有限元并行程序设计与实现*

余天堂　姜弘道

(河海大学土木工程学院，南京 210098)

Design and Implementation of FEM Parallel Program

Yu Tian-tang　　Jiang Hong-dao

(*College of civil Engineering, Hohai Univ., Nanjing* 210098, *China*)

Abstract

Parallel computation of FEM under systolic distributed network is a important direction of FEM parallel computation. A program designing method and its implementation for FEM parallel analysis under network based on PVM is presented. Substructure parallel analysis method with multi-front parallel processing is adopted in FEM parallel computation, the interface equations are solved with Preconditioned Conjugate Gradient (PCG) method. The implementation of this designing method is easy. Example shows the designing method can obtain higher speedup ratio.

Keywords parallel program design, network, PVM, multi-front, substructure, speedup ratio

§1. 引　　言

有限元并行计算的一个主要途径是利用子结构方法 [1]，并行对各子结构进行静凝聚，再并行求解界面方程，然后并行回代求内点位移和计算应变、应力. 并行程序的设计与有效实现强烈地依赖于并行机硬件的计算模型. 网络并行计算由于具有巨大的计算潜能、良好的性能价格比和可扩展性，以及灵活的体系结构等优点，和以 PVM, MPI, EXPRESS[2,3] 等为代表的一批基于消息传递的并行程序设计软件平台的出现，使得可伸缩分布式网络并行有限元成了有限元并行计算的一个重要方向. 本文详细介绍了基于 PVM 的分布式网络并行环境下有限元并行分析程序设计方法与具体实现. 有限元并行分析采用多波前并行处理的子结构并行分析方法，界面方程的求解采用预条件共轭梯度法 (PCG). 此种设计方法易于实现. 算例表明，此种设计方法能大大地节省计算时间.

* 1998 年 8 月 12 日收到.

§2. 并行环境分析

并行环境为可伸缩分布式网络，网络计算环境为 PVM (Parallel Virtual Machine). 本文采用主从式 (Master-Slave) 编程模式 [4], 这种计算模式有一个控制程序称为主进程 (Master), 负责进程的生成、初始化、收集并输出计算结果，其余的从进程 (Slave) 执行实际计算，其负载或者是由主进程分配 (有静态分配与动态分配两种), 或者是由进程本身分配. 每台微机上装有一从进程 (Slave), 主进程 (Master) 只需装在其中一台微机上. 一个问题的求解，由主进程和从进程共同完成. 进程的生成、管理及相互间的通信等并行过程通过调用 PVM 库函数实现. 主从式编程模式适用于粗粒度并行计算，因此在设计算法时并行粒度的划分不可太细，通讯与运算尽量重叠.

§3. 子结构并行分析算法

子结构并行分析法，就是并行形成和组集各子结构的劲度矩阵和荷载列阵，再对各子结构内点方程并行地进行静凝聚，然后并行求解界面整体方程，最后并行回代求得各子结构内点位移和进行应力分析. 为了节省内存，子结构内点方程的静凝聚一般采用波前法，即一边组集一边消元.

采用主从式编程模式，子结构并行分析算法的具体过程如下：

MASTER 算法

1. 读入各子结构的数据信息，然后发送给各从进程，向每个从进程只需发送一个子结构的数据信息.

2. 接收从进程发送来的结果，进行后处理并输出.

SLAVE 算法

1. 接收 MASTER 发送来的数据信息，至此每台微机上拥有一个子结构的数据信息.

2. 各从进程并行形成相应子结构的单元劲度矩阵及荷载列阵.

3. 定义子结构界面为波前，用波前法并行进行组集和消元. 最后从组集的劲度矩阵中划去内点对应的行和列和从组集荷载列阵中划去内点对应的行，就得到各子结构边界凝聚后的劲度矩阵和荷载列阵.

4. 采用 PCG 法并行求解界面方程 (只需在从进程间相互进行通讯).

5. 各从进程并行回代求解相应子结构的内点位移.

6. 各从进程并行进行相应子结构的应变、应力分析.

7. 所有从进程将结果发送给主进程.

§4. PCG 法并行求解界面方程

运用 PCG[5] 法求解界面方程可避免界面方程的组集，提高并行度. 设界面方程为

$$[K]\{\delta\} = \{R\}. \tag{1}$$

为了能用 PCG 法并行求解方程 (1)，需要把 PCG 法的各步计算转换成子结构级上的计算. 下面给出一种 PCG 法并行求解方程 (1) 的过程.

上标 s 表示子结构编号，$\sum_{i\in m}$ 表示局部通讯求和，$\sum$ 表示全局通讯求和. $[C]$ 为预条件矩阵，最简单的选取方法是取 $[C]$ 为 $[K]$ 的对角线元素，ε 为允许误差，一般取 $\varepsilon = 10^{-6}$.

1. 给 $\{\delta^{(s)}\}$ 一初值 $\{\delta^{(s)}\}_0$ (一般取 $\{\delta^{(s)}\}_0 = \{0\}$)，计算 $[\overline{C^{(s)}}] = \sum_{j\in m}[C^{(j)}] + [C^{(s)}]$ (m 为与第 s 个子结构相邻的子结构域)，$\{r^{(s)}\}_0 = \{R^{(s)}\} - [K^{(s)}]\{\delta^{(s)}\}_0$，$\{t^{(s)}\}_0 = [\overline{C^{(s)}}]^{-1}\{r^{(s)}\}_0$，$\{s^{(s)}\}_0 = \sum_{j\in m}\{t^{(j)}\}_0 + \{t^{(s)}\}_0$，$\{p^{(s)}\}_0 = \{s^{(s)}\}_0$.

2. 对 $k = 0, 1, 2, \cdots$ 重复执行 $\{h^{(s)}\}_{k+1} = [K^{(s)}]\{p^{(s)}\}_k$，$a_n^{(s)} = \{r^{(s)}\}_k^T\{s^{(s)}\}_k$，$\alpha_d^{(s)} = \{p^{(s)}\}_k^T\{h^{(s)}\}_{k+1}$，$\alpha_n = \sum\alpha_n^{(s)}$，$\alpha_d = \sum\alpha_d^{(s)}$，$\alpha = \frac{\alpha_n}{\alpha_d}$，$\{\delta^{(s)}\}_{k+1} = \{\delta^{(s)}\}_k + \alpha\{p^{(s)}\}_k$，$\Delta = \sum\|\{\delta^{(s)}\}_{k+1} - \{\delta^{(s)}\}_k\|_2$.

若 $\Delta < \varepsilon$，则终止.

$$\{r^{(s)}\}_{k+1} = \{r^{(s)}\}_k - \alpha\{h^{(s)}\}_{k+1}, \quad \{t^{(s)}\}_{k+1} = [\overline{C^{(s)}}]^{-1}\{r^{(s)}\}_{k+1},$$
$$\{s^{(s)}\}_{k+1} = \sum_{j\in m}\{t^{(j)}\}_{k+1} + \{t^{(s)}\}_{k+1}, \quad \gamma_n^{(s)} = -\alpha\{h^{(s)}\}_{k+1}^T\{s^{(s)}\}_{k+1},$$

$$\gamma_n = \sum\gamma_n^{(s)}, \quad \gamma = \frac{\gamma_n}{\alpha_n}, \quad \{p^{(s)}\}_{k+1} = \gamma\{p^{(s)}\}_k + \{s^{(s)}\}_{k+1}.$$

§5. 通讯设计

通过调用 PVM 的通讯函数 pvmfsend 和 pvmfrecv，编制一子程序 broadcast 用来实现 $\sum_{j\in m}$，当需运算 $\sum_{i\in m}$ 时，只需调用 broadcast 子程序. 运算 $\sum$ 进行全局通讯求和的一种比较简单的方法是将每个数据都进行全局广播 (调用 pvmfmcast 函数)，然后每个从进程都接收其数据，进行求和运算.

§6. 并行前后处理

由前面的介绍可知，有限元计算实现了高度并行化，同时也可知有限元并行计算的数据准备是一项极费时的工作. 为了简化数据准备和进一步缩短时间，可并行形成有限元并行分析的数据 (即并行前处理). 由从进程并行读入各子结构的超单元信息，然后各从进程并行剖分各子结构，并行优化各子结构的结点与单元编号，以实现最小半带宽和最小波前宽. 至此各微机上就有了相应子结构的计算数据. 最后的计算结果也可并行进行后处理和

输出，但得到的只是各子结构的结果，为了对整体结构进行后处理，就必须将所有结果发送到一台微机上.

§7. 算 例

例 1. 用本文所介绍的有限元并行程序设计方法，作者用 Fortran 语言编制了平面四结点等参单元有限元并行计算程序 (编译器为 Windows 95 环境下的 Powerstation 4.0). 以 PVM 作为网络并行计算的并行编程环境. 对图 1 所示结构进行弹性分析. 由于目前所使用的网络上只有两台微机 (Pentium-100) 可用. 因此只将结构划分为两个子结构 (如图 1). 图 2 为结自由度 (N) 与计算时间 (T) 关系图. 图 3 为结构自由度 (N) 与加速比 (P) 关系图. $P = T_1/T_2$, T_1 为并行程序在一台微机上的计算时间， T_2 为并行程序在两台微机上的计算时间.

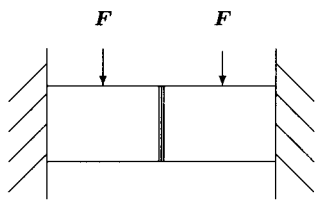

图 1 结构图

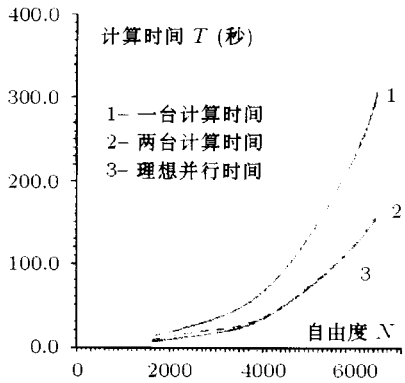

图 2 $N \sim T$ 关系图

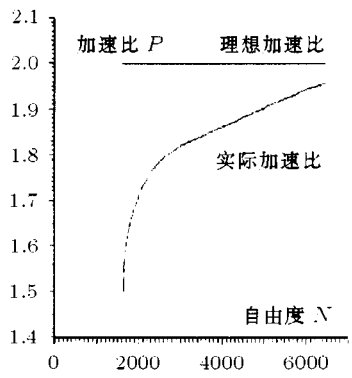

图 3 $N \sim P$ 关系图

本算例中分配到各微机上的任务相等，且整个计算过程实现了高度并行化，因此能得到较高的加速比.

本算例虽然得到了较高的加速比，但整个计算时间显得较长，其主要原因为：

a). 在 Windows 95 下运行程序比在 DOS 环境下运行要费时.

b). 没有优化各个子结构的单元编号.

c). 迭代法收敛的快慢与初值有很大的关系.

d). 由于并行计算的开销比串行计算的开销要大，因此并行程序在一台机器上的运行时间要比采用相同算法的串行程序的运行时间稍微长一些.

例 2. 根据前面的理论，作者编制了空间 20 结点的子结构多波前并行分析程序，用该程序对图 4 所示结构进行弹性应力分析，结构只受自重作用. 将结构划分为两个子结构 (如图 4)，每个子结构的单元编号都进行了优化，使用网上两台微机进行并行计算. 图 5 为结构自由度 N 与计算时间 T 关系图. 图 6 为结构自由度 N 与并行加速比 P 关系图.

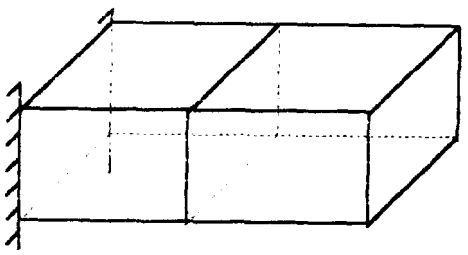

图 4　结构图

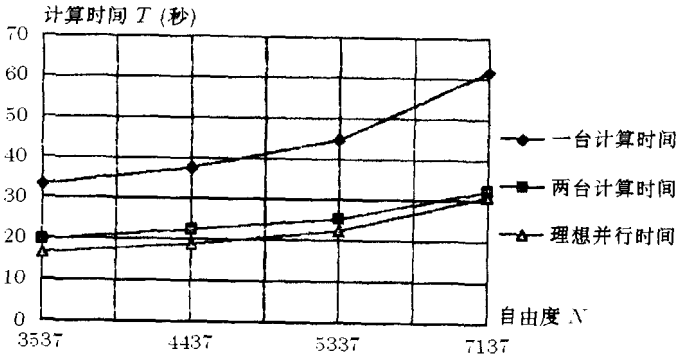

图 5　$N \sim T$ 关系图

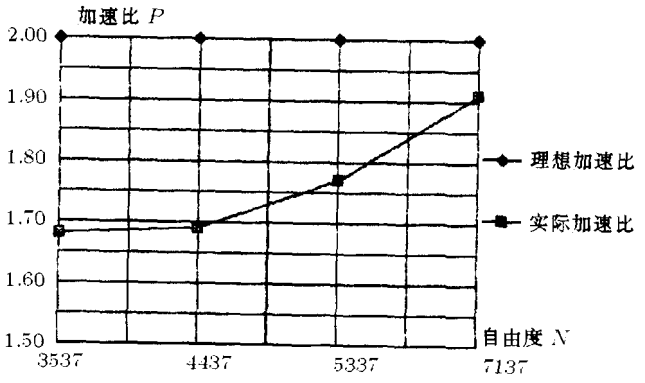

图 6　$N \sim P$ 关系图

本算例中分配在各微机上的任务有点不均衡，第一个子结构有约束，第二个子结构没有约束. 虽然整个计算过程实现了高度并行，但得到的加速比比二维情况下的略低一点.

因此任务的均衡与否直接影响并行计算的效果.

§8. 结 论

本文所给的有限元并行分析程序设计方法具有编程简单、易于对已有的串行程序进行改编，和按此方法设计的并行程序，可移植性强 (因为 PVM 的可移植性强) 及能大大地节省计算时间等优点. 为了充分利用各台机器和提高加速比，应尽量让分配到每台机器上的任务均衡和减少相互间的通讯.

参 考 文 献

[1] 张汝清，并行计算结构力学的发展与展望，力学进展， 24(4), (1994), 511–517.
[2] Bwguelin A, Dongarra J., Recent enhancements to PVm, Int. J. Supercomputer Applications, 9(2), (1995), 108–127.
[3] Dongarra J., Introduction to MPI, Int. J. Supercomputer Applications, 8(3/4), (1994), 169–174
[4] 孙农昶，张林波等，网络并行计算与分布式编程环境，北京，科学出版社， 1997.
[5] 刘万勋等，大型稀疏线性方程组的解法，北京，国防工业出版社， 1981.