in hybrid MPI and OpenMP applications [C]//Proceedings of the 12th International Conference on Parallel and Distributed Systems. IEEE Computer Society, 2006: 3-8

[5] Top 500 Super Computer Sites[OL]. http://www.top500.org/

[6] Bova S W, Breshears C P, Gabb H, et al. Parallel programming with message passing and directives[J]. Computing in Science and Engineering, 2001, 3(5): 22-37

[7] Mazzoni S, McKenna F, Scott M H, et al. OpenSees Command Language Manual[R]. PEER. University of California Berkeley, 2004

[8] 涂碧波，邹铭，詹剑锋，等. 多核处理器机群 Memory 层次化并行计算模型研究[J]. 计算机学报，2008，31(11)：1948-1955

[9] Adhianto L. A New Framework for Analyzing, Modeling and Optimizing MPI and/or OpenMP Applications[D]. Dissertation of the Degree Doctor of Philosophy University of Houston, 2007: 1-23

[10] 胡晓力，田有先. 多粒度并行计算集群研究与应用[J]. 电力学报，2007，22(4)：436-438

[11] 王惠春，朱定局，曹学年，等. 基于 SMP 集群的混合并行编程模型研究[J]. 计算机工程，2009，35(3)：271-273

[12] 单莹，吴建平，王正华. 基于 SMP 集群的多层次并行编程模型与并行优化技术[J]. 计算机应用研究，2006，23(10)：254-260

[13] 陈勇，陈国良，李春生，等. SMP 机群混合编程模型研究[J]. 小型微型计算机系统，2004，15(10)：1763-1767

[14] Benkner S, Sipkova V. Exploiting Distributed-memory and Shared-memory Parallelism on Clusters of SMPs with Data Parallel Programs[J]. International Journal of Parallel Programming, 2003, 31(1): 3-19

[15] 胡晨骏，王晓蔚. 基于多核集群系统的并行编程模型的研究[J]. 计算机技术与发展，2008，18(4)：70-73

[16] Adhianto L, Banerjee S, Fagan M, et al. HPCTOOLKIT: tools for performance analysis of optimized parallel programs[J]. Concurrency and Computation: Practice and Experience, 2010, 22: 685-701

[17] Browne S, Dongarra J, Garner N, et al. A Portable Programming Interface for Performance Evaluation on Modern Processors[J]. International Journal of High Performance Computing Applications, 2000, 14(3): 189-204

[18] McKenna F, Fenves G L. Using the OpenSees Interpreter on Parallel Computers[R]. University of California Berkeley, 2008

[19] 张健飞，姜弘道. 对称正定矩阵的并行 LDLT 分解算法实现[J]. 计算机工程与设计，2003，24(10)