并行计算机数值方法导论

〔德〕U. 申德尔 著

武汉大学出版社

并行计算机数值方法导论

〔德〕U. 申德尔 著

张正言 郑慧娩 朱方生 译

徐 绪 海 审校

武汉大学出版社

1989

内 容 提 要

本书的原著版本由自由柏林大学计算数学与计算机科学教授 U. 申德尔（U. Schendel）著、奥尔登堡(Oldenbourg) 出版社出版。1984年伦敦大学 Chelsea 学院运筹学教授 B.W. 科诺利把它译成英语，由埃利斯·霍伍德（Ellis Horwood）公司出版。中译本就是根据这一英译本翻译的。

本书简要地介绍了可供使用的并行计算机模型，较详细地讨论了构造并行数值方法的原则，并具体地介绍了求解递推问题、线性代数方程组、特征值问题、非线性问题以及非线性优化问题的并行数值方法。

本书可作为计算数学、计算机软件与硬件等专业高年级学生和研究生的教材或教学参考书，也可供有关的科技工作者使用。

并行计算机数值方法导论

〔德〕 U. 申德尔 著

张正言 郑慧娩 朱方生 译

徐 绪 海 审校

*

武汉大学出版社出版

（武昌 珞珈山）

新华书店湖北发行所发行 武汉大学印刷厂印刷

*

850×1168毫米 1/32 5.625 印张 140 千字

1989年12月第 1 版 1989年12月第 1 次印刷

印数：1——1000

ISBN 7-307-00611-1/O·54

定价：1.40元

译 者 说 明

本书是在张正言教授指导讨论班的基础上组织翻译的，具体分工如下：张正言译第一章，朱方生、郑慧娆译其余各章和两个附录，徐绪海副教授译前言并审校全稿。在译的过程中，得到张延昌、陈莘萌、费浦生等教授的热情帮助，在此表示衷心感谢。由于时间仓促，水平有限，不妥之处恐难避免，恳请读者批评指正。

英译版序言

值此德文原版译成英文版问世之际，作者和译者都希望并行数值分析这一迷人的学科能赢得广大读者的关注。到目前为止，就本书所涉及的现代先进计算机而言，阐述硬件方面的文献甚多且在不断增加，而根据需要，设计出适当数量的配套数学软件，这一问题却还未得到较好的解决，更没有提供较为系统的文本。不过，这一课题很有吸引力，它引起了很多人的注意，这些人对用现代技术提供改进计算的可能性十分感兴趣。我们希望，本书能成为一本入门书，并能提供现有方法的文件编制。

趁此机会，对原书增加和补充了一些内容，纠正了错误和错印之处。增补部分包括：第四章中求特征值方法的扩充和两个附录。两个附录含有写本书期间未能得到的一些资料。第一个附录补充第一、二两章，提供一些特定机器的技术信息及其按基准测试得到的性能特征。第二个附录简要地叙述一些非线性优化的并行算法。

译者和作者都非常欣赏这次愉快和富有成效的合作。这次合作，不仅建立了彼此间的友谊，而且提供了同行之间合作共事的有益经验。

作者非常感激科诺利（B.W.Conolly）教授对本书所作的翻译和改进，还要感谢我的同事勃兰登布格尔(J.Brandenburger)和席斯卡（M.Schyska）两位数学学士，他们阅读了全部手稿。

U. 申德尔（柏林）

B.W. 科诺利（伦敦）

1984年

前言

本书是在自由柏林大学介绍并行数值分析原理时所作一系列讲演的结果。并行数值分析是数值数学的一个分支，这一分支已从新型计算机结构的发展，特别是从并行计算机这一概念中，获得了推动力和明显的意义。并行计算机的出现已使比现今规模更大、更复杂的问题得以解决成为可能。用于求解这种问题的方法常常必须予以“并行化”，保证它们在相应的并行计算机上能成功地进行计算。

为了发展并行数值方法，本书介绍了一些公认的原则。因为至今还未能提出在理论上统一的、用于研究并行数值方法的标准计算机模型，所以在讨论中提出的各种概念还与特定种类的计算机密切相关。本书的选材大体上限制在数值分析中经典的、成熟的部分数值方法，同时也讨论了开发特殊问题并行算法的可能途径。

本书的首要目的是要让读者对并行数值方法的原理有一个基本的了解，并使它成为解决问题的初步指南。书中提供的数学内容难易适中，科学工作者、工程师及具有同等水平的人员都不难接受。

我要感谢法伊迈埃尔（Feilmeier）教授（不伦瑞克）和萨梅（Sameh）教授（伊利诺斯，厄巴纳）多次和我进行的有启发性的讨论，感谢我从前的同事库普弗内格尔（K. Kupfernagel）数学学士所作的贡献和重要的评注。最后，特别要感谢奥尔登堡（Oldenbourg）出版社的关心和良好的合作。

U. 申德尔

1981年于柏林

目　录

第一章 引 论

目前，用以描述数值方法的数字计算机模型是著名的 von Neumann 模型，如图1.1所示。

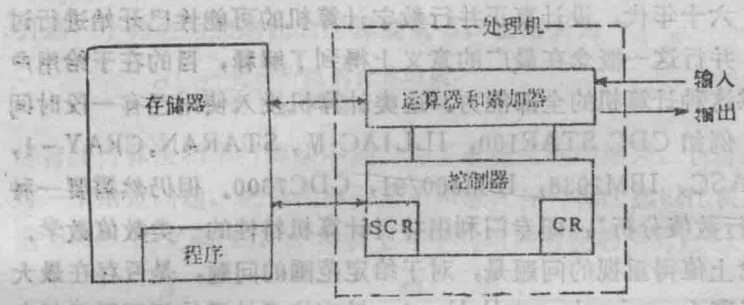

图1.1 通用计算机

在图1.1中

SCR＝顺序控制寄存器（存放下条指令的地址）。

CR＝控制寄存器（存放当前指令的复本）。

通用计算机具有如下特征：

（1）变量的数字表示；

（2）根据基本的算术运算和逻辑运算，顺次进行处理；

（3）程序是实现算法的编码形式，数据存放在主存储器中。

所有进一步的发展都是以这种基本模型为其起点的。特别是数值分析的算法，它们都以这种类型的计算机模型为基础，并且需要大量的初等运算。从而使舍入误差及其传播问题、数值稳定性问

题都变得十分重要[1].

近廿年来，由于新技术的运用和设计的改进，串行机器的性能大大提高，并行的某些特点业已采用，这包括：

（1） 输入/输出通道的结构；

（2） 执行指令的重叠；

（3） 交错存储技巧.

然而，这些发展已落后于数值方法的要求，甚至落后于程序设计这一级.

六十年代，设计真正并行数字计算机的可能性已开始进行讨论，并行这一概念在最广的意义上得到了解释，目的在于给用户提供这种计算机的全部能力. 这类计算机投入使用已有一段时间了，例如 CDC STAR100, ILLIAC Ⅳ, STARAN, CRAY-1, TI ASC, IBM2938, IBM360/91, CDC7600. 但仍然需要一种'并行数值分析'，即专门利用并行计算机特性的一类数值数学. 理论上值得重视的问题是，对于给定范围的问题，是否存在最大并行度（maximal parallelism）. 具有这类性质的问题属于复杂性理论的范畴.

下面列出并行计算机的利和弊.

有利的方面：

（1） 提高了计算速度

(a) 常规结构要受其物理限制，

(b) 具有 n 台处理机的单台计算机比每台只有单台处理机的 n 台计算机要便宜一些.

（2） 能求解对串行计算机而言仍过于复杂的问题. 世界范围内的天气预报就是一例.

（3） 能求解本来就是并行的问题. 向量运算和离散模拟提供了这方面的例子.

（4） 实时问题. 这方面的例子有印刷资料的处理（图片和

图形的处理）和航空测量.

不利的方面：

（1） 这种机器利用不佳（管理问题）.

（2） 复杂的数据组织(并行存取问题).

目前，还没有并行系统的标准模型，这是一个很大的困难. 当我们研究什么叫 SIMD 和 MIMD 模型时，再进一步考察这个问题.

估计 p 个处理机并行运算增加的速度是极为重要的. 为此，引进一个称之为加速比的运算度量 S，它定义为

$$S=\frac{\text{一台串行机的计算时间}}{\text{一台并行机的计算时间}}. \tag{1.1}$$

尽管并行算法和串行算法不同，为了获得正确的结果，也有必要对一个给定问题，把最优串行算法和最优并行算法加以比较. 根据 Stone 的文献[2]，S 实质上有下列四种可能的形式：

	S	例　子
(a)	$S=kp$	矩阵计算，离散化.
(b)	$S=kp/\log_2 p$	分类，三对角线性系统，线性递推公式，多项式计算.
(c)	$S=k\log_2 p$	搜索.
(d)	$S=k$	某些非线性递归和编译运算.

这里 k 是满足 $0<k<1$（且接近 1）而依赖于机器的一个量，p 是处理机的台数. 但是，在(a)至(d)各条还没有在复杂性理论中得到证明之前，按其任何一条对特殊问题的分类都是临时性的，一种合理的并行数值方法所能达到的性能极限在(b)和(c)之间. 对于并行计算机，一种算法所需的计算时间既不知道又与机器有关. 为了能够估价一种并行算法的优点，需要计算花费的时间单位步数. 为此，引入一种理想化的概念是方便的，即在这样一个时间单位步内，恰好在并行模式中完成一种算术运算(见第 4 章). 设

T_p 是用 $p(\geqslant 1)$ 台处理机设计的并行算法所需的时间单位步数，而 T_1 是相应串行算法所需的时间单位步数．为了估量并行运算的特征，引入下列参数．

定义 和串行算法相比，并行算法的加速比定义为

$$S_p = T_1/T_p \quad (\geqslant 1)$$

(参见1.1)，其效率定义为

$$E_p = S_p/p \quad (\leqslant 1). \tag{1.2}$$

E_p 用以量度并行机利用发挥的程度．处理机空闲的时间越长，或者由于问题的并行化引起额外的计算越多，E_p 的值就越小．

为了对同一问题比较两种不同的并行算法，下面引入一种量度 F_p：

$$F_p = S_p/C_p, \tag{1.3}$$

其中

$$C_p = pT_p \tag{1.4}$$

是度量算法‘耗费’的一个量，称其为效用．注意到

$$\begin{aligned} F_p &= S_p/(pT_p) = E_p/T_p \\ &= E_p S_p/T_1 \leqslant 1. \end{aligned} \tag{1.5}$$

则 F_p 就是既量度加速又量度效率的一种量．如果一种并行算法能使 F_p 达到最大，则相应地就认为它是有效的．

下面举一个简单的例子予以说明，假设要求和

$$A = \sum_{1}^{16} a_i.$$

用一台处理机串行计算，需做15次加法．如果把做加法的次数作为对所需时间的一种度量，那么在假定做一次加法所需的时间为一个单位的规定下，有 $T_1 = 15$．如果有两台处理机可用，则可

同时求两个和

$$b_1=a_1+a_2+\cdots+a_8,\qquad b_2=a_9+a_{10}+\cdots+a_{16}$$

这需要 7 个时间单位，然而下一步再求

$$c=b_1+b_2$$

又需要一个时间单位。这样求 A 总共需要 $T_2=7+1=8$ 个时间单位。如果有三台处理机，求 A 可分三级进行：

$$b_1=a_1+a_2+a_3+a_4+a_5,$$

$$b_2=a_6+a_7+a_8+a_9+a_{10},$$

$$b_3=a_{11}+a_{12}+a_{13}+a_{14}+a_{15};$$

$$c_1=b_1+b_2,\ c_2=b_3+a_{16};$$

$$d_1=c_1+c_2=A.$$

这需 $T_3=4+1+1=6$ 个时间单位。如果有四台和八台处理机，下面的过程及其对应的时间都是可行的。

$p=4$

$$\left.\begin{aligned}&b_1=a_1+\cdots+a_4,\ b_2=a_5+\cdots+a_8,\\&b_3=a_9+\cdots+a_{12},\ b_4=a_{13}+\cdots+a_{16}\end{aligned}\right\}\qquad(3)$$

$$c_1=b_1+b_2,\ c_2=b_3+b_4\qquad(1)$$

$$d_1=c_1+c_2\qquad(1)$$

$$T_4=5.$$

(　)中的数字表示所需的时间单位。

$p=8$

$$b_1=a_1+a_2,\ b_2=a_3+a_4,\ \cdots,\ b_8=a_{15}+a_{16}\qquad(1)$$

$$c_1 = b_1 + b_2,\ c_2 = b_3 + b_4,\ \cdots,\ c_4 = b_7 + b_8 \qquad (1)$$

$$d_1 = c_1 + c_2,\ d_2 = c_3 + c_4 \qquad (1)$$

$$A = d_1 + d_2 \qquad (1)$$

$$T_8 = 4.$$

现在就可列出一种性能量度表

p	T_p	C_p	S_p	E_p	$F_pT_1 = S_pE_p$
1	15	15	1	1	1
2	8	16	1.88	0.94	1.76
3	6	18	2.5	0.83	2.08
4	5	20	3	0.75	2.25
8	4	32	3.75	0.47	1.76

从表中可以看出，随着 p 的增加，S_p 就平稳地增大，而 E_p 减小。但当 $p=4$ 时，F_pT_1 有一最大值。这表明，对于这种计算，$p=4$ 是对处理机数量的最优选择。但是，正如我们已经看到的，对于每个 p 值，算法是大不相同的。引起 S_p，E_p 和 F_p 的变动，部分地是由于对同一问题采用了不同并行算法的缘故。上面引进的参数只给出了估价并行算法的一种量度标准，其它要考虑的方面是稳定性和误差分析。但最重要的还是要识别哪些问题已经具有并行特征和哪些问题可以‘并行化’。

第二章　可能的计算机模型

要系统地阐述一个适用于并行数值方法的标准机器模型那是困难重重的，然而这样的模型无论是对理论分析还是为算法研究提供背景来说都是必要的*)。现在的情形与经典数值分析中遇到的完全不同，在那里，von Neumann 的通用计算机模型为我们打下了一个基础。图2.1表示一台并行计算机的通用模型。不过，这个模型对于详细说明计算机的有关功能来说，还是显得过于一般化了。特别，它对于算法研究太一般化了。进一步的补充讨论见附录1。

图2.1　具有各种并行层次的并行计算机的一般结构
S_i：存储器；P_i：处理机

注意：并行是可能的

（1）　在控制器内；

（2）　在处理机之间；

（3）　在存储器之间；

（4）　在数据控制网络中。

*) Ibbett[3]、Zakharov[50]和 Hockney[51]为现代高性能的机器作了有用的讨论，不过，这些讨论对初学者稍专门化了一点。

现在我们来介绍 Flynn[4]对计算机作出的一种重要分类。首先我们考虑指令流数目并按这些数目把机器分为下面的类型：

SI＝单指令流

MI＝多指令流　　（2.1）

其次，按操作数流的数目来区分：

SD＝单数据流

MD＝多数据流　　（2.2）

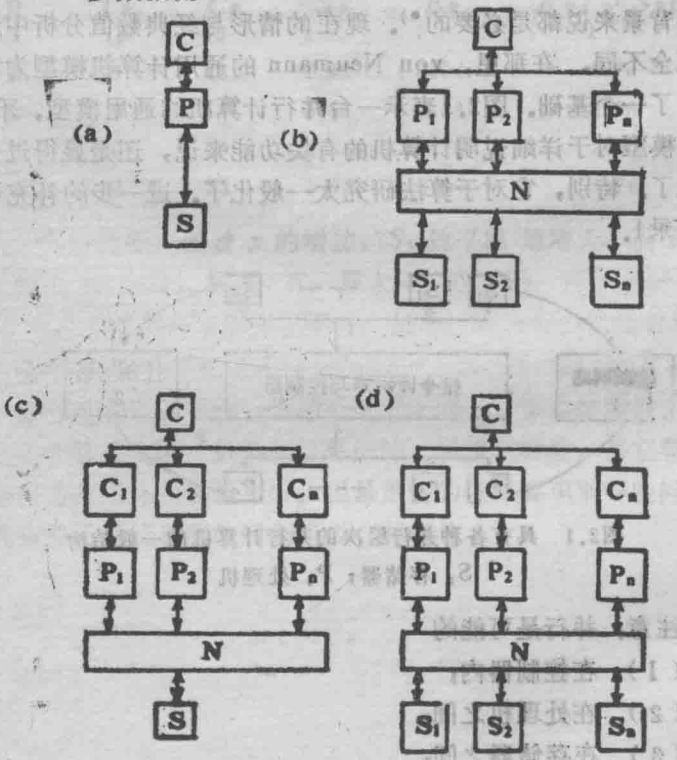

图2.2　计算机的分类。(a)SISD，(b)SIMD，(c)MISD，(d)MIMD。

（C：控制器；P：处理机；N：数据调整网络；S：存储器。）

因此，下列四种组合是可能的：

SISD　这是经典的 von Neumann 模型。单指令流对单数据流进行操作。

SIMD　这是包含阵列处理机和流水线处理机在内的一类机器。所有的处理机解释同样的指令并且对不同的数据执行之。这类机器由于其构成简单，可以具有大量的处理器。例如 ICL/DAP 有 4096 个处理器。

MIMD 这是 SIMD 的多处理机形式。各台处理机解释不同的指令并且对不同的数据进行操作。鉴于由此造成的复杂性，处理机的数目通常比较少就不足为奇了。到笔者写这一点时为止也只达到16台（DENELCOR HEP）。

MISD （处理机链）可以证明（见附录1），这种类型与 SISD 等价。因此，它在本书中就无关紧要了。

这些机器的类型通过图 2.2 作了说明。

2.1 SIMD 处理机

根据其结构和功能化的方法，这种类型的机器又可分为下面几种类型：

(i) 阵列处理机：例如 Burroughs 公司的 ILLIAC IV。

(ii) 流水线处理机：也称为向量处理机，例如 CDC STAR—100，Texas Instruments 公司的 ASC（先进的科学计算机），CRAY—1，CDC7600，IBM360/91，Manchester 大学的 MU5。

(iii) 关联处理机：例如 STARAN Ⅳ（Goodyear 宇航中心）。

采用 SIMD 处理机的算法，典型地出现在下列各类有关问题之中：