基于NOW的对象式分布式程序设计语言NC++*

顾 庆, 谢 立, 陈道蓄, 吴迎红, 孙钟秀

(南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

E-mail: guq@dislab.nju.edu.cn

摘要: 提出了一个基于工作站网(network of workstations,简称NOW)的分布式程序设计语言NC++(NOW C++). 它是DC++语言的扩充. NC++提供了一个完备的编程环境,包括NC++预编译器、图视编程界面、多目通信机制和测试系统. 它完善了组管理机制和进程通信机制,提出了一个基于信度推理网络的分布共享内存(distributed shared memory,简称DSM)机制以管理C++公共变量. 实践证明,NC++语言在确保编程方便性的前提下保证了分布式程序的性能.

关键词: 分布式系统;工作站网(network of workstations,简称NOW);程序设计语言;面向对象程序设计;进程组

中图法分类号: TP311 **文献标识码**: A

随着网络技术的飞速发展,由工作站组成的分布互连、协同工作的分布式网络系统NOW(network of workstations)因其优良的扩展性能和较强的适应性日益得到广泛的应用. 程序员要在NOW中高效率地开发各种资源共享和分布并行处理的系统与应用软件,需要有一种功能强大、设计合理、学习简单的分布式程序设计语言的支持.

人们对分布并行语言的研究已有较长的历史,如今比较有代表性的语言系统有Indiana大学开发的pC++和California大学开发的CC++. 它们都以支持并行计算为首要任务. pC++主要是基于数据并行性,提供数据并行编程接口——Collection. CC++主要是基于任务并行性,提供任务并行编程接口——par. pC++是典型的SIMD模式的并行语言,不支持不同任务的分布处理. CC++在分布式语义的处理上还有待提高,如不支持组机制、用户无法有效控制任务间的通信和协作等. 最近流行的Java语言主要是基于Internet和平台无关性,而没有考虑NOW的特点,目前,在NOW环境下尚不存在一个针对它的分布式语言系统.

基于上述考虑,我们在NOW环境中以DC++[1]为基础,研制开发了一个面向对象的分布式程序设计语言NC++,它同时提供了一个完备的编程环境和一个良好的运行支撑系统.

1 NC++语言概述

NC++语言建立在流行的C++语言基础上,与C++保持兼容,并在C++语言中增加了用于描述分布式语义的语言设施.

* 收稿日期: 1999-05-17; 修改日期: 1999-11-12

基金项目: 国家"九五"重点科技攻关项目(98-780-01-07 03)

作者简介: 顾庆(1972-),男,江苏常州人,博士生,主要研究领域为分布式语言和系统;谢立(1942-),男,江苏常熟人,教授,博士生导师,主要研究领域为分布式计算,并行处理;陈道蓄(1947-),男,江苏南京人,教授,主要研究领域为分布式处理,并行计算;吴迎红(1966-),女,江苏南京人,工程师,主要研究领域为分布式系统与网络;孙钟秀(1936-),男,江苏苏州人,教授,博士生导师,中国科学院院士,主要研究领域为分布式计算,并行处理.

1.1 NC++语言的基本功能

在NOW环境中,NC++语言的主要目的是开发任务中的并行性,所采用的并行模式称为"协调问题求解(cooperative problem solving,简称CPS)".它将处理问题所涉及到的不同实体表示成不同的进程对象"process",各个对象独立自治,彼此间相互作用且能并行执行.进程对象能够动态地创建和撤销,从而使分布式程序的结构可动态调整.进程间通过对各自入口的相互调用以及对程序中公共变量的访问来保持同步并进行通信.各进程的启动和调度都由语言来管理,程序员可以静态地决定某个进程对象的启动位置.

为了增强对分布式系统的描述能力,NC++语言中设计了全新的进程组设施,组内成员进程同外部世界交互时有统一的界面.组外进程向进程组的界面申请调用,由进程组管理者负责调度的分配和协调.NC++语言还提供了支撑图视化设计和动态测试的接口,以使其进一步趋于实用化.

以下是对NC++语言中一些分布式语言设施的描述.

1.2 NC++语言的通信机制

在NC++语言中,进程间通信有3种方式:会合机制,即进程对象间的方法调用;共享变量,即C++语言中的公共变量;报文传递,针对进程对象间的信息传递.

会合机制.会合机制允许进程对象在没有被调用时自主活动,且发出非阻塞调用的进程对象可以立即执行下一步,这样就使调用发出进程和接收进程的并行工作成为可能.会合机制主要由条件入口调用组成,其语法形式如下:

〈条件入口调用〉::=select "{"
〈入口调用〉
[or〈延时语句〉]
[else〈复合语句〉]
"}"

其中延时语句的使用使实时任务的编写成为可能.NC++语言中进程对象所提供的入口调用集是可伸缩的,提供者可随时开放和关闭自己的入口调用.对应的操作语句分别是:

〈入口开放语句〉::=open〈入口名〉";"
〈入口关闭语句〉::=close〈入口名〉";"

根据执行情况,当进程对象不允许某些入口调用时,可以先关闭它们.进程接收的被关闭的入口调用会被移入被关闭的入口调用等待队列,执行select和accept语句时不检查该队列;在执行open语句时,被开放的入口调用再被移入到正常的入口调用等待队列中.

共享变量.NC++语言采用DSM(distributed shared memory)子系统来管理对公共变量的访问,这些公共变量为所有进程共享.DSM子系统的实现采用了算法库和信度推理网络模型[2],具有较强的适应性和稳定性.

NC++语言的共享变量采用严格一致性模型,所有对共享内存的访问都是原子操作.这种方式可以保证NC++程序的鲁棒性和通用性.

报文传递.NC++语言的报文传递机制包括按名通信和组通信.按名通信提供了一种依据进程对象名(即变量名)来定位进程的机制,程序员不必关心相应进程的具体位置和标识.按名通信采用异步通信模型,使用两个函数:一个用于信息传递的NameSend()函数和一个用于信息接收的NameRecv()函数.

组通信是一个以多目通信为基础的进程组管理手段,它能够保证信件传递的原子性和顺序性,

并同组成员的变动保持一致.它的使用类似于按名通信,可以根据进程组名来定位进程组.

1.3 进程组

进程组机制为功能上或结构上相似的一组进程提供了一个逻辑上的抽象.它向外界提供统的接口,而内部事物,如任务的分配等由进程组自己来管理.NC++语言中进程组的定义方式如下:

```
〈进程组〉::=group〈进程组名〉"{"
            [entry:   〈入口说明〉]
            [private: [〈私有部分〉]
                      [void GroupChange(char * name, int whatChange)〈函数体〉]
                      [void Dispatch(〈参数表〉)〈函数体〉]]
            "}""";"
```

每一个进程组都有一个组管理者"GroupServer",它负责与外界的交互,同时管理私有数据、在组内分配任务、屏蔽出错的组内成员以及管理组成员的加入和退出.进程组有一个总的入口调用等待队列,组成员执行调用有两种方式,一种是被动执行调度分配给它的任务,另一种则是向管理者申请所需的入口调用.

进程加入某个进程组有两种方法:静态加入——在定义对象或实例时,在参数表中声明所属组;动态加入——在执行体中显式执行加入语句.组成员要退出进程组时只能动态地执行离开语句.加入语句和离开语句的语法形式如下:

```
〈加入语句〉::=join〈进程组名〉";"
〈离开语句〉::=leave〈进程组名〉";"
```

1.4 NC++语言的特点

NC++语言较好地结合了面向对象的概念和进程的概念,它有如下一些特点:

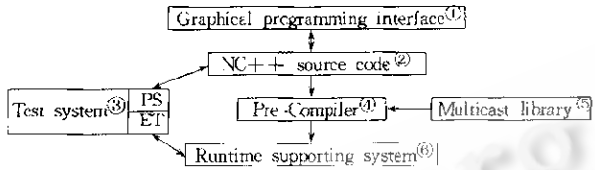
- 遵循AT&T标准,与C++语言兼容;
- 提供进程类和组类,允许程序员在源码级直接定义进程和进程组,具有一定的容错性和透明性,适于在NOW环境下编写分布式程序;
- 提供开放式的语言环境,程序员可以自行扩充语言功能,如定义自己的进程组处理函数等.
- 以C++语言为基础,采用预编译模式,易于不同机器上的移植.
- 提供完备的编程环境,方便程序员进行编程.

2 NC++语言的处理系统

2.1 NC++语言系统的总体结构

NC++语言提供了一个伴随的编程环境,以便更有效地编写分布式程序,其中包括图视编程界面、底层的多目通信机制和测试系统[3].图视编程界面提供了一个面向对象的、开放式的和可扩展的图交互接口.它以节点表示功能模块,以边反映节点间的关系,用图表示语言描述图结构,同时提供图结构与源程序间的双向转换.多目通信机制为系统和应用程序提供了一个统一的底层通信机制.它以函数接口的形式向应用程序提供服务,包括发送、接收以及状态查询等,能确保多目通信的原子性和保序性要求.测试系统为NC++语言提供了一个测试环境,提供用于显示和分析NC++程序控制和调用流程的静态测试部分,以及用于重放进程执行事件的动态测试部分,集中反映分布进程间的相互调用关系,帮助程序员发现程序的正确性问题和性能缺陷.

如图1所示,除了通过文本编辑直接输入以外,NC++程序还可以通过图视编程界面来编写.预编译器处理NC++源程序中的分布式语义,多目通信函数库为其提供底层的通信支持.预编译得到的C++代码提交给C++编译器编译以后,得到的执行程序在运行支撑系统的协助下运行于分布式环境之中.

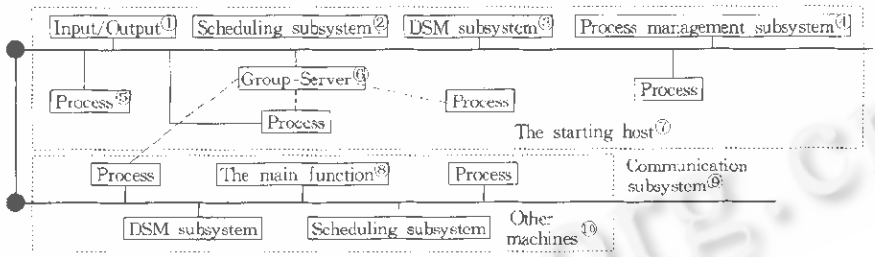


①图视编程界面,②NC++源程序,③测试系统,
④预编译器,⑤多目通信库,⑥运行支撑系统.

Fig. 1 The structure of NC++ language system
图1 NC++语言系统的总体结构

2.2 NC++语言的运行系统

NOW环境本身是分散的,其中运行分布式程序有两个基本问题[4],一个是空闲机器和资源的定位以及管理问题,另一个是如何在异构的空闲机器上建立合适的运行环境问题.NC++语言提供了一个运行支撑系统来处理这些问题,该系统的实现由NC++函数库和多目通信函数库来提供.具体包括:进程管理子系统、进程调度子系统、DSM子系统、通信子系统、进程组管理子系统和输入/输出子系统等.其结构如图2所示.



①输入输出,②调度子系统,③DSM子系统,④进程管理子系统,⑤进程,
⑥组管理者,⑦主机,⑧Main()函数,⑨通信子系统,⑩其他机器.

Fig. 2 The runtime supporting system for NC++
图2 NC++语言的运行系统

其中,通信子系统由多目通信机制和并行虚拟机(PVM)构筑而成,是整个运行支持系统的基础.DSM子系统负责共享内存的管理,在分布式系统的每个处理机上都有它的守护进程在运行.进程组管理子系统由组管理者负责,每个组管理者管理一个进程组.

调度子系统采用以信度推理网络为基础的智能调度模型[5],可根据NOW中的实际情况来选择合适的调度和资源管理算法.输入/输出子系统和进程管理子系统是集中处理的,它们被合并为一个Master进程,并与NC++程序一一对应.进程管理子系统负责进程的创建、分配和撤销,并维护一张"进程名-进程物理标识(Tid)"对照表,以实现信件的转发和目标的定位.

分布在不同处理机上的进程需要一个统一的输入/输出机制.为此,NC++库函数重写了C++语言中与输入和输出相关联的函数和对象,将进程的标准输入/输出重定向到Master进程运行的主控台上,使其给用户的感觉就像运行在一台机器上一样.

3 实例研究

我们编写并分析了两个NC++程序实例.实验环境是由多台SUN工作站和一台并行机经10M以太网互联而成的网络平台.

第1个程序是一个真实感图形的计算和绘制问题,这是一个典型的大计算量、小通信量的任务.整个程序由两大部分组成:图形计算和图形绘制.绘制部分由一个进程来完成,而计算部分则将图形均分成若干子部分并分配给不同的进程对象去计算.计算结果汇总后由绘制进程绘图.表1描述了该NC++程序在多台处理机上的执行情况.

Table 1 Running information of the graph drawing program

表1 图形绘制程序的执行情况

Number of machines①	1	2	3	4
Execution time (s)②	177	92	64	50
Accelerate rate③	—	1.92	2.76	3.54

①处理机数目(台),②执行时间(秒),③加速比.

由表1可以看出,在相对简单的任务分解方案和较低的网络速度下,NC++程序仍能获得较好的加速比.

第2个程序是一个打印机组管理程序.在该程序中使用了进程组设施:每台打印机设一个管理进程,各个打印机管理进程属于同一进程组,组管理者统一接收和分配打印任务.下面是这个程序的代码:

```
#include "stdio.h"
#define STOP_CMD 0
#define PRINT_CMD 1
#define BUF_SIZE 256
process PrinterController {
  /* 打印机管理进程 */
  entry:          void print(char * fileName);
  private:
    int printerID;
    void body()
      {while(1) accept print;}
  public:
    printerController(int id)
      {pinterID=id;}
    void print(char * fileName)
      {/* 打印文档 fileName */}
};
group PrtGrp{
  /* 组管理者 */
  entry:       void print(char * fileName);
  private:
    void GroupChange(char *,int) { }
    void Dispatch() { }   /* 采用缺省代码 */
process ConsoleType {
  /* 接收打印命令 */
  private:
    int whatCmd;
    char fileNameBuf[BUF_SIZE];
    void body( ) {
      //读入命令并存入 whatCmd 和 fileNameBuf 中
      switch (whatCmd) {
        case PRINT_CMD:
          PrtGrp.print(fileNameBuf);
          Break;
        case STOP_CMD:
          printf("Printer system stopped\n");
          halt;
        default:
          printf("Unknown command\n");
      }
    }
} console;
process PrinterController prt1(GROUP PrtGrp,1);
process PrinterController prt2(GROUP PrtGrp,2);
process PrinterController prt3(GROUP PrtGrp,3);
void main() {stop;}
```

};

这里,假设有3台打印机A,B和C,其中A的打印速度是B和C的一半.共有30个打印任务,每个任务由B或C打印时约需3分钟.表2列出了该程序模拟运行后的结果.

Table 2 Running information of the printer pool management program

表2 打印机组管理程序的执行情况

Strategy①	Distribution②	Time period③
Even assignment④	10—10—10	About 60 mirutes⑤
Process group management⑥	6—12—12	About 36 minutes⑦

①采用策略,②分配情况,③完成时间,④平均分配,⑤约60分钟,⑥进程组管理,⑦约36分钟.

由表中数据可以看出,在不增加程序编写复杂度的前提下,组机制使NC++程序获得了较好的性能.

与其他分布式语言相比,NC++语言采用预编译、C++编译加运行支撑系统的运行模式,兼顾运行速度和可移植性,比较适用于NOW环境下的分布式程序开发.NOW本身在不断发展,一些新技术在不断采用,NC++语言的开放式结构正切合这一情况.NC++的组机制和进程类机制允许程序员将分布式程序划分成多个互相协作的子集(簇),在开发较大型的程序时可以有效地避免通信阻塞,同时使程序的结构更趋合理.

4 结束语及进一步的工作

由于对象的封装和独立性与分布单元有着逻辑上的相似性,使得面向对象的分布式程序设计语言成为分布式程序设计语言研究的一个理想方向.NC++语言用主动对象来实现分布进程,则正代表着这种趋势.实践证明,NC++语言中基于对象的通信机制、进程组管理机制、调度机制和DSM机制等所进行的设计是成功的,它在确保编程方便性的前提下保证了分布式程序在NOW环境下运行的性能.

我们今后的工作包含两个方面:一方面是使NC++语言能够像Java一样包容其他语言编写的代码段;另一方面是将NC++建立在其他网络通信平台上,如MPI等,而不只是PVM.NC++语言的一些功能和子系统,如组管理进程和相关任务分配算法等,还需要作进一步的改进和完善.由于NC++具有一个可扩充的框架结构,部分功能可由用户去扩充和改进.

致谢 感谢于勐老师以及易鉴良、胡宁、李晓明等同学对本文所述工作的支持和参与.

References:

[1] Xie, Li, Yi, Jiang-liang, Du, Xing, *et al*. DC++: a kind of object-oriented distributed programming language. Journal of Electronics, 1996,24(8):102~104 (in Chinese).

[2] Yi, Jian-liang, Li, Qun, Ji, Hua, *et al*. An adaptive algorithm for keeping data consistency in a DSM system. Journal of Software, 1997,8(8):593~599 (in Chinese).

[3] Gu, Qing, Chen, Dao-xu, Xie, Li. A validation system for object oriented distributed programming language named NC++. Journal of Software, 1997,8:352~356 (in Chinese).

[4] Tanenbaum, A. S. Distributed Operating Systems. Upper Saddle River, NJ: Prentice Hall, Inc., 1996.

[5] Xie, Li, Yi, Jian liang. A model for intelligent resource management in a large distributed system. Science in China (Series E), 1997,27(6):561~565 (in Chinese).

附中文参考文献:

[1] 谢立,易鉴良,杜兴,等. DC++:一种面向对象的分布式程序设计语言. 电子学报,1996,24(8):102~104.

[2] 易鉴良,李群,汲化,等. DSM 系统中维护共享数据一致性的自适应算法. 软件学报,1997,8(8):593~599.

[3] 顾庆,陈道蓄,谢立. 基于面向对象的分布式程序设计语言 NC++的测试系统. 软件学报,1997,8:352~356.

[5] 谢立,易鉴良. 大规模分布式系统中的智能资源管理模型. 中国科学(E 辑),1997,27(6):561~566.

NC++: a NOW-Based Object-Oriented Distributed Programming Language*

GU Qing, XIE Li, CHEN Dao-xu, WU Ying-hong, SUN Zhong-xiu

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

E-mail: guq@dislab.nju.edu.cn

Abstract: An object-oriented distributed programming language NC++ which is based on NOW (network of workstations) environment is introduced in this paper. It is an extension of DC++. NC++ gives a rather complete programming environment, including NC++ pre-compiler, visual-programming interface, multicast communicating facility and a test system. It improves the process group management and inter-process communication facilities, and puts forth a DSM (distributed shared memory) subsystem based on belief network, which is used to manipulate C++ global variables. In practice, NC++ offers simplicity for distributed program design without sacrificing the performance.

Key words: distributed system; NOW (network of workstations); programming language; object-oriented program design; process group

* Received May 17, 1999; accepted November 12, 1999

Supported by the Key Sci-Tech Project of the National 'Ninth Five-Year-Plan' of China under Grant No. 98-780-01-07-03