第十章 高性能计算和并行算法

§10．1引言

在过去的近三十年间，我们经历了计算机科学和技术的迅速发展。具体表现在市场上计算机的性能价格比得到迅速地提高。

高性能计算机的概念并无明确的定义，一般认为运算速度非常快的计算机就可以认为是高性能计算机。严格地讲，高性能计算机是一个所有最先进的硬件，软件，网络和算法的综合概念，“高性能”的标准是随着技术的发展而发展的。

高性能计算系统中最为关键的要素是单处理器的最大计算速度，存贮器访问速度和内部处理器通讯速度，多处理器系统稳定性，计算能力与价格比，以及整机性能等。

冯.纽曼“瓶颈”。

为了要超越这个“瓶颈”，人们发展了两种计算机体系结构和相关软件技术的应用原则。一个是并行算法(parallelism)，另一个是流水线技术(pipelining)。

实际上，这两个原则都与并行有关。实际上并行的概念有两种：一种是仅仅时间上的并行，称为流水线计算机，另一种既是时间，也是空间上并行的并行计算机。

流水线技术是源于工业自动化生产中的流水线操作思想。在生产流水线上，一批产品的一部分安装完毕以后，进入流水线的下一段完成另一部分安装任务；整个流水线上同时对产品进行不

同部分的安装工作。在并行处理中，流水线技术是一项重要的并行技术，目前已经在超级计算机和工作站上得到广泛的应用，

并行计算机上算法的具体实现则要复杂得多，要高效使用它就更不容易。造成这样局面的原因是修改算法来适应流水线操作，比去适应并行计算容易一些。但是。毫无疑问,向量处理超级计算机和并行计算系统的应用会给我们物理学的发展提供更多的机遇。

由于高性能计算机与当前能够应用的新计算技术相关联，因而它与并行算法和流水线技术有着密切的联系。

§10．2 并行计算机和并行算法

并行计算机是由多个处理器组成(在这些处理器之间还可以相互通讯和协调)，并能够高速、高效率地进行复杂问题计算的计算机系统。并行计算机的得名是相对于串行计算机而言的。串行计算机是指只有单个处理器，顺序执行计算程序的计算机，也称为顺序计算机。并行计算作为计算机技术，是在上世纪 70 年代中期提出来的，至今已有近 30 年的发展历史了。该技术的应用已经带来单机计算能力的巨大改进。

为什么要采用并行计算呢？我们有三个方面的理由：

(1) 并行计算可以大大加快运算速度，即在更短的时间内完成相同的计算量，或解决原来根本不能计算的非常复杂

的问题。

(2) 提高传统的计算机的计算速度一方面受到物理上光速极限和量子效应的限制，另一方面计算机器件产品和材料的生产受到加工工艺的限制，其尺寸不可能做得无限小。因此我们只能转向并行算法。

(3) 并行计算对设备的投入较低，既可以节省开支又能完成计算任务。

通常的冯.纽曼计算机是属于 SISD(Single Instruction Single Data stream computers) 单指令单数据流计算机类型计算机，它的结构只有一个处理器，同时可以处理一个单数据流。其单数据流是指被处理器从存贮器取出或写到存贮器上的数据序列。

并行计算机若采用与某种网络相关的多处理器结构，则会使其性能得到改善。计算机系统的处理器和内存条有各种各样的可能排列。我们可以根据一个并行计算机能够同时执行的指令与处理数据的多少，将如今广泛使用的结构分成以下类型：

(1)SIMD 计算机具有处理器器件阵列，它通过单个控制单元来控制功能设备。这个控制单元向所有处理器发出相同的执行指令，随后所有处理器对进入的不同数据流进行相同的操作。例如在 SIMD 并行计算机上做数组的赋值运算

X=X+1

即用加法指令对数组 X 的所有元素同时做加 1 的操作。可以看出

在 SIMD 并行计算机上特别适合进行向量（数组）的运算，它的结构可以以很高的处理速度直接支持这种运算。

(2)MIMD 计算机具有可以同时独立运行多条指令，对不同的数据进行操作。例如对数组的如下运算 A=B*C+D+E+F/G,可以分解为乘法（B*C)，加法（D+E）和除法（F/G）直接交给 MIMD 计算机的直接处理部分同时进行计算。

在非常大的科学计算程序中，大部分的工作常常是反复地重复同样的操作。SIMD 结构更适用于处理这类问题的计算。因为在同样的价位下，SIMD 体系结构的处理器元件可以做得比 MIMD 计算机的所有处理器更快。但是 MIMD 计算机明显地提供了更大的灵活性，并可以用作多用户计算机，而这在 SIMD 计算机上是不可能的。

按照同时执行程序的不同并行计算机又可以分为：

(1) SPMD（Single Program Multiple Data Stream Computers）单程序多数据流并行计算机;

(2) MPMD （ Multiple Program Multiple Data Stream Computers）多程序多数据流并行计算机。这种划分依据的执行单位不是指令而是程序。

SPMD 并行计算机一般是由多个相同的计算机或处理器构成；而 MPMD 并行计算机内计算机或处理器的地位是不同的，根据分工的不同，它们擅长完成的工作也不同，因此可以根据需要将不同的程序放到 MPMD 并行计算机上执行，使得这些程序协调一致地

完成给定任务。

并行计算机也可以按照内存的结构来划分成“分布式内存”和“共享式内存”两种基本结构的并行计算机。

在分布式系统中每个处理器有它自己的局域内存，不存在公共可用的存贮单元，各处理器可以通过通讯网络相互作用(例如，与它们的局域内存交换数据)。

在共享式内存结构中，所有处理器都能够通过某些通讯网络访问共享的内存(有时它们也可以与向量寄存器或其它处理器的高速缓冲存储器沟通)。在仅仅只有单地址空间的情况下，共享式内存结构比分布式内存更容易编程。

分布式内存模型在当今几种超级计算机和并行计算机工作站上已经采用，这些并行计算机上都装有数十个功能强大的向量处理器。然而，目前流行的机群计算（cluster computing）大部分采用分布式共享内存模式结构。

并行算法是在给定并行模型下的一种具体明确的计算方法和步骤。其分类有不同的分类方法。根据并行计算任务的大小分类，可以分为三类：

(1) 粗粒度并行算法；它所含的计算任务有较大的计算量和较复杂的计算程序。

(2) 细粒度并行算法；它所含的计算任务有较小的计算量和较短的计算程序。

(3) 中粒度并行算法；它所含的计算任务的大小和计算

程序的长短在粗粒度和细粒度两种类型的算法之间。

根据并行计算的基本对象可以分为：数值并行计算和非数值并行计算。实际上两者之间并无严格的界限。非数值计算也会用于高精度数值计算，数值计算中也会有查找、匹配等非数值计算成分。实际分类时，主要是根据主要的计算量所属范畴以及宏观的计算方法来判断。

根据并行计算进程间的依赖关系可以分为：

(1) 同步并行算法；该算法是通过一个全局的时钟来控制各部分的步伐，将任务中的各个部分计算同步地向前推进。

(2) 异步并行算法；它执行的各部分计算步伐之间没有关联，互不同步；在操作中，它们根据计算过程的不同阶段决定等待、继续或终止。对于 SIDM 并行计算机一般适合用同步并行算法，而 MIMD 并行计算机则适合用异步并行算法。

§10．3 并行编程

设计一个高效的并行算法并不是一件容易的事，它的设计过程比较复杂。通常编程设计过程可以分为四步：

- 任务划分(Partitioning)。该阶段将整个使用域或功能分解

成一些小的计算任务，它的目的是要揭示和开拓并行执行的机会；

- 通信(Communication)分析。由任务划分产生的各项任务一般都不能独立完成，可能需要确定各项任务中所需交换的数据和协调各项任务的执行。通信分析可以检测在任务划分阶段划分的合理性；
- 任务组合(Agglomeration)。按照性能要求和实现的代价来考察前两个阶段的结果，必要时可以将一些小的任务组合成更大的任务以提高执行效率和减少通信开销；
- 处理器映射(Mapping)。这是设计的最后阶段。要决定将每一个任务分配到哪个处理器上去执行。目的是要最小化全局执行时间和通讯成本，并最大化处理器的利用率。

上述过程简称为 PCAM 设计过程。

目前最重要的并行编程模型：

(1) 数据并行（Data Parallel）模型（它的初衷是为了 SIMD 并行机)、

(2) 消息传递(Message Passing)模型（其初衷是为了多计算机)、

(3) 共享变量模型、

(4) 函数式模型等，

后面两种的应用都没有前面两种那样普遍。

数据并行的含义是将相同的操作同时作用于不同数据，因此不但适用于在 SIMD 计算机上实现，也可以在 SPMD 并行计算机上运行，这取决于粒度大小。SIMD 程序着重开发指令级中细粒度的并行性，SPMD 程序着重开发子程序级中粒度的并行性。在向量机上通过数据并行求解问题的实践也说明数据并行可以高效率地解决一大类科学和工程计算问题。数据并行编程模型是一种较高层次上的模型。它提供给编程者一个全局的地址空间。一般这种形式的语言本身就提供并行执行的语义。对于编程者，实现数据并行的程序，只需要简单地指明执行什么并行操作以及并行操作对象。例如对于数组的运算，通过语句

A=B+C（或其它的表达方式）

就可以实现 B 和 C 数组的对应元素相加后结果赋给 A。因此数据并行的表达是相对简单和简洁的，它不需要编程者关心并行机是如何对该操作进行并行执行的。对于非数据并行类问题编程模型，如果也采用数据并行的方式来解决，一般难以取得高的效率，数据并行不容易表达，甚至无法表达其它形式的并行特征。数据并行编程目前面临的主要问题是要实现高效的编译。有了高效的编译器，数据并行程序就可以在共享内存和分布式内存的并行机上都得到高执行效率。有了高效的编译器，就可以提高并行程序的开发效率，提高并行程序的可移植性。

消息传递编程模型是在各个并行执行部分之间传递消息，相互通讯。消息可以是指令、数据、同步信号或中断信号等。消息

传递一般是对分布式内存并行计算机的方法，但是也可以适用于共享式内存的并行计算机。消息传递为编程者提供了更灵活的控制手段和表达并行的方法，一些用数据并行很难表达的并行算法都可以用消息传递编程模型来实现。消息传递编程模型比数据并行编程模型更灵活，还具有各种各样的控制手段，这就可以使程序高效率运行。消息传递程序是由多个进程组成，每个进程都有自己的控制线。由于采用消息传递编程模型需要编程者来明确地为进程分配数据和负载，因此它也使得编程者的工作量增加，其编程的级别比较低。虽然这样，消息传递的基本通信模式是简单和清楚的，学习和掌握这些部分并不困难，因此大量的并行程序设计仍然是消息传递并行编程模式的。

并行程序是需要通过并行语言来表达。并行语言的产生有三种方式：

（1） 设计全新的并行语言；

（2） 扩展原来串行语言的语法成分，使它支持并行特征；

（3） 为串行语言提供可调节的并行库，并不改变串行语言。

目前常用的是第（2）和（3）两种方式，最常用的是第（3）种。并行语言的发展十分迅速，并行语言的种类也很多，但是真正使用起来并被广泛接受的语言却寥寥无几。对 FORTRAN 和 C 语言的扩充是最常见的并行语言产生办法。如 MPI 就是 FORTRAN 和 C 语言结合起来实现的。