doi:10.3772/j.issn.1002-0470.2014.11.005

有限元数值仿真的高性能计算系统的开发①

苗新强②*** 金先龙③*** 丁峻宏***

(*上海交通大学机械系统与振动国家重点实验室 上海 200240)
(**上海交通大学机械与动力工程学院 上海 200240)
(***上海超级计算中心 上海 201203)

摘 要 为实现各专业的高精度复杂性有限元数值仿真,研究了开发可在超级计算机上运行的高性能计算软件的方法。在借鉴和利用国内外并行计算和有限元数值仿真领域的优秀成果的基础上,提出了一种开发有限元数值仿真的高性能计算系统的通用方案。该方案按并行计算的过程将系统划分为前处理、区域分割、数学方程形成、数学方程求解、后处理 5 个模块,在完成各模块的开发后,依据接口规则将其组装在一起构成一个完整的并行计算软件。它通过对现有成熟软件和软件包的二次开发,避免大量繁琐复杂的编程,高效地开发出适合各专业领域的高性能计算程序,有效地缩短软件开发周期,保证软件开发的质量和性能。算例分析表明,用该方案开发的并行计算程序具有良好的可扩展性和并行效率,验证了该方案的正确性和有效性。

关键词 高性能计算,有限元分析,数值仿真,模块化程序设计

0 引 言

有限元分析在土木、船舶、汽车、能源和航空航天等工程领域有着广泛的应用。随着分析对象模型的规模、复杂性和求解精度的提高,有限元分析对计算机硬件性能的要求也越来越高[1],传统计算机内存和浮点运算峰值有限,难以有效克服大规模数值仿真所需内存空间大和求解时间漫长等问题,而超级计算机的出现则为解决这类问题提供了新的切实可行的方法[2]。我国超级计算机硬件技术研究起步较晚但进展很快[3]。1983 年国防科技大学研制的"银河 I 型"亿次巨型机问世,标志着我国具备了研制高端超级计算机的能力。2004 年"曙光 4000A"研制成功,使中国成为继美国、日本之后第三个能研制十万亿次高性能计算机的国家。2013 年中国自主研制的"天河二号"超级计算机以每秒 5.49 亿亿次的浮点运算速度在全球 Top500 排名中位居第一[4],这标志着我国在超级计算机硬件技术的研发上已达到世界领先水平。为了使超级计算机更好地服务于科学研究、工业创新、商业金融、社会公共服务和国家安全等方面,我国目前已在天津、深圳、长沙、济南和广州建立了 5 个运算速度在 1 千万亿次/秒以上的国家级超级计算中心。各高校和科研机构也都纷纷建立了自己的高性能计算分中心。然而,我国高性能计算软件开发严重滞后,这大大限制了超级计算机的应用。在有限元数值仿真领域,国外著名的大型商业化软件如 ANSYS, ABAQUS, LS-DYNA 和 ADINA 等均已发布并行计算版本,但由于种种原因其允许用户操作的最大核数要么受到严格限制,要么收取昂贵的使用费用。此外,这些商业化软件一般只能解决某一类或某几种类型的问题,不能满足一些特定专业和特定需求的有限元数值仿真要求。此时,完全独立自主开发有限元数值仿真高性能计算软件固然很好,但投资成本大,技术难点多,市场风险高,短期内也难以开发出在功能和性能上与国外著名大型商业软件相比的软件。针对这种情况,本文在借鉴和利用国内外并行计算和有限元数值仿真领域优秀成果的基础上,提出了一种开发有限元数值仿真的通用高性能并行计算系统的通用方案,以避免大量繁琐复杂的编程,高效地开发

① 863 计划(2012AA01A307)和国家自然科学基金(11272214, 51475287)资助项目。
② 男,1983 年生,博士生;研究方向:高性能计算,数值仿真;E-mail: newsturdy _ miao@163.com
③ 通讯作者, E-mail: jxlong@sjtu.edu.cn
(收稿日期:2014-05-21)

出适合专业领域的高性能计算程序,从而拓宽超级计算机的应用范围,为大型复杂系统的有限元分析提供有力支持。

1 并行计算软件开发新思路

有限元数值仿真的并行计算过程大致可分为 5 个阶段:划分有限元网格、分割有限元区域、形成数学方程、求解数学方程和输出计算结果。按照传统并行软件开发模式,程序员需要针对这 5 个阶段分别独立编写相应代码才能完成整个系统开发流程。考虑到大部分科研和工程人员只具备一定的专业知识,并不具备深厚的编程功底和良好的并行算法设计能力,因而本文提出了一种新的合理的开发思路,即在借鉴和利用国内外并行计算和有限元数值仿真领域优秀成果的基础上进行二次开发,将有限元数值仿真的高性能计算的 5 个阶段进行模块化设计和组装,以尽量减少或避免大量繁琐复杂的编程工作。

在并行计算和有限元数值仿真领域目前国内外已有很多优秀的研究成果。现将本文涉及到的几个列举如下:用于有限元分析前后处理的 HyperMesh 和 GiD 软件[5,6],分割有限元区域的开源软件包 METIS 及其并行版 ParMETIS[7,8],自动生成串行有限元程序源代码的 FEPG 软件[9],面向高性能科学和工程计算的 Intel MKL 数学库[10]。在这些优秀成熟软件和软件包的基础上进行二次开发,可以高效地开发出满足各个专业领域要求的有限元数值仿真所用高性能计算软件。

如图 1 所示,本文采取的有限元数值仿真高性能计算系统的总体开发流程如下:首先通过 HyperMesh 或 GiD 软件开发前处理模块用于建立有限元模型;然后通过开源软件包 METIS 和 ParMETIS 开发区域分割模块对有限元模型进行剖分;接着根据求解问题的偏微分方程,通过 FEPG 软件生成形成数学方程的串行代码并对其进行并行化处理;再采用 Intel MKL 数学库开发并行求解数学方程的核心算法;最后通过 GiD 软件开发后处理模块,实现输出结果的可视化显示。

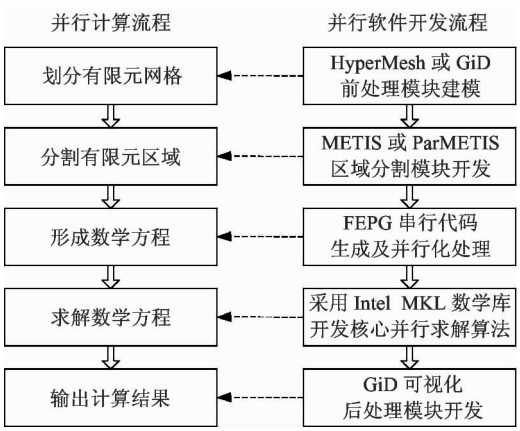

图 1 并行软件开发流程

2 高性能计算系统的开发与集成

本文采用模块化程序设计的概念进行有限元数值仿真的高性能计算系统的开发与集成。首先根据并行计算的过程将系统划分为 5 个功能相对集中、接口单一的模块:前处理模块、区域分割模块、数学方程形成模块、数学方程求解模块和后处理模块。在分别完成各子模块的开发后,把它们按照接口规则组装在一起即可构成一个完整的并行计算软件。

各子模块在开发过程中可直接使用或对现有国内外并行计算和有限元数值仿真领域优秀成熟的软件和软件包进行适当修改。在优秀成熟软件和软件包的基础上进行二次开发不但避免了大量繁琐复杂的编程工作,有效缩短了软件开发周期,而且更容易保证软件开发的质量和性能。

2.1 前处理模块

前处理模块主要完成建立有限元模型的功能。它的输入为分析对象的几何和力学模型信息,输出为该对象的有限元模型信息,包括其单元、节点、材料属性、载荷和边界条件等。

HyperMesh 和 GiD 都具备有限元分析前处理的功能。HyperMesh 是针对有限元主流求解器如 ANSYS, ABAQUS, LS-DYNA 和 ADINA 等的世界领先的高性能前后处理软件[5],它提供了功能强大而齐全的三角形单元、四边形单元、四面体单元和六面体单元等网格划分功能,能够帮助用户快速生成高质量的有限元网格,因此,进行高性能数值仿真的用户可将其作为首选的前处理软件。对大多数用户而言,直接使用 HyperMesh 提供的交互式建模功能就可顺利完成有限元模型的建立,无需再对其进行二次开发。

GiD 是西班牙巴塞罗那数值研究中心开发的有限元前后处理软件[6]。它具有成本低廉、功能齐全和使用简单等特点,最大优点在于有良好的通用性和强大的用户自定义功能。GiD 提供交互式图形用户界面,不仅可以完成复杂几何建模,还可以完成各

种网格剖分以及添加和存储多种相关信息,如材料属性、载荷和边界条件等。对大多数用户而言,直接使用这些基本功能就可简单方便地实现有限元模型的建立,无需再对其做任何修改。有条件和实力的用户可以根据它提供的用户自定制功能和脚本语言 TCL/TK 对其进行二次开发,如定制新的用户界面,通过自定义"问题类型"控制文件使 GiD 将模型输出为用户所需的特定格式等。

2.2 区域分割模块

区域分割模块主要完成有限元模型的分区功能。它的输入为前处理模块输出的有限元模型信息,输出为各个子区域的有限元模型信息及其分区信息。其中,分区信息包括该子区域的所有相邻分区编号以及该子区域的边界节点信息。

METIS[7]是由美国密西根大学开发的用于有限元网格分区、图分区和稀疏矩阵排序的开源串行软件包。它提供了多种有限元网格分区和图分区功能如递归二分分区法、多级 K 路图分区法等。ParMETIS[8]是 METIS 的开源并行版。它扩展了 METIS 的功能,尤其适用于为大规模数值仿真和并行计算提供高质量的分区网格。METIS 和 ParMETIS 在对有限元网格进行分区时,首先要将其转换为图的形式。接着对图进行分区,最后得到单元数目大致相等并且边界节点最少的网格分区结果。当然,在分区的过程中用户也可以根据实际计算需要通过对不均衡系数进行设置来控制各分区单元数目的多少。

以 METIS 和 ParMETIS 程序库为基础进行区域分割模块的开发可以节省大量编程工作。用户只需读入前处理模块输出的有限元模型信息并将其转化为 METIS 或 ParMETIS 特有的数据格式,然后调用程序库里面的无结构图分区函数或网格分区函数即可将大量繁琐复杂的分区工作交给 METIS 或 ParMETIS 自动完成。分区结束后,用户需要编写少量代码将各子区域的有限元模型信息及其分区信息输出为并行计算程序所支持的数据格式。

2.3 数学方程形成模块

数学方程形成模块主要完成各子区域总体刚度矩阵及其总体外部载荷向量的计算和组集。它的输入为区域分割模块输出的各子区域有限元模型和分区信息,输出为各子区域的总体刚度矩阵及其总体外部载荷向量。

有限元自动生成系统 FEPG 是一个开放的有限元程序开发平台[9]。它的设计思想主要是采用元件化的程序设计方法和人工智能技术,根据有限元方法统一的数学原理及其内在规律由计算机自动产生有限元程序。用户只需输入有限元方法所需的各种数学方程表达式和公式,即可自动产生所需的全部有限元程序。它不但免去了用户大量繁琐的有限元编程劳动,而且保证了程序的正确性和统一性。它适合求解各个学科和工程领域的有限元问题,突破了目前商业软件只适用于求解特定领域和特定类型问题的限制,为有限元方法在各个领域的推广和应用创造了前所未有的前景。

在 FEPG 的基础上进行形成数学方程子模块的开发一方面可使开发者根据求解特定领域和特定类型问题生成满足需要的代码,保证了软件开发的灵活性;另一方面,免去了用户大量繁琐复杂的有限元编程劳动,保证了软件开发的高效性和正确性。由于 FEPG 只能生成串行有限元程序源代码,故需要对这些源代码进行适当修改才能为并行计算程序所用。本文基于区域分解法的工作原理将这些串行程序代码修改为并行程序代码。区域分解法的基本思想是把一个完整系统首先剖成若干个子区域,然后将每个子区域分配给一个处理器分别计算,最后将各子区域结果汇总得到问题的完整解。在采用区域分解法进行有限元并行计算时,除了数学方程的求解外,每个子区域执行的操作如刚度矩阵和外部载荷向量的组集(即数学方程的形成)以及应变、应力的计算等,与传统串行有限元程序完全相同。因此,基于区域分解法对 FEPG 生成的串行源代码进行并行化处理时只需去除其中的数学方程串行求解模块并添加必要的并行运行环境信息即可。为提高程序计算效率,在去除原数学方程串行求解模块后需要开发合适的并行求解模块。

2.4 数学方程求解模块

数学方程求解模块主要实现系统数学方程的并行求解功能。它的输入为数学方程形成模块输出的各子区域的总体刚度矩阵及其总体外部载荷向量,输出为求得的各子区域自由度值。

Intel MKL 数学库是面向科学和工程计算的高性能数学库[10]。以 Intel MKL 数学库为基础进行二次开发不但可以节省大量底层的编程工作,而且还可以借助其高度优化和线程安全的特点保证软件开发的质量和效率。本文采用 Intel MKL 数学库基于区域分解法[11]进行数学方程求解子模块核心并行算法的开发。

基于区域分解法进行并行计算时,首先需要按照先内部节点后边界节点的编号原则按式

$$\begin{bmatrix} \boldsymbol{K}_{II} & \boldsymbol{K}_{IB} \\ \boldsymbol{K}_{BI} & \boldsymbol{K}_{BB} \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_I \\ \boldsymbol{x}_B \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{P}_I \\ \boldsymbol{P}_B \end{Bmatrix} \tag{1}$$

同时独立形成各子区域的数学方程。这部分工作已在形成数学方程子模块完成。式中 $\boldsymbol{x}_I, \boldsymbol{x}_B, \boldsymbol{P}_I, \boldsymbol{P}_B$ 分别为内部节点和边界节点对应的位移和外部载荷向量，$\boldsymbol{K}$ 为刚度矩阵。

在数学方程求解子模块需要完成的任务是各子区域的缩聚、边界自由度的求解和内部自由度的回代。下面依次进行介绍：

在式(1)的基础上各子区域同时独立地通过缩聚消去内部自由度后，可得到只含边界自由度未知量的界面方程

$$\widetilde{\boldsymbol{K}} x_B = \widetilde{\boldsymbol{P}} \tag{2}$$

式中缩聚刚度矩阵为

$$\widetilde{\boldsymbol{K}} = \boldsymbol{K}_{BB} - \boldsymbol{K}_{BI}\boldsymbol{K}_{II}^{-1}\boldsymbol{K}_{IB} \tag{3}$$

缩聚载荷向量为

$$\widetilde{\boldsymbol{P}} = \boldsymbol{P}_B - \boldsymbol{K}_{BI}\boldsymbol{K}_{II}^{-1}\boldsymbol{P}_I \tag{4}$$

为节省内存和减少计算量，一般不直接采用矩阵求逆缩聚。本文根据 Han[12] 等提出的改进乔列斯基分解法进行缩聚。

对大规模三维问题系统总体界面方程的组集需要消耗大量的内存资源，往往会超过单个节点机的内存容限而导致求解失败。故界面方程一般不采用直接法求解，而采用能够分布式迭代求解的并行预条件共轭梯度算法求解[13]。采用并行预条件共轭梯度算法求解界面方程的伪代码如图 2 所示。

1. $D=\mathrm{diag}(\widetilde{\boldsymbol{K}})$, $u_0=0$, $r_0=\widetilde{P}$
2. send r_i^b, recv r_i^{eb} from neighbor subdomains
3. compute $\bar{r}_i=r_i+r_i^{eb}$
4. **do** i=0, 1, 2, ...
5. $h_i=\mathrm{D}^{-1}\bar{r}_i$
6. $\alpha_i=\bar{r}_i^T h_i$
7. reduce α_i, $\alpha_i=\Sigma\alpha_i$
8. send h_i^b, recv h_i^{eb} from neighbor subdomains
9. compute $\bar{h}_i=h_i+h_i^{eb}$
10. **if** (i=0) $\bar{q}_i=\bar{h}_i$
11. **else** $\bar{q}_i=\bar{h}_i+(\alpha_i/\alpha_{i-1})\bar{h}_{i-1}$
12. $h_i=K\bar{q}_i$
13. $\beta_i=\bar{q}_i^T h_i$
14. reduce β_i, $\beta_i=\Sigma\beta_i$
15. send h_i^b, recv h_i^{eb} from neighbor subdomains
16. compute $\bar{h}_i=h_i+h_i^{eb}$
17. $u_{i+1}=u_i+(\alpha_i/\beta_i)\bar{q}_i$
18. $\bar{r}_{i+1}=\bar{r}_i-(\alpha_i/\beta_i)\bar{h}_i$
19. **if** ($\alpha_i/\alpha_0<\xi$) exit
20. **end do**

图 2 采用并行预条件共轭梯度算法求解界面方程的伪代码

在上述伪代码中，i 为迭代次数；D 为缩聚刚度矩阵 $\widetilde{\boldsymbol{K}}$ 的对角预条件子；u 为各子区域的边界未知自由度；$\boldsymbol{r}$ 为残差向量；$\boldsymbol{h}$ 和 $\boldsymbol{q}$ 为工作向量；ξ 为特定的误差容限；所有累积向量上面都有横线标记。为了把分布式向量转化为它的累积向量形式，相邻子区域间需要交换边界数据并把接收到的数据进行累积处理。

采用并行预条件共轭梯度算法求得各子区域边界自由度后，其内部自由度可根据下式同时独立回代求解：

$$\boldsymbol{x}_I = \boldsymbol{K}_{II}^{-1}(\boldsymbol{P}_I - \boldsymbol{K}_{IB}\boldsymbol{x}_B) \tag{5}$$

以上就是采用区域分解法并行求解数学方程的基本过程。其中大部分操作，如矩阵与向量的乘积、向量的点积以及向量间的加减操作等，均可直接调用 Intel MKL 数学库函数实现，从而大幅度减少了编程工作量。数学方程求解子模块程序开发完成后，按照模块间的接口规则将其集成到数学方程形成子模块就构成了一个完整的并行计算程序。

2.5 后处理模块

后处理模块主要实现有限元数值仿真计算结果的可视化显示功能。它的输入为数学方程求解子模块输出的各子区域的计算结果，输出为各子区域计算结果的可视化显示。

HyperMesh 目前只针对有限元主流求解器如 ANSYS，ABAQUS，LS-DYNA 和 ADINA 等提供了高性能的后处理功能，而对用户自己开发的有限元数值仿真程序暂无提供相应接口。

GiD 则不同，它针对用户自己开发的有限元数值仿真程序提供了强大的后处理功能。用户只需将计算结果数据按照规定的格式输出，就能借助它的图形界面实现数据的可视化显示。为方便用户使用，也为了使后处理更具有通用性，GiD 官方提供了专门输出后处理文件的函数库 GiDpost。GiDpost 支持 FORTRAN 语言接口和C/C++语言接口，使得用户在自己的计算程序中仅通过增加一个输出函数，就能得到符合要求的后处理文件。

从以上各子模块的开发过程可以看出，在优秀成熟软件和软件包的基础上进行有限元数值仿真高性能计算系统的二次开发可使科研和工程技术人员从大量繁琐复杂的编程中解放出来，高效地开发出适合不同专业领域的高性能计算程序，从而拓宽了超级计算机的应用范围，为大型复杂系统的有限元分析提供了有力工具。

3 算例验证

采用前面介绍的有限元数值仿真所用高性能计算系统的开发方法,本研究基于 FORTRAN 语言且按 MPI 标准开发了结构动力响应并行计算软件,并在上海超级计算中心对其进行了测试。上海超级计算中心的魔方超级计算机由 A、B、C 三个分区组成。其中,A 区主要部署各类工程计算商业软件,B 区和 C 区主要用于各种源代码类计算。本文算例通过了在 C 区的测试运行。C 区由 800 个刀片式计算节点组成,每个节点包括 4 颗 AMD Barcelona 1.9 GHz 四核处理器,64 GB 共享内存。节点机间采用 Infiniband 光纤网络互联,理论带宽 20 Gbit/s。

本研究开发的并行算法评估的动力学计算模型如图 3 所示。该模型为塔楼-裙楼建筑结构,该并行算法完成了它在爆炸冲击载荷作用下的动力响应分析。模型长 44m,宽 31.5m。塔楼高 8 层,裙楼高 3 层,层高为 3m。采用六面体单元进行网格剖分后,该模型具有 4472334 单元,5483900 个节点,16451700 个自由度。动力响应分析总时间 $T=0.4$s,时间步长 $\Delta t=0.0002$s,总计 2000 时间步。

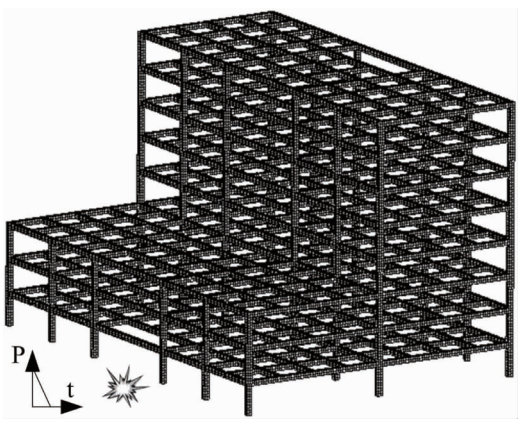

图 3 塔楼-裙楼计算模型

加速比和并行效率是评价并行算法性能的两个重要指标。本文采用分段加速比对并行算法的性能进行评估,其定义为:

针对某一问题,假定分别采用 $m,n,\cdots,z$ 个处理器进行并行计算,相应的耗时分别为 t_m, $t_n,\cdots,t_z$,且 $1<m<n<\cdots<z$ 成立,则 i 个处理器对应的分段加速比为

$$S_i=\frac{t_m}{t_i}\quad(i=m,n,\cdots,z) \tag{6}$$

相应的并行效率为

$$E_i=\frac{S_i}{i/m}\times 100\% \tag{7}$$

采用不同数目的处理器核求解上述塔楼-裙楼动力学计算模型得到的并行计算时间和性能统计如表 1 所示。

由表 1 可见,对给定规模的有限元数值仿真问题,当核数小于 512 时,随着处理器核数的增加并行求解总时间明显减少,加速比明显增大,表明开发的并行计算程序具有良好的可扩展性和并行效率。当核数超过 512 时,并行求解总时间出现了明显增加的现象。这主要是由于随着核数的增多,界面方程的规模急剧增大导致其求解时需要更多的迭代次数才能收敛。因此,为高效利用计算资源采用本文方法进行有限元数值仿真的高性能计算时应将核数限制在合理范围之内。

表 1 并行计算结果

核数	总时间(s)	加速比	并行效率(%)
128	144959	1.00	100
256	76838	1.89	94.33
512	44838	3.23	80.82
1024	77395	1.87	23.41

4 结 论

在借鉴和利用国内外并行计算和有限元数值仿真领域优秀成果的基础上,本文提出一种开发有限元数值仿真所用高性能计算系统的通用方案。它可使科研和工程技术人员从大量繁琐复杂的编程中解放出来,高效地开发出适合各专业领域要求的高性能计算程序,从而为有限元数值仿真的高性能计算在各个领域的推广和应用提供有力的工具。最后实现的系统和算例表明,用该方案开发的并行计算程序具有良好的可扩展性和并行效率,为高性能计算软件开发探索了一条新的途径。

参考文献

[1] Li Y Y, Jin X L, Li L J, et al. A parallel and integrated system for structural dynamic response analysis. *The International Journal of Advanced Manufacturing Technology*, 2006, 30:40-44

[2] 王建炜,金先龙,曹源. 列车与结构动态耦合分析的并行计算方法. 计算力学学报,2012,29(3):352-356

[3] 杨学军. 并行计算六十年. 计算机工程与科学. 2012,

34(8):1-10

[4] Tianhe-2 (MilkyWay-2) - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P.. http://www.top500.org/system/177999, 2014

[5] 黄康,张卫霞. 基于 HyperMesh 的电主轴系统动态性能建模与仿真. 系统仿真学报,2013,25(6):1393-1398

[6] GiD - The personal pre and post processor. http://www.gidhome.com/, 2014

[7] Karypis G, Schloegel K, Kumar V. METIS - serial graph partitioning and fill-reducing matrix ordering. http://glaros.dtc.umn.edu/gkhome/metis/metis/overview, 2014

[8] Karypis G, Schloegel K, Kumar V. ParMETIS - parallel graph partitioning and fill-reducing matrix ordering. http://glaros.dtc.umn.edu/Gkhome/metis/parmetis/overview, 2014

[9] 梁国平,唐菊珍. 有限元分析软件平台 FEPG [J]. 计算机辅助工程,2011,20(3):92-96

[10] Intel Math Kernel Library-Fastest and most used math library for Intel and compatible processors. http://software.Intel.com/en-us/intel-mkl/, 2014

[11] Xicheng W, Baggio P, Schrefler B. A multi-level frontal algorithm for finite element analysis and its implementation on parallel computation. *Engineering Computations*, 1999, 16(4): 405-427

[12] Han T Y, Abel J E. Substructure condensation using modified decomposition. *International Journal for Numerical Methods in Engineering*, 1984, 20(11): 1959-1964

[13] Rao A R M. MPI-based parallel finite element approaches for implicit nonlinear dynamic analysis employing sparse PCG solvers. *Advances in Engineering Software*, 2005, 36:181-198

An approach to development of high performance computing systems for finite element numerical simulation

Miao Xinqiang[*,**], Jin Xianlong[*,**], Ding Junhong[***]

([*] State Key Laboratory of Mechanical System and Vibration, Shanghai Jiaotong University, Shanghai 200240)

([**] School of Mechanical Engineering, Shanghai Jiaotong University, Shanghai 200240)

([***] Shanghai Supercomputer Center, Shanghai 201203)

Abstract

To realize high accurate finite element numerical simulations in different fields, a study on development of high performance computing software capable of running on supercomputers was conducted, and a general scheme for development of high performance computing systems for finite element numerical simulation was proposed based on the reference and utilization of the excellent research results in the areas of parallel computing and finite element numerical simulation in the world. The scheme divides a computing system into five moduls of pre-processing, region-dividing, equation forming, equation solving and post-processing, and then develops them. Furthermore, it combines them together to form a complete parallel computing system according to the interface rule. It releases the broad researchers and engineers from a lot of complex programming work through the further development of the existing mature software and software packages and helps them to efficiently develop high performance computing programs suited for their own professional fields. The correctness and effectiveness of the proposed scheme were verified by an experiment on a real example. The study shows that the scheme is a useful tool for popularization and application of high performance computing for finite element numerical simulations in various fields.

Key words: high performance computing, finite element analysis, numerical simulation, modular program design