XIAO XI CHUAN DI BING XING
BIAN CHENG HUAN JING

消息传递并行编程环境 MPI

莫则尧　袁国兴 编著

科学出版社
Science Press

消息传递并行编程环境 MPI

莫则尧　袁国兴 编著

科学出版社
2001

内 容 提 要

本书围绕消息传递并行程序设计，全面介绍了当前最流行的消息传递并行编程环境 MPI 1.0 版本，以及 MPI 2.0 版本中已在各类并行机上被普遍实现的部分，并简单介绍了 MPI 2.0 版本的许多新特征。

本书面向的读者是非计算机专业毕业的科学与工程计算科研人员，注重用简明易懂的语言，采用函数说明和程序实例相结合的方式来组织内容，并尽量将 MPI 的许多抽象概念具体化，使读者容易理解。由于消息传递并行程序设计的主要应用领域是科学与工程计算，因此，本书注重用 Fortran 语言结合具体应用实例来介绍，但对 C 语言也进行了详细的讨论。

从消息传递并行程序设计角度出发，本书吸取了众多国外 MPI 资料的内容，全面介绍了目前应用最广泛的消息传递并行编程环境 MPI。内容丰富，取材新颖，覆盖面广，可作为科学与工程计算部门科研人员设计消息传递并行程序的参考资料，也可作为高等院校计算数学、计算物理、计算化学、计算流体力学、计算材料、计算气象、计算生物、计算机及其相关专业的本科高年级学生和研究生的并行程序设计教学用书。

图书在版编目(CIP)数据

消息传递并行编程环境 MPI/莫则尧，袁国兴编著.

北京：科学出版社，2001

ISBN 7-03-009805-6

Ⅰ.消… Ⅱ.①莫…②袁… Ⅲ.并行程序语言-程序设计环境，MPI Ⅳ.TP.312

中国版本图书馆 CIP 数据核字(2001)第 066689 号

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

源海印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2001 年 11 月第 一 版 开本：787×1092 1/16

2001 年 11 月第一次印刷 印张：12 3/4

印数：1—2 500 字数：290 000

定价：26.00 元

(如有印装质量问题，我社负责调换〈北燕〉)

前　言

近年来，由于核禁试后加速战略计算创新(ASCI)计划的推动，世界上和我国高性能并行计算机取得了长足的发展，每秒数千亿次以上的高性能并行机对一般科研部门已经不再是一种奢望，以前许多无法求解和研究的问题现在已经成为可能。

随着并行机的发展，高性能并行计算已经在我国科学与工程计算部门和高等院校得到很好的推广，越来越多的青年科技工作者开始学习并行计算的知识。并行计算是一门交叉学科，涉及计算机、计算数学、计算物理、计算化学、计算流体力学、计算材料、计算气象、计算生物等学科，以及其他非数值应用领域，是高性能并行机成功应用的前提条件。

并行计算包含的内容较广，但大体可分为并行机体系结构、并行计算支撑环境、并行算法设计、并行程序设计和并行计算性能评价等五个方面。这五个部分相辅相存，缺一不可。其中，并行程序设计是在具体并行机上实现并行算法的关键，其复杂度远远超过了串行程序设计。一个好的并行算法，如果没有一个好的并行程序设计来具体实现，就无法得到推广应用。从 20 世纪 90 年代开始，国内并行计算专家已经出版了许多与并行计算相关的书籍，但是他们大多注重于并行机体系结构和并行算法设计，而很少进行并行程序设计的介绍。本书是对它们的一个很好补充。

目前，高性能并行机主要可分为对称多处理(SMP)共享存储并行机、分布共享存储(DSM)并行机、大规模并行机(MPP)和微机机群等四类。在这些并行机上，并行程序设计平台主要可分为消息传递、共享存储和数据并行三类，其中消息传递具有很好的可移植性，它能被所有这些类型的并行机所支持，而共享存储只能在 SMP 和 DSM 并行机中使用，数据并行只能在 SMP，DSM 和 MPP 并行机上使用。

消息传递并行编程环境 MPI(Message Passing Interface)是目前国际上最流行、可移植性和可扩展性很好的并行程序设计平台，并被当前流行的所有高性能并行机所支持。目前，国外关于 MPI 的文献和资料非常多，但这些资料大多是面向计算机专业毕业的科研人员编写的，不太适合非计算机专业毕业的科学与工程计算科研人员。在国内，全面介绍 MPI 的中文文献和资料却几乎是一个空白，这与我国对高性能并行计算的需求是不相匹配的。因此，在实际工作中，我们发现有必要针对我国非计算机专业毕业的科学与工程计算科研人员的特点，编写一本全面的、简明易懂的 MPI 并行编程参考书，它将有利于推动高性能并行计算在我国科学与工程计算部门的具体应用。

消息传递并行编程环境 MPI 1.0 版于 1994 年 6 月由全球工业、政府和科研部门联合推出，至今已经发展到 MPI 2.0 版。但是，目前流行的并行机一般只能支持 MPI 1.0 版本的全部和 MPI 2.0 版本的部分。本书面向非计算机专业毕业的科学与工程计算科研人员，以 MPI 1.0 版为基准，加上目前 MPI 2.0 版本中已经在各类并行机上被普遍实现的 MPI 并行 I/O 函数，采用函数说明与应用实例相结合的方法，由浅入深地介绍 MPI 并行编程的各个方面，最后还简单介绍了 MPI 2.0 版本的许多新的特征。同时，考虑到 Fortran 是目前在科学与工程计算部门应用最为广泛的编程语言，本书着重于用 Fortran

语言来介绍 MPI 的应用同时，对 C 语言，本书也进行了详细的讨论。

本书共分为十三章。其中，第一章介绍消息传递并行程序设计的基础知识，包括并行机体系结构和操作系统概念，第二章介绍 MPI 并行编程的一些预备知识，第三章至第九章分别介绍点对点通信、自定义数据类型、聚合通信、进程通信器、进程拓扑结构、并行 I/O和 MPI 系统环境管理，第十章讨论 MPI 与共享存储并行程序设计平台 OpenMP 的混合编程技术，第十一章给出一个求解二维 Laplace 方程的应用程序实例，它综合利用了前面各章介绍的 MPI 函数，第十二章介绍 MPI 2.0 标准的许多新特征，第十三章简单介绍 MPI 的应用现状和发展。此外，本书最后还附有 5 个附录，分别介绍 MPI 程序的编译和运行、MPI 系统的安装、MPI 网站、MPI 函数索引和 MPI 术语中-英文对照及索引。

中国科学院数学与系统科学研究所张林波研究员审阅了书稿，并提出了许多宝贵建议，在此表示衷心感谢。本书编写过程中，北京大学湍流国家重点实验室蔡庆东、北京应用物理与计算数学研究所刘兴平等老师提出了许多有益的建议，在此表示衷心感谢。

消息传递并行程序设计涉及的内容广泛，加上作者学识有限，写作时间仓促，书中错误和片面之处在所难免，恳请读者不吝批评指正。

莫则尧　袁国兴

2001 年 5 月

目　录

第一章　消息传递并行程序设计基础

本章主要讨论当前流行的并行程序设计平台所依赖的并行计算机硬件和软件环境，以及消息传递并行程序设计的基础理论知识。

1.1　并行计算环境

并行计算环境是并行算法设计和并行程序设计的基础，是并行计算发展的前提条件，它可以分为并行计算机硬件环境和软件环境两类。本节简要介绍当前流行的高性能并行计算机体系结构及其软件支持环境。

1.1.1　并行机的发展动力

自从 1972 年世界上第一台并行计算机 ILLIAC-IV 诞生以来，并行机已经经历了近 30 年的发展，它们对推动计算机技术的飞速发展和高性能计算在各个领域的应用做出了重要贡献。其中，应用问题的实际需求和微电子技术的革新是推动并行机不断发展的两个主要动力。

人类对计算机性能的需求是无止境的，在诸如预测模型的构造、工程设计和自动化、能源勘探、医学、军事、国家安全以及基础理论研究等领域中，都对计算提出了极具挑战性的需求。进入 20 世纪 90 年代以来，以美国为主的西方发达国家以应用需求为背景，提出了一系列的发展规划，极大地推动了高性能并行机的发展。其中，两个最主要的规划是美国于 1993 年提出的高性能计算与通信（HPCC：High Performance Computing and Communication）计划和 1996 年提出的加速战略计算创新（ASCI：Accelerated Strategic Computing Initiative）计划。

为了保持在高性能计算与通信领域中的世界领先地位，1993 年美国科学、工程、技术联邦协调理事会向国会提交了题为“重大挑战性项目：高性能计算与通信（HPCC）”的报告，简称 HPCC 计划，目的是研制能提供 3T 性能目标（1Tflops 计算能力、1TB 内存容量和 1TB/s 的 I/O 带宽，$1T=10^{12}$）的高性能并行机，解决科学与工程计算中的重大挑战性课题，其中包括新药设计、磁记录技术、高速民航、催化作用、燃料燃烧、海洋建模、臭氧耗损、数字解析、大气污染、蛋白质结构设计、图像理解与密码破译等。世界上第一台峰值速度超过 1Tflops 的高性能计算机是由 Intel 公司于 1996 年 12 月研制成功的。

“全面禁试条约”签定后，核武器的研究转到以实验室研究和数值模拟为基础。这样，1996 年 6 月，美国能源部联合美国三大核武器实验室（Los Alamos 国家实验室、Lawrence Livermore 国家实验室和 Sandia 国家实验室）共同提出了“加速战略计算创新（ASCI）计划”，提出通过数值模拟评估核武器的性能、安全性、可靠性、更新等，要求数值模拟达到高分辨率、高逼真度、三维、全物理、全系统的规模和能力。为此，三大实验室分别向美国三大公司（Intel，IBM 和 SGI）预定了峰值速度超过 1Tflops 的并行机，计划分四个阶段，分别

实现万亿次、10 万亿次、30 万亿次和 100 万亿次的高性能并行机。目前，前 2 个阶段已经初步实现，第 3 阶段即将实现。

与此同时，随着我国国民经济的快速发展，能源、气象、国家安全等各个应用部门对高性能计算的需求也越来越迫切，我国的并行机研制水平也得到了快速提高。

从 90 年代初期开始至今，得益于超大规模集成电路等微电子技术的革新，商用微处理器的性能几乎每年增长 1 倍，内存容量几乎每年增长 3～4 倍。为了追求最优的性能价格比，各并行机厂商纷纷改变研制方向，直接将多个高档商用微处理器通过高性能互联网络相互联接而构成高性能并行机。从 90 年代中期开始，由于网络通信技术的快速发展，高性能互联网络的拓扑结构和处理器间的距离也不再是影响并行机性能的重要关键因素。因此，高性能并行机的体系结构逐步趋于统一，从而进一步促进了操作系统、编译系统和并行编程等并行软件环境的统一。

1.1.2 并行机体系结构

对称多处理共享存储并行机(SMP：Symmetric Multi-Processing)、分布共享存储并行机(DSM：Distributed Shared Memory)、大规模并行机(MPP：Massively Parallel Processors)和微机机群(Beowulf PC-Cluster)构成了当前最流行的四类高性能并行机体系结构，它们均属于典型的多指令流多数据流(MIMD)机器。其中，MPP 又可以分为分布式存储大规模并行机(DM-MPP)和 SMP(或 DSM)大规模机群(SMP-MPP，DSM-MPP)两类。下面，我们逐一介绍这四类并行机的体系结构。

对称多处理共享存储并行机(SMP) 典型的对称多处理并行机(SMP)的体系结构如图 1.1 所示，它的基本组成单位是当前流行的商用微处理器，例如 SGI 公司的 MIPS R10000-R12000 系列、Compaq 公司的 Alpha 21264 系列、IBM 公司的 P3 系列、HP 公司的 PA9000 系列、Intel 公司的 Pentium-III 系列。为了缓和快速的处理器速度和较慢的内存访问速度之间的不匹配，每个处理器均配置 1MB-8MB 大小的局部高速缓存 Cache。所有处理器通过系统总线或交叉开关与多个内存模块和输入输出(I/O)模块相联接，所有内

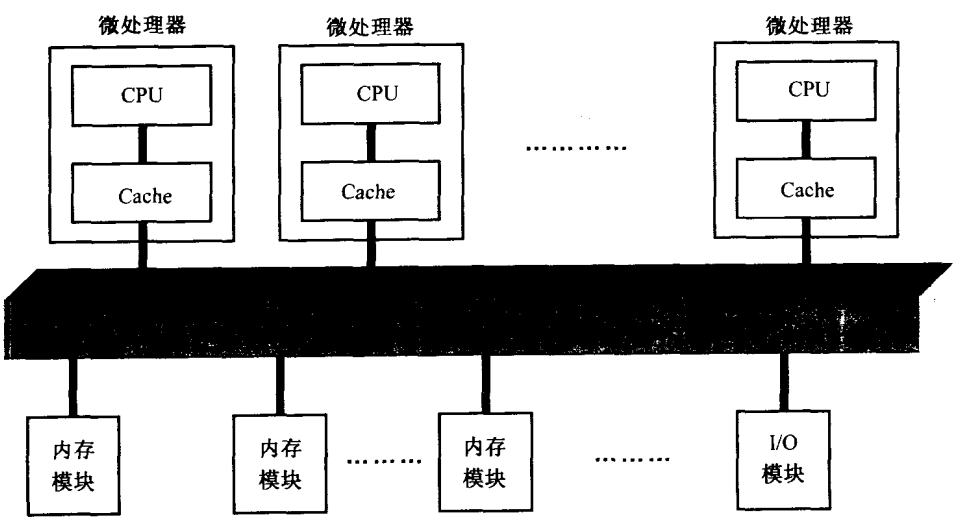

图 1.1 SMP 体系结构示意图

存模块构成 SMP 的内存地址空间，称之为 SMP 的共享存储器。

SMP 具有如下特征：(1)*对称共享存储*：系统中任何处理器均可直接访问任何存储模块中的存储单元和 I/O 模块联接的 I/O 设备，且访问的延迟、带宽和访问成功的概率是一致的。所有内存地址单元统一编址。各个处理器之间的地位等价，不存在任何特权处理器。操作系统可在任意处理器上运行。(2)*单一的操作系统映像*：全系统只有一个操作系统驻留在共享存储器中，它根据各个处理器的负载情况，动态地分配各个进程到各个处理器，并保持各处理器间的负载平衡。(3)*局部高速缓存 Cache 及其数据一致性*：每个处理器均配备局部 Cache，它们可以拥有独立的局部数据，但是这些数据必须保持与存储器内数据的一致。(4)*低通信延迟*：各个进程通过读/写操作系统提供的共享数据缓存区来完成处理器间的通信，其延迟通常小于网络通信的延迟。(5)*共享总线带宽*：所有处理器共享总线的带宽，完成对内存模块和 I/O 模块的访问。

同时，SMP 也具有如下一些问题：(1)*欠可靠*：总线、存储器或操作系统失效可导致系统崩溃。(2)*可扩展性较差*：由于所有处理器共享总线带宽，而总线带宽每 3 年才增加 2 倍，跟不上处理器速度和内存容量的发展步伐，因此，SMP 并行机的处理器个数一般少于 32 个，且只能提供每秒数百亿次的浮点运算功能。

SMP 并行机的典型代表有：SGI POWER Challenge XL 系列并行机(36 个 MIPS R10000 微处理器)、DEC Alphaserver 84005/440 (4 个 Alpha 21264 个微处理器)、HP9000/T600 (4 个 HP PA9000 微处理器)和 IBM RS6000/R40(16 个 RS6000 微处理器)。

分布共享存储并行机(DSM) 分布共享存储并行机 DSM 是对称多处理共享存储并行机 SMP 的扩展。它以结点为基本组成单位，每个结点包含多个微处理器、局部存储器和集线器(HUB)。每个微处理器拥有局部 Cache 。微处理器、局部存储器和 I/O 设备接口通过 HUB 相互联接。同时，HUB 还与一个路由器联接，所有结点的路由器相互联接形成并行机的高性能互联网络。

DSM 的典型代表为 SGI 公司的 Origin 2000 和 Origin 3000 系列并行机，图 1.2 列出了 SGI Origin 2000 的体系结构。Origin 2000 可扩展到 8 个机柜，每个机柜含 8 个结点，结点是构成 Origin 2000 的基本单位，它包含：

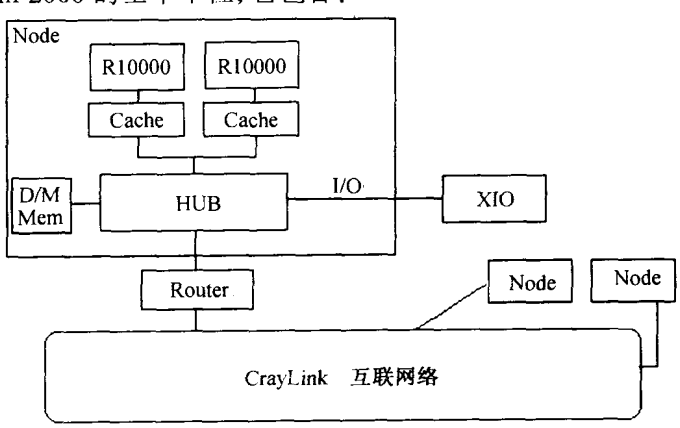

图 1.2 SGI Origin 2000 并行机体系结构示意图

- 1～2 个主频为 195MHz 或 250MHz 的 MIPS R10000 CPU，每个 CPU 含 4MB 的二级 Cache；
- 内存 512MB～4GB，分主存和目录内存两类，后者主要用于保持结点间的 Cache 数据一致性；
- 集线器(HUB)含 4 个端口：CPU 端口、内存端口、XIO 端口和 CrayLink 互联网络端口，采用交叉开关实现两个 CPU、内存、输入输出和互联网络路由器(router)之间的全互联，分别提供 780MB/Sec，780MB/Sec，1.5GB/Sec，1.5GB/Sec 的传送速率。

Origin 2000 的所有结点通过 CrayLink 高性能互联网络进行联接，路由器是构成 CrayLink 的基本单位，它包含 6 个端口，内部采用交叉开关实现端口间的全互联，具有 9.3GB/Sec 的峰值带宽。每个路由器的两个端口用于联接结点，其余 4 个端口实现路由器间的互联，形成互联网络拓扑结构。该 CrayLink 的半分带宽与结点个数成线性递增关系，对任意两个结点，至少能提供两条路径，保证了结点间的高带宽、低延迟联接和互联网络的稳定性和容错能力。

相对于 SMP，DSM 具有如下主要特征：(1)*物理上分布存储*：内存模块局部在各结点中，并通过高性能互联网络相互联接，避免了 SMP 访存总线的带宽瓶颈，增强了并行机的可扩展能力。(2)*单一的内存地址空间*：尽管内存模块分布在各个结点，但是，所有这些内存模块都由硬件进行了统一的编址，并通过互联网络联接形成了并行机的共享存储器。各个结点即可以直接访问局部内存单元，又可以直接访问其他结点的局部内存单元。为此，引入两个概念：如果某次内存访问的对象存在于结点自身的局部内存模块中，则称该次内存访问为**本地访问**；如果某次内存访问的对象存在于其他结点的局部内存模块中，则称该次内存访问为**远端访问**。(3)*非一致内存访问(NUMA)模式*：由于远端访问必须通过高性能互联网络，而本地访问只需直接访问局部内存模块，因此，远端访问的延迟一般是本地访问延迟的 3 倍以上。(4)*单一的操作系统映像*：类似于 SMP，在 DSM 并行机中，用户只看到一个操作系统，它可以根据各结点的负载情况，动态地分配进程。(5)*基于 Cache 的数据一致性*：通常采用基于目录的 Cache 一致性协议来保证各结点的局部 Cache 数据与存储器中数据的一致性。同时，我们也称这种 DSM 并行机结构为 CC-NUMA 结构。(6)*低通信延迟与高通信带宽*：专用的高性能互联网络使结点间的延迟很小，通信带宽可以扩展。例如，目前最先进的 DSM 并行机 SGI Origin 3000 的双向点对点通信带宽可达 3.2GB/秒，而延迟小于 1 个微秒。

DSM 并行机较好地改善了 SMP 并行机的可扩展能力。一般地，DSM 并行机可扩展到上百个结点，能提供每秒数千亿次的浮点运算功能。例如，SGI Origin 2000 可以扩展到 64 个结点(128 个 CPU)，而 SGI Origin 3000 可以扩展到 256 个结点(512 个 CPU)。但是，由于受 Cache 一致性要求和互联网络性能的限制，当结点数目进一步增加时，DSM 并行机的性能也将大幅下降。

大规模并行机(MPP) 大规模并行机(MPP)一词的含义并不明确，但通常是指数百个乃至数千个处理器组成的大规模并行机。例如，当前位于 TOP 500 前列的并行机均属于这一类，其中包括 IBM ASCI White(8192 个处理器)，Intel ASCI Red(9632 个处理器)，IBM ASCI Blue Pacific(5808 个处理器)，SGI ASCI Blue Mountain(6144 个处理器)，IBM

SP POWER3(1336 个处理器),CRAY T3E1200(1084 个处理器)等。按存储结构的不同,MPP 又可以分为两类:一类是分布式存储大规模并行机(DM-MPP),另一类是由多台 SMP 或 DSM 并行机通过高性能互联网络相互联接的大规模机群(SMP-MPP 或 DSM-MPP)。

图 1.3 列出了 MPP 并行机的典型体系结构,它由数百个乃至数千个计算结点和 I/O 结点组成,这些结点由局部网卡(NIC)通过高性能互联网络相互联接而成。每个结点相对独立,并拥有一个或多个微处理器(P/C)。这些微处理器均配备有局部 Cache,并通过局部总线或互联网络与局部内存模块和 I/O 设备相联接。

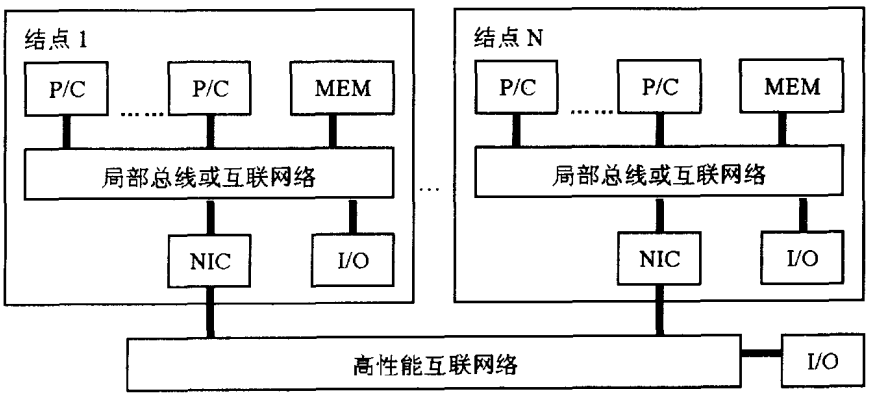

图 1.3 MPP 体系结构示意图

不同于 SMP 和 DSM,MPP 的各个结点均拥有不同的操作系统映像。一般情况下,用户可以将作业提交给作业管理系统,由它负责调度当前最空闲、最有效的计算结点来执行该作业。但是,MPP 也允许用户登录到某个特定的结点,或在某些特定的结点上运行作业。各个结点间的内存模块相互独立,且不存在全局内存单元的统一硬件编址。一般情形下,各个结点只能直接访问自身的局部内存模块,如果要求直接访问其他结点的局部内存模块,则必须有操作系统的特殊软件支持。

特别地,如果每个结点仅包含一个微处理器,则称该 MPP 为分布式存储并行机,90 年代早期的 MPP 并行机均属于这一类,其中包括 CRAY T3D, CRAY T3E, Intel Paragon, IBM SP-2, YH-3 等;如果每个结点是一台 SMP 并行机,则称该 MPP 为基于 SMP 的大规模机群(SMP-MPP),当前位于 Top500 排名前列的多台 MPP 并行机均属于这一类,其中包括 IBM ASCI White, Intel ASCI Red, IBM Blue Pacific 等;如果每个结点是一台 DSM 并行机,则称该 MPP 为基于 DSM 的大规模机群(DSM-MPP),其典型代表为包含 6144 台处理器的 ASCI Blue Mountain MPP 并行机,它由 48 台 Origin 2000 构成,其中每台含 128 个微处理器。

目前,单纯的分布式存储 MPP 并行机已经退出了历史舞台,而 SMP-MPP 已成为当前国内外并行机研制的主流方向。

微机机群(Beowulf PC-Cluster) 随着商用微处理器性能的飞速发展,低延迟、高带宽商用网络交换机的出现,和 Linux 操作系统等自由软件的成熟,并行计算机不再是一个只有大型科研单位才能拥有的设备。例如,将 128 台当前市场上最高性能的 Intel

Pentium-III/800MHz 的微机通过 6 个 24 端口的 100Mbps 的网络交换机进行联接，即可构成浮点峰值性能在 1000 亿次左右的并行机，而其成本不超过 200 万元人民币，性能价格比远远高于以上提到的各类并行机，国际上称该类自行研制的并行机为 Beowulf 机群。

尽管微机机群在通信性能、稳定性和使用方便等方面有待大幅度提高，但是，它们以其他并行机无法比拟的性能价格比，近年来已经成为了高性能并行计算中的一支不可忽视的重要力量。目前，在我国的各个大学和科研机构，例如中国科学院、北京大学、清华大学、北京应用物理与计算数学研究所等，微机机群也得到了快速发展和推广应用。特别地，在 2000 年底的 Top 500 排名中，美国 Sandi 国家重点实验室自行研制的机群 Cplant 排名第 84 位。

Beowulf 微机机群的体系结构如图 1.4 所示，多台高性能微机通过商用网络交换机相互联接，并拥有各自独立的操作系统、主板、内存、硬盘和其他 I/O 设备，构成机群的计算结点。配置一台或多台文件服务器，一方面管理机群计算结点共享的所有软件和用户计算资源，另一方面充当机群与外部网络的联接桥梁，外部科研网的用户只有通过文件服务器才能使用机群的计算资源。

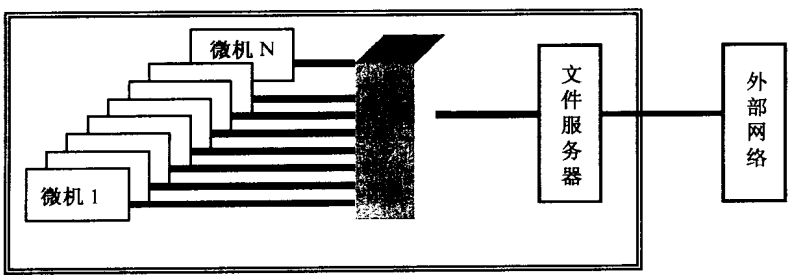

图 1.4　Beowulf 微机机群示意图

由于受商用交换机网络性能和操作系统功能的影响，Beowulf 微机机群的处理机规模一般限制在 100 台左右。但是，如果将交换机替换成专用机群网络，例如 GigaNet, Myrinet 等，则它们的规模可以进一步扩大。因此，在当前技术条件下，微机机群一般可提供千亿次左右的浮点峰值功能。

1.1.3　并行机软件环境

UNIX 操作系统几乎是当前所有高性能并行机（SMP, DSM, MPP, Beowulf PC-Cluster）采用的标准操作系统，其中包括 SGI 公司的 IRIX, COMPAQ 公司的 Tru64™, HP 公司的 HPUX, IBM 公司的 AIX, SUN 公司的 Solaris 和自由软件 Linux 等。虽然各并行机厂商研制的 UNIX 操作系统的实现原理不尽相同，但是，它们给用户提供的基本 UNIX 操作系统界面大体是一致的。因此，用户只要对 UNIX 操作系统有一定的了解，就可以方便地使用以上介绍的各类并行机。

RedHat 是 Linux 操作系统中一个应用较为广泛的分支之一，它能提供很好的机群管理功能。目前，该系统的最新版本为 7.0，属于自由软件，可在 Linux 网站（http://www.redhat.com/）中自由下载，或在市场上购买。

在程序设计语言方面，SMP, DSM 和 MPP 并行机一般均提供符合国际标准的

Fortran 77, Fortran 90, C/C++, HPF 等语言，而机群系统一般免费提供 GNU Fortran 77, GNU C/C++ 等语言。特别地，我们将在第 4 节介绍各类并行机支持的并行程序设计平台。

1.2 进程与进程间通信

现代 UNIX 操作系统中，与消息传递并行程序设计密切相关的一个重要概念便是进程。正是由于多个进程之间的相互通信，才决定了各类消息传递并行程序设计平台的出现。本节主要介绍进程和进程间通信的基本概念。

1.2.1 进程

进程(process)可表示成四元组(P, C, D, S)，其中 P 是程序代码、C 是进程的控制状态、D 是进程的数据、S 是进程的执行状态。任何进程总和程序联系在一起，程序一旦在具体操作系统环境中投入运行，就变成了进程。各个进程拥有独立的执行环境，其中包括内存数据和指令地址空间、程序计数器、寄存器、栈空间、文件系统、I/O 设备等，并在操作系统的控制、管理、保护和调度下，在不同的时刻，动态地申请和占有计算资源。特别地，我们称进程的内存地址空间为该进程的局部内存空间。

进程具有两个明显的特征：一个是资源特征，包括那些程序执行所必需的计算资源，例如程序代码、内存地址空间、文件系统、I/O 设备、程序计数器、寄存器、栈空间等；另一个是执行特征，包括那些在进程执行过程中动态改变的特征，例如**指令路径**(即进程执行的指令序列)、进程的控制与执行状态等。进程的资源特征反映了进程是操作系统调度的资源拥有的最小单位，而执行特征反映了进程是操作系统调度执行的基本单位。即使是同一个程序，不同的程序执行也将产生不同的进程，因为这些进程拥有完全不同的执行特征。

任何进程，在执行过程中，均涉及以下几种状态：(1)*非存在状态*：进程依赖的程序还没有投入运行；(2)*就绪状态*：进程由其父进程(例如，操作系统的内核进程和 Shell 进程，或其他应用程序进程)调入并准备运行；(3)*运行状态*：进程占有 CPU 和其他必需的计算资源，并执行指令；(4)*挂起状态*：由于 CPU 或其他必需的计算资源被其他进程占有，或必需等待某类事件的发生，进程转入挂起状态以后，一旦条件满足，由操作系统唤醒并转入就绪状态；(5)*退出状态*：进程正常结束或因异常退出而被废弃。

只对消息传递并行程序设计感兴趣的读者，了解以上的进程概念就够了，有关进程的详细定义，将涉及到进程的影像、进程的执行模式、进程的现场活动、进程描述符和进程控制等诸多方面的知识，有兴趣的读者请参考专门的操作系统专著。

1.2.2 进程间通信

多个进程可同时存在于同一台处理机中，拥有独立的执行环境，在操作系统的调度下，分时共享计算机资源。但是，它们拥有的内存指令和数据地址空间必须互不相交，且每个进程只能访问自己的局部内存空间。如果一个进程执行的某条指令要求访问该进程局部内存空间之外的内存地址单元，则隐含该进程的程序存在错误，而进程的执行可能会

中断。当然，位于不同处理机中的多个进程也拥有独立的执行环境，在各自操作系统的调度下，占有各自的计算机资源。

无论是位于同一台处理机中还是位于不同处理机中，进程始终是操作系统资源调度的基本单位，且各个进程不能直接访问其他进程的局部内存空间。但是，现代操作系统提供基本的系统调用函数，允许位于同一台处理机或不同处理机的多个进程之间相互操作交流信息，操作具体表现为三种形式：通信、同步和聚集。

通信 进程间的数据传递称为进程间通信。在同一台处理机中，通信可以通过读/写操作系统提供的共享数据缓存区来实现；在不同处理机中，通信可以通过网络传输数据来实现。特别地，称两个进程之间传递的数据为**消息**，称这种操作为**消息传递**。显然，消息传递可以在同一台处理机的多个进程之中发生，也可以在不同处理机的多个进程之间发生。

同步 同步是使位于相同或不同处理机中的多个进程之间相互等待的操作，它要求进程的所有操作均必须等待到达某一控制状态之后才进行。其实，同步也是进程之间相互通信的一种方式。

聚集 聚集将位于相同或不同处理机中的多个进程的局部结果综合起来，通过某种操作，例如求最大值、最小值、累加和，产生一个新的结果，存储在某个指定的或者所有的进程变量中。其实，聚集也是进程间相互通信的一种方式。

在以后的讨论中，为了方便，我们将进程间相互操作的三种形式：通信、同步和聚集，统称为**进程间通信**，而操作的具体数据对象为**消息**，具体操作为**消息传递**。

进程间通信的具体实现大体可以分为两类：(1)在共享存储环境中，通过读/写操作系统提供的共享数据缓存区来实现；(2)在分布式存储网络环境中，通过套接字(Socket)网络通信来实现。但是，无论哪种形式，实现的具体细节对并行编程的用户都是屏蔽的，用户看到的均是统一的应用程序接口(API)。

在以后各章消息传递 MPI 并行编程的详细介绍中，读者将会逐步深入理解进程间通信各种方式的具体含义。

1.3 线　　程

为了更好地理解消息传递并行编程环境，我们介绍另一个重要概念：**线程**(Thread)，它是在进程的基础上，基于对称多处理的现代操作系统的一个重要发明。

由于进程具有独立的局部内存空间，使得操作系统对它们的管理非常费时。例如，当 Unix 进程执行系统调用(fork)生成一个子进程时，操作系统就必须为该子进程分配内存地址空间和寄存器、复制父进程描述符、设置运行栈空间、保留进程上下文及切换进程，所有这些操作是非常费时的。为此，我们称该类 Unix 进程为**重量级进程**。

由于管理重量级进程的高开销较大地影响了并行机性能的发挥，因此，该类进程不适合细粒度的共享存储并行程序设计。为了在共享存储环境下有效地开发应用程序的细粒度并行度，我们将一个进程分解为两个部分，其中一部分由其资源特征构成，称之为进程；另一部分由其执行特征构成，称之为**线程**，或者**轻量级进程**。具体地，如图 1.5 所示，进程的指令路径可以分解为并行的互不相关的多条子路径(用曲线表示)，而每条子路径可由