CCA环境下构件化线性解法器的设计

周静静 盛鑫芽
(浙江工商大学信电学院 杭州 310018)

摘 要 基于构件的程序设计是解决高性能科学计算软件开发复杂度高、周期长的重要途径。首先介绍了CCA(Common Component Architecture)论坛提出的针对高性能科学计算的构件体系结构。然后介绍了CCA构件程序设计方法,分析了其设计原理和实现思想。并基于CCA环境设计实现了一款偏微分方程线性解法器。对构件化软件与传统并行程序的性能进行了比较,分析了构件化软件的优势。

关键词 CCA,构件技术,线性解法器
中图法分类号 TP31 **文献标识码** A

Design of Linear System Solver on CCA

ZHOU Jing-jing SHENG Xin-ya
(College of Information & Electronic Engineering, Zhejiang Gongshang University, Hangzhou 310018, China)

Abstract Programming based on component is an important way for decreasing complexity and program development time in high performance computing. In this paper we introduced CCA, which is a component architecture towards HPC. Programming in CCA environment, design and implementation of CCA were also discussed. According to design and implement linear system solver, people can realize advantages of HPC software based on component. Result of experiment shows different performance between running MPI application and component application.

Keywords CCA, Component technology, Linear system solver

1 引言

目前,高性能科学计算软件的开发还处于手工组装阶段,开发出的软件还很难做到易修改、易维护和易复用。而科学计算软件开发的一个特点是:某些功能模块的实现具有潜在的复用性。如方程组的迭代求解法等均广泛应用于各种科学计算应用中。而开发者在开发中,基本上都是从头做起,要实现所有功能某块,就增加了开发的复杂度和开发周期。

基于构件的程序设计,正是为了提高科学计算软件的复用性和可维护性,降低开发复杂度,缩短开发周期。本文一方面介绍了基于CCA的构件程序设计,分析了CCA构件体系结构的设计思想和原理;一方面基于CCA环境设计了偏微分方程线性解法器。与传统并行程序设计进行比较,探讨了构件化高性能计算软件的特点和优势。

2 CCA

目前,在商业应用领域已有的构件环境并不适合科学计算领域。面向科学计算的构件技术有着如下一些特定的需求:可以对已存在的大量的科学计算代码较方便地实现构件化,以较好地继承已有的成果;强调程序的性能表现;支持并行计算和分布式计算;支持适合科学计算的数据类型;支持常用的科学计算编程语言;支持多语言实现等。

为了实现面向高性能计算的构件环境,由美国众多国家实验室和大学等机构联合组成了CCA论坛[1],并提出了通用构件体系结构。该构件体系结构相对于业务构件的优势在于其适合高性能科学计算的需求,强调性能因素。目前CCA规范还处于不断发展和完善中。

2.1 CCA概念

CCA是专为高性能计算设计的构件模型规范。其设计目的包括:使现存的科学计算软件更方便地实现构件化,提高科学计算软件的可重用性和互操作性。其内容包括:探索基于构件的设计模式;定义CCA构件的接口表达;定义CCA构件的职能;定义CCA framework的职能等[2]。

其具体概念包括:1)构件(Component)。构件是可以组装成应用程序的具有一定功能的软件单元。它将复杂的功能实现进行封装,而只给出端口供其它构件调用。2)端口(Port)。在CCA规范中,端口是其中的一个SIDL接口。其作用是完成构件之间的相互调用。在面向对象编程语言中,一个端口可以是类或接口(interface);在Fortran等面向过程语言中,一个端口可以是一组子函数或模块(module)。构件

到稿日期:2010-06-22 返修日期:2010-12-09 本文受浙江省教育厅科研项目"可重构的网络测量体系结构及其关键技术的研究"(Y200908196),"ForCES传输映射层(TML)关键技术问题研究"(60903214),基于转发与控制分离思想实现开放可重构路由器的若干基础性技术问题研究"(60970126),863高技术研究与发展计划课题"可重构路由器构件组研制"(2008AA01A323)资助。

周静静(1980—),女,博士,讲师,主要研究方向为计算机网络、网络测量,E-mail:zhoujingjing@zjgsu.edu.cn;**盛鑫芽**(1986—),女,硕士生,主要研究方向为开放路由器。

可以提供端口，实现提供的端口，即实现类或函数。也可以调用端口，即调用端口的方法或函数。3）框架（Framework）。框架负责管理构件的组装和程序的执行，负责连接构件提供和使用的端口，还为所有构件提供了一组标准服务。

2.2 基于 CCA 构件程序设计

图 1 为基于 CCA 环境的编程过程。图中 CCA 工具包括 Framework 和 Babel 语言互操作工具[3]。具体过程为：

（1）编写 SIDL 文件定义端口和构件。

（2）利用 Babel 生成粘合代码和实现文件。Babel 编译器将根据用户要使用的高级语言选项生成相应的服务器端粘合代码。粘合代码的作用是当客户端调用服务器端已编译好的构件时，为服务器端已实现的构件提供中间表达。Babel 还会根据用户选择的高级语言生成相应的实现文件（impl），用户需在实现文件内编制构件实现源代码，其中也可以封装基于 mpi 的并行实现代码。

（3）编译代码生成共享库。

（4）将已生成的构件导入框架搜索路径中，实例化构件，连接构件端口，组合成应用程序并执行。

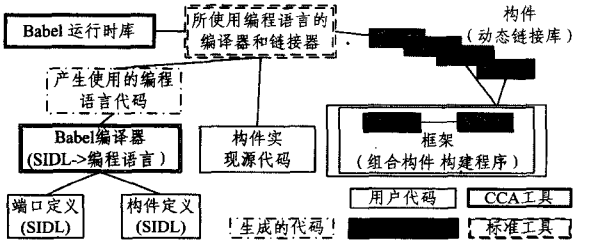

图 1 CCA 环境下编程过程

3 构件体系结构

CCA 规范是用 SIDL 写的一组接口。对框架和构件应完成的功能和服务进行了定义，框架开发者可以通过实现全部或部分接口构建自己的构件框架。

3.1 框架构建程序过程和构件生命周期

程序从源代码经过编译、链接、加载执行的过程与框架使用构件构建执行程序的过程是这样对应的：在基于 CCA 规范下编写构件源代码，源代码中会加入部分与框架交互的代码，利用 CCA 工具生成以共享库形式存在的构件。启动框架开始构建程序时，框架开始读取用户输入的命令或读取包含命令的批处理文件（这些命令包含了编译好的构件的存储路径），实例化构件命令，连接构件命令，执行命令。框架通过读取命令一步步生成日志，当收到执行命令时，框架解析日志文件生成主程序，该程序包含了要连接的共享库的位置以及用户要进行的全部操作，框架执行该程序即可定位能使用的构件的名称和位置，进行链接操作调用共享库，生成所要实例化的构件并运行程序。下面具体介绍框架的执行过程。

编译好的构件是以动态连接库的形式存在的，框架使用构件构建应用程序的过程在逻辑上分为组合时、执行时和销毁时。在组合时，框架进行实例化构件和构件连接操作。实例化构件的过程为：框架通过设置的搜索路径确定可以使用的构件，具体实现方法是框架读取搜索路径中的 XML 文件，通过文件中的 class 元素的 name 属性确定可以使用的构件名。在创建构件实例时，通过 library 元素的 uri 属性所确定的动态连接库的位置创建构件实例对象。

构件连接操作是指通过指定要连接的构件端口，确定构件间的调用关系。由于每个构件类都是继承自某个端口，因此在连接时要指定某个构件实例的调用端口与哪一个具体构件实例被调用端口连接。

组合时框架要执行以下动作：框架实例化 Service 对象，该对象用于构件与框架间的通讯，拥有的方法包括 addProvidesPort()，用于注册构件所提供的端口；registerUsesPort()，用于注册构件所要使用的端口；getPort()，用于从框架获取端口；ReleasePort()，释放端口；getComponentID()，获取端口号等。只有框架拥有 Service 对象，每个构件才拥有一个 Service 对象的引用。每个构件的 Service 对象引用指向框架里的一个 Service 对象。构件 1 可以调用 Service 的 addProvidesPort()方法，从而通过 Servide 对象通知框架该构件所提供的端口。构件 2 可以通过 registerUsesPort()方法通知框架构架该构件所要使用的端口。在组合时，框架将构件 1 的提供端口和构件 2 的使用端口进行连接。框架还将构件 1 所要使用的端口的提供者，即构件 2 的一些相关信息复制到构件 1 的 Service 对象里。在执行时，构件 1 通过调用 Service 的 getPort()方法获取构件 2 所提供的端口的句柄，从而通过端口调用构件 2 中的方法。

销毁时执行端口断开，销毁实例化对象等操作。

4 基于 CCA 的构件化线性解法器

线性解法器主要是针对方程组 $AX=b$ 进行求解。在实际科学计算求解中，该方程一般是由偏微分方程经过有限差分等方法生成的差分方程，再经过线性处理得到的。

4.1 整体设计

对方程组的求解分为直接法和迭代法，迭代法又分为定常迭代法（古典迭代法）和变常迭代法。解法器首先要读入系数矩阵 A、未知向量 x 初始值和右端向量 b。解法器构件设计实现了迭代求解法解法器，定常求解法中设计了 Jacobi，Gauss_seidel，Sor 方法。变常求解法设计了共轭梯度法，设计了 Jacobi，Sor 预条件子、文件和屏幕命令行两种输入输出构件。整体设计如图 2 所示。

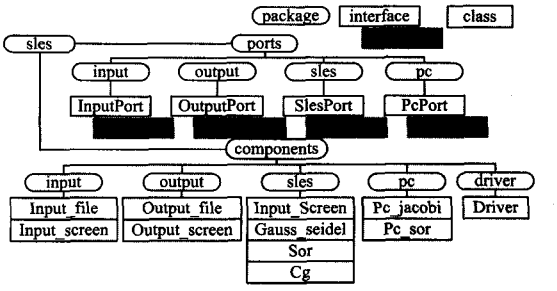

图 2 解法器整体设计图

4.2 构件生成

编写构件的实现步骤为：

1. 编写 sidl 文件，声明端口。该文件负责向 Babel 传送构件的端口信息。端口必须继承 CCA 规范的 gov. cca. Port。

2. 编写 sidl 文件，声明构件。该文件负责向 Babel 传送构件实现信息。构件需实现 CCA 规范的 setServices 方法，从而通过框架的 Service 对象引用完成构件与框架的通讯。

3. 编制 Makefile，利用 Babel 编译器处理 sidl 文件，生成服务器端粘合代码。

（下转第 163 页）

意一个节点出错或任意两个节点顺序颠倒，都会导致恢复数据不正确，使长期部署的 CDP 系统失效。通过添加少量快照可提高 TD 链的可靠性[3]，其邻近时间点恢复算法也需随之做相应调整，这一点有待进一步研究。

参 考 文 献

[1] Yang Q, Xiao W, Ren J. TRAP-Array: A disk array architecture providing timely recovery to any point-in-time[C]// Proc of the Int Symp on Computer Architecture. New York: ACM, 2006

[2] 王彦龙，李战怀，徐娟. 基于块的数据库系统连续数据保护[J]. 计算机研究与发展，2006，43(Suppl.)：168-173

[3] 李旭，谢长生，等. 一种改进的块级连续数据保护机制[J]. 计算机研究与发展，2009，46(5)：762-769

[4] 喻强. 基于 iSCSI 连续数据保护系统的研究和实现[D]. 北京：清华大学，2008

[5] Keeton K, Santos C, Beyer D, et al. Designing for disasters[C]// Proc. of 3rd Conference on File and Storage Technologies. San Francisco, CA, 2004

[6] 魏冰璐. 一种持续数据保护系统的设计与实现[D]. 上海：复旦大学，2008

[7] Walter O. Programming the Microsoft Windows Driver Model [M]. USA: Microsoft Press, 2000

[8] 王彦龙，李战怀，郑然. 多平台数据容灾系统的研究与设计[J]. 计算机应用研究，2007，24(2)：215-218

(上接第 128 页)

4. 在由 Babel 生成的 impl 文件中添加端口的具体实现代码。可以在该文件中包含头文件以链接库文件，比如可以通过包含 mpi. h 头文件来调用 mpi 通信函数，与其他构件通信。其中，自定义的方法中可以封装 mpi 代码，但代码中不可含有 mpi 初始化、结束化语句，使框架进行初始化、结束化工作。SetServices 为与框架交互部分，应给出提供和使用的端口信息。在 sles_Jacobi_Impl. cc 文件中实现 setServices 方法。

4.3 运行构件程序

安装好 ccaffeine 后，可以在 shell 下使用 ccafe-single 或 ccafe-batch 命令启动框架运行构件程序，前者以单进程启动框架，允许用户交互式执行；后者以批处理方式处理文件中的一组命令，以完成实例化构件、连接端口、运行等操作。ccafe-batch 可以调用 mpich 进行初始化、结束化，可以用多进程启动框架，以并行执行。以 ccafe-batch 命令调用 mpi 并行执行求解线性方程组程序的命令如下：

mpirun-p4pg pgfile /ccaffeine-0. 5. 10/bin/ccafe-batch--ccafe-rc /ccaffeine/sles. rc

其中 pgfile 指定 ccafe-batch 位置和进程数。

localhost 0 /ccaffeine-0. 5. 10/bin/ccafe-batch

localhost 1 /ccaffeine-0. 5. 10/bin/ccafe-batch

--ccafe-rc 选项指定批处理程序 ccafe-batch 要处理的文件：sles. rc。

在 ccafe-gui 下，可以直观地看到已实例化的构件和构件间通过端口的调用关系。图 3 的 Arena 中显示了实例化的构件和连接好的端口，图中红线连接的 4 个构件构成了一个 SOR 解法器。

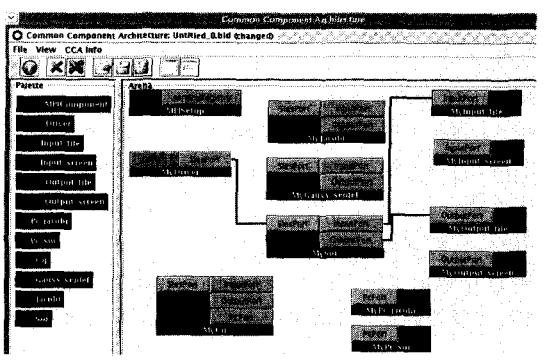

图 3 ccafe-gui 下实例化构件以及构件间调用关系图

4.4 构件化程序性能测试

为测试构件化程序与使用 c 或 fortran 结合 mpi 的并行程序的开销，设计实现以下实验。设一维椭圆偏微分方程为：

$$u_{xx}=0, x\in\Omega=[0,1] \tag{1}$$

其中，Dirichlet 边界条件为：

$$u(x)=x, x\in\partial\Omega \tag{2}$$

对求解区域[0,1]使用网格划分，用空间间隔为 $1/(n-1)$ 的长度把求解区域网格化。用 $u_i(i=0,\cdots,n-1)$ 表示有限差分近似值。对式(1)进行 3 点有限差分近似，得：

$$u_{i-1}-2u_i+u_{i+1}=0 \tag{3}$$

考虑边界值条件 $x_0=0$ 和 $x_{n-1}=1$。测试 n 取不同值时使用 c+mpi 程序与构件程序的执行时间($n=50$ 和 $n=500$ 时，由于通信时间已大于计算时间，故未比较此时的并行程序执行时间)。

从表 1 中可以看出，在方程个数为 50 时，构件程序与手工程序相比开销较明显，当方程个数为 500 时，二者执行时间差别已很小。当方程个数为 500000 时，二者串并行执行时间几乎相同。分析原因，构件程序比手工程序开销多在于引入多余的函数调用，而随着方程个数的增加，计算时间加大，函数调用时间占总执行时间的比例越来越小，甚至可以忽略。

表 1 c+mpi 程序与构件程序的执行时间比较

n	手工程序	构件程序	手工并行程序	构件并行程序
50	0.250(ms)	0.608(ms)	\	\
500	32.609(ms)	33.676(ms)	\	\
500000	282.5(s)	283.3(s)	166.18(s)	167.62(s)

结束语 本文介绍了 CCA 的概念和如何基于 CCA 开发构件程序。分析了基于 CCA 制作构件、管理构件和构建程序的步骤及实现原理。在此基础上设计实现了一款偏微分方程线形解法器，用户可以基于该解法器系统所提供的构件方便地构建程序以实现线性方程组求解，也可以进行扩展，继承已有的端口开发新的求解构件或输入输出构件。

构件化科学计算软件开发还处于探索阶段，CCA 是构件化科学计算领域的一个良好范例。

参 考 文 献

[1] CCA Forum homepage[OL]. http://www. cca-forum. org

[2] Bernholdt D E, Allan B A, Armstrong R. A Component Architecture for High-Performance Scientific Computing[J]. International Journal of High Performance Computing Applications, 2006, 20(2): 163-202

[3] 谭祎炙，黄春，赵克佳. 基于 Babel 的构件程序设计[J]. 计算机科学，2006，33(12)：235-237