

基于工作站机群的有限元结构分析并行计算

付朝江

FU Chao-jiang

福建工程学院 土木工程系, 福州 350007

Department of Civil Engineering, Fujian University of Technology, Fuzhou 350007, China

E-mail: cjfu@163.com

FU Chao-jiang. Parallel computing for finite element structural analysis on workstation cluster. Computer Engineering and Applications, 2008, 44(23): 236-238.

Abstract: Parallel computing has become a promising trends in science and engineering computing. In this paper, parallel finite element method using domain decomposition technique has been adopted to the distributed parallel environment of workstation cluster. The parallel algorithm is presented for solving the conjugate gradient method with element based domain decomposition. Model of dam structural analysis problem has been solved on workstation cluster. The parallel performance is analyzed.

Key words: parallel computing; workstation cluster; finite element; dam

摘要: 并行计算正成为科学和工程计算中的一个新趋势。将采用区域分裂技术的并行有限元方法应用于工作站机群的分布式并行环境。提出了基于单元区域分裂的共轭梯度并行算法。在工作站机群上对坝体结构进行求解, 对其并行性能进行分析。

关键词: 并行计算; 工作站机群; 有限元; 坝体

DOI: 10.3778/j.issn.1002-8331.2008.23.072 **文章编号:** 1002-8331(2008)23-0236-03 **文献标识码:** A **中图分类号:** TP31

1 引言

随着科学研究和工程技术领域的不断开拓和发展, 出现了各种各样的大型和超大型的复杂结构。如航天器的设计、大型水利设施建筑等, 这类结构不但是具有很大自由度的组合结构, 还包含非线性本构关系、随机载荷和复杂的边界条件等多种因素。对这些结构的分析必须借助高阶的数值分析模型和大规模数学计算, 以保证数值解的精度。而传统的串行有限元分析方法受求解此类问题所需的计算时间长和内存大的限制, 难以实现这类结构的计算, 使其应用受到限制。并行计算的出现对这类结构进行分析计算成为现实^[1]。

以往的各种并行算法基本上是在巨型机和向量并行机上实现^[2-3], 由于并行机的体系结构、内存结构相差很大, 因此在不同的并行机上, 有限元算法也会有很大的不同^[4]。现在, 随着工作站内存和计算速度的增加以及高速信息网络的出现及其突飞猛进的发展, 通过网络相连的工作站机群作为并行计算资源正在成为并行计算的一个主要趋势。使得网络并行计算成了当前国际上计算机科学与应用领域中的一个研究热点^[5]。

有限元法是结构分析中最广泛使用的方法之一, 有限元法的“化整为零、积零为整”的基本思想与并行处理技术的基本原则“分而治之”基本一致, 因此有限元法存在高度的内在并行性, 这为有限元算法的并行研究提供了依据^[6]。有限元分析的各种并行算法一直在研究。这些算法中, 粗粒度的区域分裂法很适应分布式环境。按这种方法, 将求解问题的整个区域划分为

几个子区域, 每个子区域分配给各个处理器^[7]。相应子区域中的计算在各处理器中进行。

2 有限元并行算法

有限元结构分析的基本过程可以分为以下几个步骤: (1) 数据准备阶段, 包括将一个受力连续体“离散化”剖分成一定数量的有限元单元, 以及生成各单元、节点的信息; (2) 单元分析阶段, 包括单元刚度矩阵、单元荷载向量的计算; (3) 单元刚度矩阵组集, 包括总体刚度矩阵的装配和总荷载向量的装配; (4) 约束处理阶段, 即根据结构的约束信息对系统方程组进行适当修改; (5) 有限元方程组的求解, 由此可以求得系统的全部节点位移; (6) 单元应力应变的计算。

其中步骤(2)、(3)、(5)是求解过程中计算量大的部分, 是有限元结构分析的关键部分, 占绝大部分时间, 因而成为有限元并行计算的主要部分。按照区域分裂“分而治之”的基本思想, 把一个问题的计算区域分解为几个小的计算区域, 分别映射在不同的计算机上, 同时并发地进行有限元分析。通常, 有限元法采用两类并行^[8]: (1) 单元级并行; (2) 节点级并行。将单元或节点的计算数据进行划分并分配到并行计算平台的每个处理器。这可通过基于单元的区域分裂或基于节点的区域分裂来实现。在有限元数值方法中, 大部分计算是在单元级。为此, 本文采用基于单元的区域分裂进行并行计算。基于单元的区域分裂如图1所示。

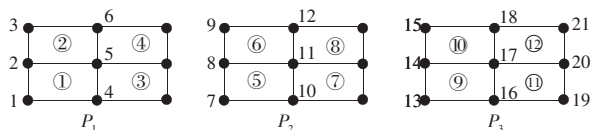


图1 有限元网格划分成三个子区域

2.1 基于区域分裂的有限元法

为在工作站分布式环境下实现有限元法,采用结合共轭梯度算法的区域分裂法^[9]。基于区域分裂的有限元法可概述如下。

用有限元法进行静力分析得到的刚度矩阵方程可表示为:

$$Ku=f \quad (1)$$

式中 K 为整体刚度矩阵, u 为线性方程系统的未知量位移向量, f 为整体力向量。式(1)重新整理并分块表示为:

$$\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \end{Bmatrix} \quad (2)$$

式中 u_1 表示子区域内部未知量位移向量, u_2 为子区域边界未知量位移向量, f_1 、 f_2 为相应自由度的力向量。整个区域划分为 n 子区域后,第 k 个子区域组集的刚度矩阵方程为:

$$\begin{bmatrix} K_{11}^{(k)} & K_{12}^{(k)} \\ K_{21}^{(k)} & K_{22}^{(k)} \end{bmatrix} \begin{Bmatrix} u_1^{(k)} \\ u_2^{(k)} \end{Bmatrix} = \begin{Bmatrix} f_1^{(k)} \\ f_2^{(k)} \end{Bmatrix} \quad (3)$$

式中 $u_1^{(k)}$ 和 $u_2^{(k)}$ 分别为第 k 个子区域的 u_1 和 u_2 的分量。 $K_{11}^{(k)}$ 、 $K_{12}^{(k)}$ 、 $K_{21}^{(k)}$ 、 $K_{22}^{(k)}$ 和 $f_1^{(k)}$ 在每个子区域中计算, $f_1^{(k)}$ 为子区域的 f_1 分量。由于相邻子区域间的相互作用力是未知, $f_2^{(k)}$ 是满足以下关系的未知向量:

$$\sum_{k=1}^n B^{(k)} f_2^{(k)} = f_2 \quad (4)$$

式中 n 为子区域总数, $B^{(k)}$ 为 Boolean 矩阵,它按相应的 f_2 分量的顺序重新排列 $f_2^{(k)}$ 的每个分量。由于对第 k 个子区域 $u_2^{(k)}$ 和 $f_2^{(k)}$ 为未知,式(3)不能直接求解。因而,为得到所求的解, $u_2^{(k)}$ 的各分量的值应满足以下方程:

$$F(u_2) = F(u_2^{(1)}, u_2^{(2)}, \dots, u_2^{(n)}) = f_2 \quad (5)$$

并且 $F(u_2)$ 可表示为:

$$F(u_2) = \sum_{k=1}^n B^{(k)} (K_{21}^{(k)} u_1^{(k)} + K_{22}^{(k)} u_2^{(k)}) \quad (6)$$

式中 $u_1^{(k)}$ 可求解下面的方程得到:

$$K_{11}^{(k)} u_1^{(k)} = f_1^{(k)} - K_{12}^{(k)} u_2^{(k)} \quad (7)$$

将 $u_1^{(k)} = K_{11}^{(k)-1} (f_1^{(k)} - K_{12}^{(k)} u_2^{(k)})$ 代入式(6),采用共轭梯度法并行求解式(5)得到 u_2 。一旦得到 u_2 , u_1 可在每个区域中由式(7)进行计算得到。

2.2 共轭梯度法的并行实现

有限元方程组的求解是有限元并行计算的核心,为了求解系统方程,可采用两种方法,一种方法是组集整个结构的系统方程,然后进行并行求解;另一种方法是采用不需组集整个结构系统方程的预处理共轭梯度法(PCG)求解。由于有限元结构分析单元数量巨大,并行组集有限元系统方程需要进行大量的通信,因而有限元系统方程的组集成为并行求解的一大障碍,且并行直接求解系统方程的效果也不太好。本文采用非重叠型

区域分解算法,不组集整个结构系统方程,采用并行迭代 PCG 求解。

预处理共轭梯度法是求解线性方程组的一种迭代算法^[10]。该方法主要是由矩阵-向量和向量-向量相乘,因而其并行化易实现。在并行平台上,本文提出一个基于单元区域分裂的 PCG 算法。该算法是将单元接单元的共轭梯度算法扩展到子区域级,即子结构接子结构的 PCG 算法。预处理共轭梯度算法的运算步骤如下。上标(s)表示处理器号。

$$(1) r_0^{(s)} = f^{(s)} - K^{(s)} u_0^{(s)}; p_0^{(s)} = L^{-1} r_0^{(s)}; \gamma_0 = \sum (p_0^{(s)T} \cdot p_0^{(s)}); L \text{ 为条件矩阵}$$

$$(2) \text{loop for } j=1, \dots, \text{until convergence}$$

$$(3) w_j^{(s)} = K^{(s)} p_{j-1}^{(s)}$$

$$(4) \rho_j^{(s)} = (w_j^{(s)}, p_{j-1}^{(s)})$$

$$(5) \sigma_j^{(s)} = \rho_j^{(s)} / \gamma_{j-1}$$

$$(6) \text{Barrier()}; \text{all processors complete to } (5)$$

$$(7) 1/\alpha_j = \sum_{s=1}^{np} \sigma_j^{(s)} \quad s=1, 2, \dots, np$$

$$(8) u_j^{(s)} = u_{j-1}^{(s)} + \alpha_j p_{j-1}^{(s)}$$

$$(9) r_j^{(s)} = r_{j-1}^{(s)} - \alpha_j w_j^{(s)}$$

$$(10) h_j^{(s)} = L^{-1} r_j^{(s)}$$

$$(11) \text{Barrier()}; \text{all processors complete } (10)$$

$$(12) z_j^{(s)} = h_j^{(s)} + \sum_{k \in \text{adj to } s} h_j^{(k)}$$

$$(13) \gamma_j^{(s)} = (z_j^{(s)}, h_j^{(s)})$$

$$(14) \text{Barrier()}; \text{all processors complete } (13)$$

$$(15) \gamma_j = \sum_{s=1}^{np} \gamma_j^{(s)}; s=1, 2, \dots, np$$

$$(16) \beta_j = \gamma_j / \gamma_{j-1}$$

$$(17) \hat{z}_j^{(s)} = L^{-T} z_j^{(s)}$$

$$(18) p_j^{(s)} = \hat{z}_j^{(s)} + \beta_j p_{j-1}^{(s)}$$

$$(19) \text{End for}$$

步骤(8)、(9)、(18)为 $\alpha x + y$ (SAXPY) 运算。 α 为标量, x 和 y 为向量。这些运算可不考虑预处理子和系数矩阵的类型顺序实现。由于所有向量 $u^{(s)}$ 、 $p^{(s)}$ 、 $r^{(s)}$ 、 $w^{(s)}$ 和 $z^{(s)}$ 为子区域的量,因而并行计算时不需要进行通信。步骤(4)、(13)为向量内积运算,其结果得到一个标量。因此并行实现要求各处理器进行向量内积运算(步骤(4)、(13))并收集所有这些标量(步骤(7)、(15))。步骤(3)的矩阵-向量运算无需通信就能实现,由于所有这些量 $w^{(s)}$ 、 $p^{(s)}$ 、 $K^{(s)}$ 在相应处理器中。处理器之间的通信是通过 MPI 实现。这些通信可用 MPI_SEND/MPI_RECV 语句实现。由于向量内积在各处理器中进行并且其结果所有的处理器都需要,因而可通过 MPI_Allreduce 运算来实现这一需要。

3 并行平台

坝体结构分析的数值计算是在 DELL 工作站机群上进行。该机群由 4 台双 CPU 的 DELL 工作站通过 100 MB/s 以太网连接而成,共有 8 个 CPU (2.4 GHz Xeon chips, 512 KB cache), 每个节点内存 1.0 GB。每台工作站都有真实的 IP 地址,使用消息传递接口(MPI)^[11]。操作系统为 Red Hat Linux 9.0。

4 结果及分析

为测试有限元分布式并行计算的效率,对坝体及其地基的二维平面应变模型进行分析计算。整个区域划分为与处理器数相等的子区域,每个子区域有大约相等的单元数。每个子区域映射到各相应的处理器。各处理器得到子区域单元数据后,相应子区域中的计算在各处理器中进行。计算模型如图 2 所示。为验证本文方法的有效性和可靠性,对该模型进行了 3 种网格划分,即网格 A 包含 28 057 个 8 节点二次单元,170 240 个自由度;网格 B 包含 41 098 个 8 节点二次单元,248 922 个自由度;网格 C 包含 273 827 个 8 节点二次单元,1 647 436 个自由度。

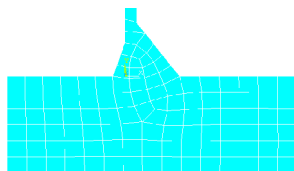


图 2 坝体及其地基的计算模型

为评价并行性能,分别用 1、2、4、6、8 个处理器对所求问题进行求解。用如下性能指标对结果进行了分析。

并行加速比是并行计算常用的性能指标。其定义为:

$$sp = \frac{\text{用 1 个处理器的运行时间}}{\text{用 } n \text{ 个处理器的运行时间}}$$

然而,由于区域分裂算法的计算特性,其总 CPU 时间或总计算量受局部刚度矩阵的带宽影响很大,总 CPU 时间随着子区域数的增加而发生变化。因而,用这个性能指标会得到较差的性能结果,如图 3 所示。用有效加速比来反映其并行性能。其定义如下,其结果如图 4 所示。

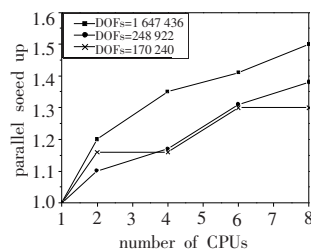


图 3 并行加速比

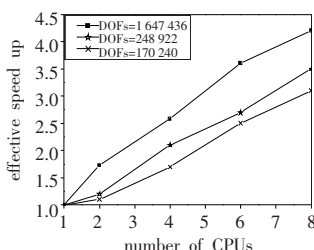


图 4 有效加速比

$$sp_{eff} = \frac{\text{用 } n \text{ 个处理器的总 CPU 时间}}{\text{用 } n \text{ 个处理器的运行时间}}$$

如果一个处理器所用的运行时间等于 n 个处理器的总 CPU 时间,则该加速比与并行加速比相同。CPU 效率可定义为:

$$\eta_{CPU} = \frac{\text{用 } n \text{ 个处理器的总 CPU 时间}}{\text{用 } n \text{ 个处理器的总运行时间}}$$

图 3 和图 4 分别显示了并行加速比和有效加速比,表 1 列出了各种网格时的 CPU 效率。从中可发现,处理器相同时,随着自由度增加,问题的计算粒度增加,因而计算性能变好。然而,自由度相同时,随着处理器数的增加,计算性能变差(CPU 效率降低)。这是由于出现大部分数据通信和同步的 PCG 迭代部分增加;和问题的粒度减小而造成,因而通信开销增加,计算效率降低。这说明通信是并行计算的瓶颈,是影响性能的主要因素。

表 1 各处理器的 CPU 效率 (%)

CPU 数	1	2	4	6	8
网格 A	100	55.0	45.0	41.6	38.7
网格 B	100	60.0	52.5	45.0	43.7
网格 C	100	86.5	64.5	60.0	52.5

为获得好的并行性能,需减小子区域边界节点数和保持处理器之间良好的负载平衡。但 PCG 迭代步中的负载平衡不能完全仅仅通过单元划分的负载平衡实现,由于在这些步中的运算是关于边界自由度(DOFs)的矩阵或向量划分,这是性能降低的主要原因。矩阵或向量划分的负载平衡是可能的,但这样破坏数据的局部性和通信量的增加。由于采用较慢通信速度的网络工作站作为并行计算资源,在共轭梯度迭代步中选择减小通信而不是好的向量划分的负载平衡。所以,在网络机群环境下,为获得良好的并行性能需在算法设计中尽量减小通信。

5 结论

本文实现了采用区域分裂法的有限元结构分析并行计算,并行性能在工作站机群上进行了测试。提出了基于单元区域分裂的共轭梯度并行算法,并作为并行求解器。矩阵-向量计算是该迭代求解器的主要计算部分。该算法对矩阵-向量计算不需要共享变量,计算在各处理器中进行,因而可获得较高的性能。通过并行测试,有限元分布式并行计算算法的有效性得到验证,影响分布式并行计算性能的重要因素进行了分析。

本文采用的区域分解并行策略是一种粗粒度并行算法,即将有限元计算区域划分为与处理器数相等的子区域,然后将这些子区域的计算映射到每个处理器上并行计算。每个处理器负责本子区域的有限元全部计算,只在区域公共部分进行通讯和交换数据信息,数据通讯量少。这种策略的优点在于高度并行、通信少。适合于分布内存工作站机群系统。

参考文献:

- [1] Almasi G S, Gottlieb A. Highly parallel computing [M]. Redwood City, CA: The Benjamin/Cummings Publishing Company Inc, 1990.
- [2] Farhat C, Wilson E. A new finite element concurrent computer program architecture [J]. International Journal for Numerical Methods in Engineering, 1987, 24: 1771-1792.
- [3] Carter W T, Sham T L, Law K H. A parallel finite element method and its prototype implementation on a hypercube [J]. Computers and Structures, 1989, 31: 921-934.
- [4] 张汝清. 并行计算结构力学 [M]. 重庆: 重庆大学出版社, 1993.
- [5] Hipper G, Tavangarian D. Advanced workstation cluster architectures for parallel computing [J]. Journal of Systems Architecture, 1997, 44 (3/4): 207-226.
- [6] Horie T, Kuramae H. Possibilities of workstation cluster for parallel finite element analysis [J]. Microcomputers in Civil Engineering, 1997, 12: 129-139.
- [7] Yakawa G, Shioya R. Parallel finite elements on a massively parallel computer with domain decomposition [J]. Computing Systems in Engineering, 1993, 4: 495-503.
- [8] Kanapady R, Tamma K K. High performance computing on a symmetric multiprocessor environment for RTM process modeling of large complex structural geometries [J]. Advances in Engineering Software, 1998, 29(3): 399-408.
- [9] Kim S J, Kim J H. Large-scale structural analysis using domain decomposition method on distributed parallel computing environment [C] // High-performance Computing on the Information Superhighway, HPC-Asia, Proceedings, 1997.
- [10] Hestenes M R, Stiefel E. Methods of conjugate gradients for solving linear systems [J]. Journal of Research of the National Bureau of Standards, 1992, 49: 409-436.
- [11] 付朝江. 集群 MPI 环境下有限元结构分析并行计算研究 [D]. 上海: 上海大学, 2006.