

# 基于 MPICH 的多核并行程序设计

张沙清, 赵 洁

(广东工业大学 经济管理学院 信息管理工程系, 广州 510520)

**摘要:**概述了多核程序设计的发展以及 MPICH 的概念,介绍了利用 Visual C++ 6.0 在 MPICH 环境下进行多核并行程序设计的方法,并给出了一个多核程序设计实例,阐述了主从模式并行计算流程。

**关键词:**MPICH;多核;并行程序设计;主从模式

**中图分类号:**TP311

**文献标识码:**A

**文章编号:**1006-0707(2011)01-0104-03

20世纪70年代以来,处理器的发展突飞猛进,许多知名厂家如 Intel、AMD、IBM 等都纷纷推出一代又一代处理器。随着处理器集成晶体管数量的增多,其主频越来越高,处理器的性能也越来越强大。处理器性能不断提高主要基于2个原因:一是半导体工艺的逐渐提高;二是处理器体系结构的不断发展。但是,进入21世纪以来,传统的处理器体系结构技术已面临瓶颈,处理器集成的晶体管数目已经超过上亿个,很难单纯通过提高主频来提升性能,而且主频提高的同时也带来功耗的增加,从而直接促使处理器从单核向多核发展。另一方面,日益复杂的多媒体、科学计算、商业智能、数据挖掘、图像处理等多个应用领域都呼唤更为强大的计算能力。在这种背景下,各主流处理器厂商纷纷将产品战略从提高芯片的时钟频率转而向多内核、多线程方面发展<sup>[1]</sup>。

尽管多核技术是近几年才兴起的,但是操作系统对多核的基本支持并不难解决。从软件角度看,多核处理器和多路处理器是一样的,所有针对单核多处理器的软件优化方式都可以用在多核处理器系统上。但操作系统对多核的支持,只能提供任务一级的并行,所以多核平台上应用软件的开发与以前的软件编写思想有很大区别,设计者必须认识到底层多核的存在,需要把软件设计成多进程,并且将这些进程与底层的硬件处理器绑定,才能真正发挥多核的性能。许多软件厂商和学术团体也都纷纷推出了各自的支持多核并行计算的应用程序接口,如 MPI (message passing interface)、PVM (parallel virtual machine)、OpenMP<sup>[2]</sup>等。

本文将介绍一种基于 MPI 的支持多核并行计算的应用编程接口 MPICH 及其应用。

## 1 MPICH 概述

MPI 是目前应用最为广泛的支持多核并行程序设计的编程接口标准之一,该标准是由消息传递接口论坛 (message passing interface forum) 发起讨论并进行规范的。MPI 是一种

消息传递编程模型,定义了一个实现消息传递模型标准的程序库,最终目的是服务于进程间通信这一目标<sup>[3]</sup>。MPI 合并了包括异构性管理、模块化程序创建、延迟(异步操作)管理以及全局操作(集合操作)等各方面的复杂支持,成功地成为并行计算机的标准编程模型<sup>[4]</sup>。

MPICH 是一种最重要的 MPI 实现,其版本开发与 MPI 规范制定同步,现行最新版本为 MPICH2-1.0.8,可以从 <http://www.mcs.anl.gov/research/projects/mpich2/> 网站下载,MPICH 是 MPI-2 的完全实现。MPICH 除包含 MPI 函数库之外,还包含了一套程序设计与运行环境(包括并行性能可视化分析工具和性能测试工具等),支持 UNIX、Linux 和 Windows 平台,支持 C/C++、Fortran77 和 Fortran90 的绑定。其主要特点是:① 开放源码;② 高效率、可移植性好、功能强大;③ 支持多核 (Multi-core)、多程序多数据 (multiple program multiple data, MPMD) 编程、对称多处理系统 (symmetrical multi-processing, SMP)、异构集群系统和大规模并行计算系统。

## 2 基于 MPICH 的多核程序实现

MPICH 程序初始时在处理器的每个内核上生成一个进程,这些进程执行的程序可以相同,也可以不同。MPICH 程序不需要指明如何启动生成进程,而是将这个问题留给系统去解决(当然也可以根据实际需要,在程序执行过程中动态创建新的进程)。每个进程都必须在调用其接口之前调用 MPI\_Init 接口来初始化 MPICH 环境。然后通过调用 MPI\_Comm\_size 和 MPI\_Comm\_rank 接口分别得到缺省的进程组大小和每个进程的编号。进程之间的通信可以是点对点的,也可以是广播的。进程之间的通信可以分为标准通信模式、缓存通信模式、同步通信模式、就绪通信模式等多种模式。在标准通信模式下,进程之间的通信通过调用 MPI\_Send 和 MPI\_Recv 函数来发送和接收消息;当 MPICH 程序结束时,

收稿日期:2010-09-27

基金项目:国家自然科学基金资助项目(50875051);广东工业大学青年基金资助项目(20062014)

作者简介:张沙清(1973—),男,讲师,主要从事计算机应用、网络化制造与信息系统研究。

每个进程调用 `MPI_Finalize` 来终止 MPICH 环境。一般的 MPICH 程序的框架结构如图 1 所示。



图 1 MPICH 程序框架结构

MPICH 最基本的 2 种程序设计模式是对等模式与主从模式<sup>[5]</sup>。大部分 MPICH 程序都是对等或主从 2 种模式之一或者是二者的组合。限于文章篇幅,下面以一个简单的例子来说明主从模式 MPICH 程序设计的思路。

假设现在要求一个二维数组所有元素之和。设二维数组共有  $M$  行,则设一个主进程,  $M$  个从进程,每个从进程分别对二维数组的每行数据求和,然后将结果返回给主进程,由主进程合并从进程的计算结果。主进程和从进程的执行流程分别如图 2、图 3 所示。



图 2 主进程执行流程

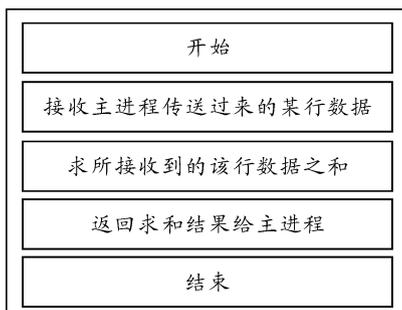


图 3 从进程执行流程

在 Visual C++ 6.0 下编写相应的代码如下 (Array2.cpp, 假设二维矩阵为 3 行 5 列):

```
#include "stdio.h"
#include "mpi.h" //MPICH 头文件
void MasterProcess();
void SlaveProcess(int rank);
```

```
void main(int argc, char * argv[])
{
    int size, rank;
    MPI_Init( &argc, &argv ); //初始化 MPICH 环境
    MPI_Comm_size( MPI_COMM_WORLD, &size ); //获得进程组大小
    MPI_Comm_rank( MPI_COMM_WORLD, &rank ); //获得进程编号
    if (rank == 0) MasterProcess(); //执行主进程
    else if (rank >= 1) SlaveProcess(rank); //执行从进程
    MPI_Finalize( ); //退出 MPICH 环境
}

void MasterProcess()
{
    int Array[3][5] = { {1,2,3,4,5}, {6,7,8,9,10}, {11,12,13,14,15} };
    int Buffer1[5];
    for( int i=0; i<3; i++ )
    {
        for( int j=0; j<5; j++ ) Buffer1[j] = Array[i][j];
        //分别将每行数据发送给从进程
        MPI_Send( Buffer1, 5, MPI_INT, i+1, i+1, MPI_COMM_WORLD );
    }
    int ArraySum = 0;
    MPI_Status status;
    for( i=0; i<3; i++ )
    {
        //分别接收从进程的计算结果
        MPI_Recv( Buffer1, 1, MPI_INT, i+1, i+1, MPI_COMM_WORLD, &status );
        ArraySum += Buffer1[0];
    }
    printf( " ArraySum = %d\n", ArraySum ); //输出最终结果
}

void SlaveProcess(int rank)
{
    MPI_Status status;
    int Buffer2[5];
    //接收主进程发送的某行数据
    MPI_Recv( Buffer2, 5, MPI_INT, 0, rank, MPI_COMM_WORLD, &status );
    int sum = 0;
    for( int i=0; i<5; i++ ) sum += Buffer2[i];
    //发送计算结果给主进程
    MPI_Send( &sum, 1, MPI_INT, 0, rank, MPI_COMM_WORLD );
}
```

要对上述代码进行编译,需要更改工程文件的某些设置(设“C:\Program Files\MPICH2”为 MPICH 的安装目录)。步

骤如下:

1) 新建一个工程(取名为 Array2),通常为 Win32 Console Application;

2) 在工程 Array2 的编辑界面下,按 Alt + F7 打开工程设置对话框;

3) 切换到“C/C++”选项卡,选中“Category:”下拉框下的“Preprocessor”,然后在“Preprocessor definitions:”的文本框中添加“HAVE\_NO\_VARIABLE\_RETURN\_TYPE\_SUPPORT”(由于 Visual C++ 6.0 不能处理多个函数只有返回类型不同而其他特征均相同的情况);同时在“Additional include directories:”的文本框中输入 MPICH 所附带的头文件的目录“C:\Program Files\MPICH2\include”,如图 4 所示。

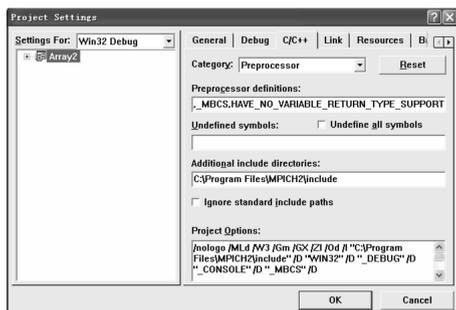


图 4 Visual C++ 6.0 设置 1

4) 切换到“Link”选项卡,选中“Category:”下拉框中的“Input”,在“Object/library modules:”的文本框中添加“cxx.lib mpi.lib”,然后在“Additional library path:”的文本框中输入 MPICH 所附带的库文件的目录“C:\Program Files\MPICH2\lib”,如图 5 所示。

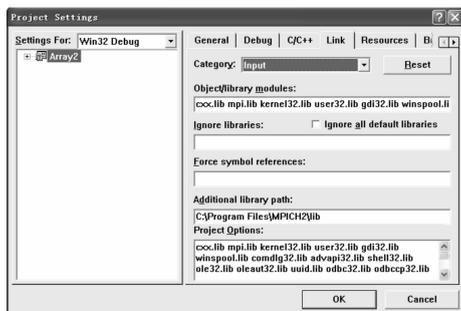


图 5 Visual C++ 6.0 设置 2

这样以后,就可以用 Visual C++ 6.0 对程序进行编译了,结果生成可执行文件 Array2.exe。接着点击“开始”->

“程序”->“MPICH2”->“mpiexec.exe”,选中要执行的可执行文件 Array2.exe,然后设置要使用的内核个数(进程个数),就可以运行程序了,如图 6 所示。

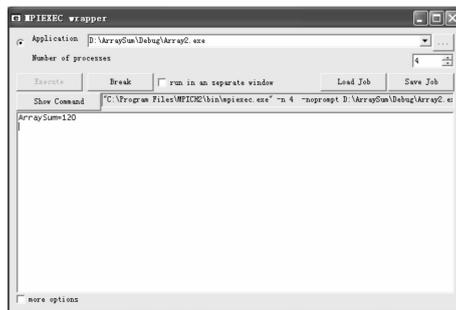


图 6 MPICH 程序运行

### 3 结束语

MPICH 的高效率、高可移植性、强大功能是其迅速普及并广泛流行的主要原因。本文讲述了 MPICH 多核并行编程的基本思路,但实际问题通常比较复杂,MPICH 提供的基本数据类型往往不能满足要求,需要用户自定义派生数据类型。另外,为了充分发挥多核的性能,需要平衡各内核之间的负载,限于文章篇幅,这些将另文探讨。MPICH 的主要缺点是不提供并行语言,而是通过并行库的方式为用户提供并行程序设计的功能,随着多核技术和高性能并行计算技术的不断发展,MPICH 的功能也将越来越完善。

### 参考文献:

- [1] 多核系列教材编写组. 多核程序设计[M]. 北京:清华大学出版社,2007:12-25.
- [2] 陈国良,安虹. 并行算法实践[M]. 北京:高等教育出版社,2004:189-192.
- [3] 都志辉. MPI 高性能并行编程技术[M]. 北京:清华大学出版社,2001:13-15.
- [4] Ian Foster, Carl Kesselman. The Grid 2[M]. BEIJING: Publishing House of Electronics Industry,2004:57-58.
- [5] 张建文,徐琼,王强. 基于 MPI 环境的并行程序设计[J]. 东华理工大学学报,2007,30(1):81-84.

(责任编辑 周江川)