国家八六三计划资助项目

唐策善 梁维发 编著

并行图论算法

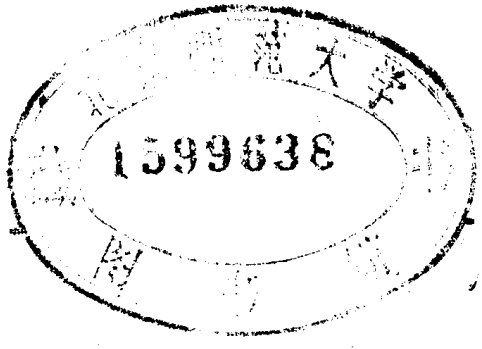
SIMD
MIMD
VLSI

中国科学技术大学出版社

并行图论算法

唐策善　梁维发　　编著

国家八六三计划资助项目



中国科学技术大学出版社

1991・合肥

内 容 简 介

本书反映了并行处理应用于图论领域研究的最新进展．主要内容包括：并行计算模型；图的搜索、求连通分支、最小生成树、最短路径等一些基本图论问题的并行算法和分布式算法；求基本回路、双连通分支、关节点和桥、欧拉路径和哈密顿路径，AOE 网和最大流，极大独立集和图着色等问题的并行和分布式算法；并行矩阵乘法在图论中的应用；组合搜索的并行算法；以及神经网络在图论中的应用．

读者对象：计算机及其应用专业、应用数学专业研究生和高年级学生，以及从事计算机、应用数学、网络工程、最优化设计等领域的科学研究和工程设计的科技工作者．

并行图论算法

唐策善　梁维发　编著

*

中国科学技术大学出版社出版

(安徽省合肥市金寨路 96 号，邮政编码：230026)

中国科学技术大学印刷厂印刷

安徽省新华书店发行

*

开本：787×1092／16　印张：　20.25　字数：　480　千

1991 年 10 月第 1 版　1991 年 10 月第 1 次印刷

印数：1－ 3000　册

ISBN7-312-00316-8/TP・38

[皖]第 08 号　定价：11.0 元

前　言

通常，图是由顶点集合和边集合组成的，用它表示的问题就是所谓的图论问题．它广泛地出现在计算机科学、信息科学、人工智能、网络理论、系统工程、运筹学、控制论、物理学、化学、心理学、语言学、经济管理等领域，因而研究图论问题及其解法具有极其重要的理论和实际意义。

很多图论问题表达容易，但求解却很困难，常常需要花费大量的计算时间和存贮空间．现已证明相当多的图论问题是 NP−完全的，即当问题的规模足够大时，往往出现"指数爆炸"现象，这是传统的顺序计算机无法求解的。随着 VLSI 技术的发展和并行计算机的出现，为人们快速求解图论问题提供了一条新的途径。运用多处理机系统来求解图论问题，在科学上和工程应用中发挥了重要的作用。

形式地讲，并行算法是一些可同时执行的诸进程的集合，这些进程相互作用和协同动作，从而达到对给定问题的求解。读者将会看到：并行算法和计算机结构密切相关，不同结构的并行计算机将会导致不同风格的并行算法。但是，本书不可能研究所有具体的并行计算机上的并行图论算法，只能从现有的并行计算机中抽象出若干典型的并行计算模型，并围绕这些模型讨论各种图论问题的并行算法。

本书描述在多处理机系统上协同求解图论问题的并行和分布式算法，这些算法并不完全是现行的顺序图论算法简单的推广和扩充。事实上，有的图论问题可以从顺序算法转换为相应计算模型上的有效并行算法；有的图论问题则不然，必须从问题本身出发，结合并行计算模型，研究和设计新的并行算法。在各种计算模型上，如何设计和分析并行图论算法，是本书写作的宗旨，我们将通过众多图论问题并行算法的讨论，向读者介绍若干基本的研究方法和技术。

最近十余年来，并行处理应用于图论领域的研究异常活跃，每年都涌现出大量的学术论文和研究报告。在这浩如烟海的文献中，摄取哪些材料作为本书的基本内容，才能反映这一领域的最新成果和发展趋势，是我们的努力方向。作者通过对计算机系研究生、高年级大学生和青年教师讲授本书部分内容的教学实践，以及从事这一领域研究工作的切身体会，编著成书，奉献给广大读者。

本书侧重于非数值计算方面的内容。全书共分十四章。前两章是并行算法的基础，包括并行计算机与并行计算模型；并行算法的度量与设计技术。第三章到第六章集中介绍基本图论问题的并行和分布式算法。它们是：图的搜索；求图的连通分支；计算加权图的最小生成树；以及找图的最短路径等。第七章讨论并行矩阵乘法及其在图论算法中的应用。第八章到第十二章，叙述了图论的基本性质和一些较难的图论问题的并行和分布式算法。内容包括：无向图的基本回路、双连通分支、关节点和桥；欧拉路径和哈密顿回路；AOE 网和最大流；极大独立集和图的着色等问题的算法。第十三章讨论了用状态空间树表示的组合搜索问题的并行算法；最后一章研究了神经网络在图论中的应用。每章的结尾还给出了小结和参考文献，简要概述了该章的主要内容，列出了有关的最新进展，以便读

者进一步研究时参考.

本书前六章和第十三章由唐策善提供初稿，第七章至第十二章由梁维发提供初稿，第十四章由唐锡南提供初稿．最后，经唐策善统一修改后定稿．

本书是并行算法丛书之一，受国家 863 计划项目资助．

我们在撰写中，曾直接或间接引用了许多专家、学者的文献；陈国良教授对本书的编写和出版非常关心，仔细审阅了全书内容，提出过许多宝贵意见；唐锡南付出了辛勤的劳动，除了提供第十四章初稿外，对其它章节亦提出了不少有益的建议；唐锡晋帮助整理了本书的部分章节；陈一栋、贾南在毕业实习过程中为本书第十三章做了部分工作；本校教务处激光照排中心的同志们为本书的计算机录入和排版做了大量的工作，作者谨此一并致以诚挚的谢意．

当国内外系统、深入地阐述这方面的教材或专著尚属鲜见的时候，我们为能向读者奉献此书而感到高兴；同时，也深感水平所限，书中缺点和错误在所难免，恳请广大读者批评指正．

作　　者

1991 年 5 月于中国科学技术大学

目　录

前　言 …… (Ⅰ)
第一章　并行计算机与并行计算模型 …… (1)
1.1　并行计算机及其分类 …… (1)
1.1.1　并行计算机介绍 …… (2)
1.1.2　并行计算机分类 …… (4)
1.2　并行计算模型 …… (4)
1.2.1　SIMD 共享存贮模型 …… (6)
1.2.2　SIMD 互连网络模型 …… (7)
1.2.3　MIMD 并行计算模型 …… (13)
1.3　小结 …… (14)
参考文献 …… (14)
第二章　并行算法的度量与设计技术 …… (16)
2.1　并行算法的基本概念 …… (16)
2.2　MIMD 机器上的算法 …… (16)
2.3　并行算法的度量标准 …… (17)
2.3.1　算法复杂性的基本概念 …… (17)
2.3.2　并行算法的复杂性度量 …… (18)
2.3.3　并行算法的性能评价 …… (19)
2.4　并行算法的表示及约定 …… (20)
2.5　并行算法的设计技术 …… (21)
2.5.1　几种基本的设计技术 …… (21)
2.5.2　设计并行算法应注意的几个问题 …… (24)
2.6　小结 …… (25)
参考文献 …… (25)
第三章　图的搜索 …… (27)
3.1　图的并行搜索 …… (27)
3.1.1　算法的基本原理 …… (27)
3.1.2　p一深度优先搜索 …… (27)
3.1.3　p一宽深优先搜索 …… (28)
3.1.4　p一宽度优先搜索 …… (28)
3.2　图的分布式搜索 …… (30)
3.2.1　纯遍历搜索算法 …… (30)
3.2.2　深度优先搜索算法 …… (31)
3.2.3　改进的深度优先搜索算法 …… (33)

3.2.4 宽度优先搜索算法 …… (35)
3.2.5 改进的宽度优先搜索算法 …… (37)
3.3 小结 …… (41)
参考文献 …… (42)
第四章 求连通分支的并行算法 …… (43)
4.1 传递闭包法 …… (43)
4.2 顶点倒塌法 …… (46)
4.2.1 算法的基本原理 …… (46)
4.2.2 算法的形式化描述 …… (46)
4.2.3 算法的正确性证明 …… (47)
4.2.4 算法的复杂性分析 …… (49)
4.3 最优的连通分支算法 …… (51)
4.4 稀疏图的连通分支算法 …… (54)
4.5 一维阵列上的连通分支算法 …… (54)
4.6 二维网孔上的连通分支算法 …… (56)
4.7 小结 …… (59)
参考文献 …… (61)
第五章 最小生成树的并行算法 …… (63)
5.1 Sollin 算法的并行化 …… (63)
5.2 树机上的 MST 算法 …… (65)
5.3 二维网孔上的 MST 算法 …… (68)
5.3.1 算法的基本原理 …… (68)
5.3.2 算法的非形式化描述 …… (68)
5.3.3 算法的复杂性分析 …… (71)
5.4 MST 的更新算法 …… (71)
5.4.1 基本概念 …… (71)
5.4.2 顶点更新的 MST 算法 …… (74)
5.4.3 边更新的 MST 算法 …… (75)
5.5 MIMD 共享存贮模型上的 MST 算法 …… (77)
5.6 分布式 MST 算法 …… (78)
5.6.1 算法的基本原理 …… (79)
5.6.2 算法的非形式化描述 …… (79)
5.6.3 算法的复杂性分析及正确性证明 …… (81)
5.6.4 其它改进的分布式 MST 算法 …… (82)
5.7 小结 …… (82)
参考文献 …… (84)
第六章 最短路径的并行算法 …… (87)
6.1 单源最短路径算法 …… (87)
6.2 所有顶点对的最短路径算法 …… (89)

6.3 二维网孔上的最短路径算法 …… (91)
6.4 MIMD 共享存贮模型上的最短路径算法 …… (92)
6.4.1 算法的基本原理 …… (93)
6.4.2 算法的形式化描述 …… (94)
6.5 分布式单源最短路径算法 …… (98)
6.5.1 算法的基本原理 …… (98)
6.5.2 算法的形式化描述 …… (99)
6.5.3 算法的正确性证明 …… (101)
6.6 小结 …… (102)
参考文献 …… (104)
第七章 矩阵乘法及其在图论算法中的应用 …… (106)
7.1 矩阵乘法的一个简单并行算法 …… (106)
7.2 二维网孔上矩阵乘法的下界 …… (107)
7.3 二维网孔上的矩阵乘法算法 …… (108)
7.4 超立方上的矩阵乘法算法 …… (109)
7.5 洗牌网络上的矩阵乘法算法 …… (112)
7.6 MIMD 共享存模型上的矩阵乘法算法 …… (114)
7.7 矩阵乘法在图论算法中的应用 …… (117)
7.7.1 计算图的传递闭包 …… (117)
7.7.2 计算所有顶点对的最短路径 …… (117)
7.7.3 计算图的中值和中值长度 …… (118)
7.8 小结 …… (119)
参考文献 …… (120)
第八章 基本回路、关节点、双连通分支和桥的并行算法 …… (121)
8.1 逆树的一些基本性质 …… (121)
8.2 找图的基本回路算法 …… (122)
8.3 找图的双连通分支算法 …… (124)
8.3.1 算法的基本原理 …… (124)
8.3.2 算法的非形式化描述 …… (128)
8.4 用欧拉遍历技术求双连通分支算法 …… (130)
8.4.1 算法的基本原理 …… (130)
8.4.2 算法的非形式化描述 …… (130)
8.4.3 算法在 SIMD 共享存贮模型上的实现 …… (133)
8.5 桥的算法 …… (137)
8.5.1 桥的基本性质 …… (137)
8.5.2 算法的非形式化描述 …… (138)
8.5.3 二维网孔上的桥算法 …… (139)
8.6 双连通分支算法的应用——求图的关节点 …… (141)
8.7 小结 …… (142)

参考文献 …… (142)
第九章 欧拉图及哈密顿图的并行算法 …… (144)
9.1 找欧拉回路的算法 …… (144)
9.1.1 欧拉图的基本概念及性质 …… (144)
9.1.2 找有向欧拉回路的算法 …… (145)
9.2 在竞赛图中找哈密顿回路算法 …… (151)
9.2.1 竞赛图的一些基本性质 …… (151)
9.2.2 算法的基本原理 …… (153)
9.2.3 算法的形式化描述 …… (157)
9.2.4 算法的复杂性分析 …… (158)
9.3 小结 …… (159)
参考文献 …… (159)
第十章 流图的并行算法 …… (161)
10.1 共享存贮模型上的 AOE 网问题的并行算法 …… (161)
10.1.1 AOE 网的存在性测试算法 …… (161)
10.1.2 AOE 网的拓扑排序算法 …… (162)
10.1.3 AOE 网的关键路径算法 …… (165)
10.2 超立方和洗牌网络上的 AOE 网算法 …… (170)
10.2.1 超立方和洗牌网络上的拓扑排序算法 …… (170)
10.2.2 超立方和洗牌网络上的关键路径算法 …… (172)
10.3 AOE 网的分布式算法 …… (173)
10.3.1 算法的形式化描述 …… (174)
10.3.2 算法的正确性证明 …… (176)
10.4 最大流问题的并行算法 …… (177)
10.4.1 有向流网络的基本概念 …… (177)
10.4.2 最大流并行算法的高层描述 …… (178)
10.4.3 PS-树及其上的操作 …… (181)
10.4.4 最大流算法的并行实现 …… (183)
10.4.5 最大流算法的有效实现 …… (187)
10.4.6 算法的复杂性分析 …… (188)
10.5 最大流问题的分布式算法 …… (190)
10.5.1 最大流问题的一些基本概念 …… (190)
10.5.2 深度优先搜索的最大流问题的分布式算法 …… (191)
10.6 小结 …… (193)
参考文献 …… (194)
第十一章 极大独立集的并行算法 …… (195)
11.1 引言 …… (195)
11.2 概率算法的基本概念 …… (195)
11.3 极大独立集问题的简单并行算法 …… (197)

11.3.1 极大独立集问题的基本知识 …… (197)
11.3.2 极大独立集问题并行算法的高层描述 …… (198)
11.3.3 极大独立集问题的随机并行算法 …… (199)
11.3.4 随机并行算法的复杂性分析 …… (201)
11.3.5 极大独立集问题的并行确定性算法 …… (204)
11.4 有效的极大独立集并行算法 …… (210)
11.4.1 基本概念 …… (210)
11.4.2 算法的形式化描述 …… (211)
11.4.3 算法的复杂性分析 …… (218)
11.5 小结 …… (219)
参考文献 …… (220)
第十二章 图着色的并行算法 …… (221)
12.1 图的顶点着色的并行算法 …… (221)
12.1.1 常数度图的着色算法 …… (221)
12.1.2 常数度图着色算法的应用 …… (223)
12.1.3 平面图 5-着色并行算法 …… (225)
12.1.4 平面图 5-着色最优的并行算法 …… (229)
12.2 图的边着色的并行算法 …… (235)
12.2.1 树的边着色并行算法 …… (235)
12.2.2 d-路图和 d-路二分图的基本概念 …… (236)
12.2.3 2-路图的边着色并行算法 …… (237)
12.2.4 二次幂 d-路二分图边着色的并行算法 …… (239)
12.2.5 正整数 d-路二分图边着色的并行算法 …… (241)
12.2.6 多重图边着色的并行算法 …… (244)
12.3 小结 …… (245)
参考文献 …… (246)
第十三章 组合搜索 …… (247)
13.1 分治法 …… (248)
13.1.1 SIMD 模型的分治算法 …… (248)
13.1.2 分治法在 MIMD 上的实现途径 …… (249)
13.1.3 分治算法的复杂性 …… (249)
13.2 分枝限界法 …… (251)
13.2.1 8-谜问题——一个例子 …… (251)
13.2.2 分枝限界方法 …… (253)
13.2.3 旅行商问题 …… (255)
13.2.4 并行分枝限界算法的异常情况 …… (259)
13.3 α-β 搜索 …… (261)
13.3.1 α-β 算法 …… (262)
13.3.2 改进的 α-β 搜索 …… (264)

13.3.3 并行搜索算法 …… (265)
13.4 小结 …… (269)
参考文献 …… (272)
第十四章 神经网络在图论问题中的应用 …… (274)
14.1 概述 …… (274)
14.1.1 生物神经元模型 …… (274)
14.1.2 神经网络的基本特征 …… (275)
14.2 Hopfield 模型和旅行商问题 …… (279)
14.2.1 引言 …… (279)
14.2.2 Hopfield 模型简介 …… (280)
14.2.3 HT 模型下的旅行商问题 …… (282)
14.2.4 HT 模型的改进 …… (284)
14.3 其它模型和旅行商问题 …… (286)
14.3.1 弹性网法 …… (286)
14.3.2 自组织映射 …… (288)
14.3.3 模拟退火 …… (291)
14.3.4 均场退火 …… (293)
14.4 应用举例 …… (297)
14.5 小结 …… (307)
参考文献 …… (307)
算法索引 …… (309)

第一章　并行计算机与并行计算模型

1.1 并行计算机及其分类

计算机发展的趋势是运算速度越来越快，存贮容量越来越大，软件越来越丰富，体系结构越来越完善，因此，处理能力越来越强。从早期的简单数据处理到近年来的复杂知识处理，都表明了这一点。计算机发展的历史表明，为了达到更高的处理性能，除了提高元器件的速度外，系统结构也必须不断改进，特别是当元器件速度达到极限（比如光速）时，后者将成为问题的焦点。计算机系统结构的改进，主要是围绕在同一时间间隔内增加操作量，即所谓的**并行处理**(Parallel Processing)技术。为了并行处理而设计的计算机系统称为**并行计算机**(Parallel Computer)。在并行计算机上求解问题的过程称为**并行计算**(Parallel Computing)。在并行计算机上设计求解给定问题的算法称之为**并行算法**(Parallel Algorithm)。

并行计算是一个比较年轻的领域：因为著名的阵列处理机 Illiac Ⅳ 1975 年才开始运行；第一台向量流水机 Cray-1 1976 年才交付使用；尤其是 VLSI 的发展，使得计算机的价格急剧下降，由多个处理器组成的并行计算机才流行起来。

并行计算机的发展主要是某些大型应用领域的要求：如气象预报，空气动力学，人工智能，卫星图象处理，核反应堆数据处理以及军事上的应用等。为了达到上述高性能要求，除了提高线路及器件速度外，主要是改进计算机的系统结构，例如：

(1) **引入 I／O 通道**：将费时的 I／O 操作交给 I／O 处理器（通道）去做，从而使 CPU 集中在计算上；

(2) **交叉存贮**：将存贮体分成多个模块，以达到并行存取和减少存取冲突为目的；

(3) **高速缓冲存贮器**：减少主存与处理器的数据交换时间，平滑这两者之间的数据流动速率；

(4) **指令先行**：一次取出适当多条将要执行的指令，使得取指令可以与指令译码同样快；

(5) **多功能单元**：在中央处理器中设置多个功能单元（加法、乘法器等）可以提高单一程序的吞吐量，缩短周转时间；

(6) **指令重叠和流水线技术**：在同一时间允许多条指令在不同的阶段按流水方式执行不同的操作；

(7) **向量处理技术**：一条向量指令可以同时处理 n 个分量，它们可以同时在各个处理器中进行运算，也可以连续地送入流水线中进行重叠处理；

(8) **多道程序设计**：同时把若干作业放在内存中，允许在同一时间有多道程序处于运行状态；

(9) **分时系统**：允许多个终端用户同时交互地使用同一台计算机；

(10) **数据驱动**：与冯·诺依曼计算机不同，它不是采用指令驱动操作，而是采用操作数（据）驱动操作，这样如有多个操作数准备就绪时，就可以彼此并行地执行而不受指令顺序执行的限制，从而可以充分开拓并行度。

1.1.1. 并行计算机介绍

首先我们介绍现有的一些典型并行计算机，以增加大家的感性认识。这里不去描述细节，仅仅讨论一下并行计算机的类型、结构特点和运行方式。

1. 阵列处理机(Array Processor)[5]

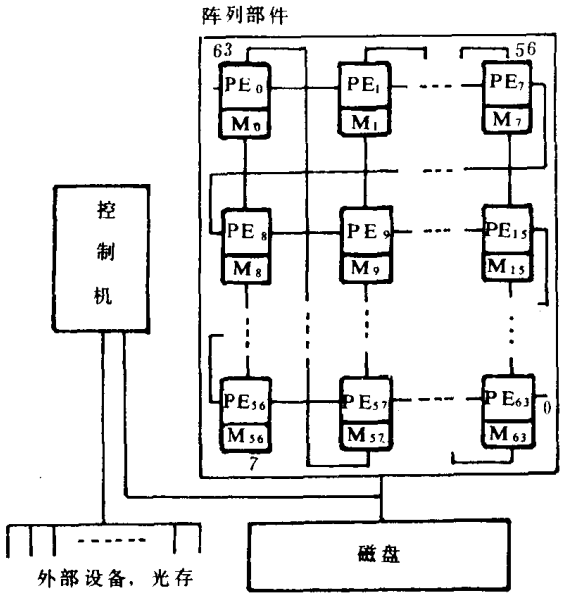


图 1.1 Illiac IV 机器框图

有名的 Illiac IV 就是阵列结构的。如图 1.1 所示，64 个 PE(Processing Element)各自带有局部存贮器 M，它们排成 8×8 的阵列。每个 PE 均可和它的上、下、左、右四个相邻的 PE 相连。所有 PE 在同一控制器控制下，按同一指令要求对不同的数据进行操作，从而达到操作级并行。

2. 流水线处理机(Pipeline Processor)

将生产流水线装配技术应用于计算机结构中，把计算机的运算部件或控制部件等装配成一些有序的子部件，利用功能部件分离与时间重叠办法，使每个被操作对象处在整个操作流程的不同功能部件中，且保持在不同的完成阶段，从而达到操作级的并行，这种结构的计算机称之为**流水线处理机**，亦叫**向量机**。流水线思想首先应用于指令的操作上；继之在功能划分的基础上，以流水线方式组成高速中央处理器；进而出现了在一台机器的中央处理器设置多条专用流水线，让它们并行工作或协同完成一些复合操作。这种系统对向量加工甚为有效，是目前解决大型数值计算问题巨型机的主要型式.

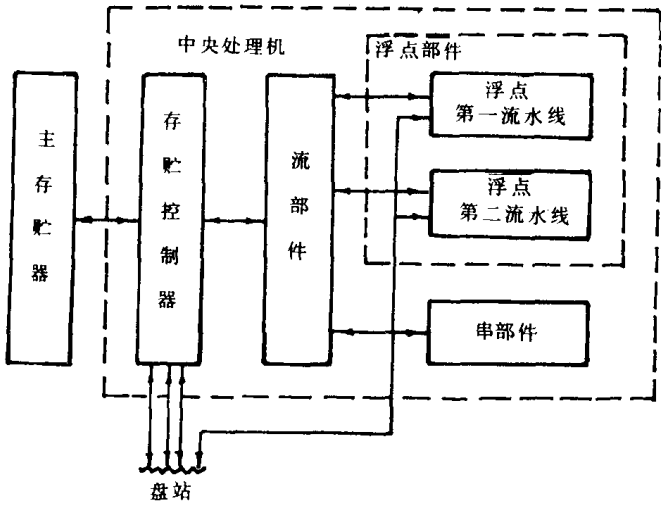


图 1.2 Star-100 机器框图

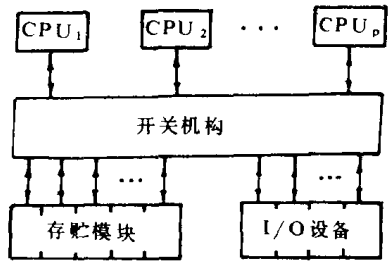


图 1.3 紧耦合的多处理机系统

Star-100 是典型的流水线向量处理机[6]。如图 1.2 所示，它由 CPU、主存、外围机及盘、站和外部设备组成。

3. 多处理机(Multiprocessor)和多计算机(Multicomputer)

多处理机（多处理器）和多计算机是由一些可编程的且可各自执行自己程序的多个处理器组成。其中，多处理机以各处理器共享公共存贮器为特征；而多计算机以各处理器经通信链路传递消息为特征。它们与阵列机的根本区别在于：阵列机中每个处理器只能执行中央处理器的指令，而后者可以执行自己的指令，这样可以达到指令级、任务级的并行。

如图 1.3 所示，在多处理机系统中，如果所有处理器都通过一个中央开关机构（如公共总线、交叉开关，包开关等）去访问全局共享存贮器，则这种形式的多处理机称之为**紧耦合多处理机**(Tightly Coupled Multiprocessor)系统，美国卡内基-梅隆大学的 $C._{mmp}$ 就是一个著名的紧耦合多处理机系统[7]。同紧耦合多机系统不同，若每个处理器各自有一个局部存贮器，所有的局部存贮器地址空间加在一起形成整个共享存贮空间，则这样的多处理机系统称为**松散耦合多处理机**(Loosely Coupled Multiprocessor)系统。因为松散耦合的多处理机没有集中的开关机构，所以可连接大量的处理器。卡内基一梅隆大学的 C_m^* 是另一个著名的松散耦合的多处理机系统[8]。如图 1.4 所示。

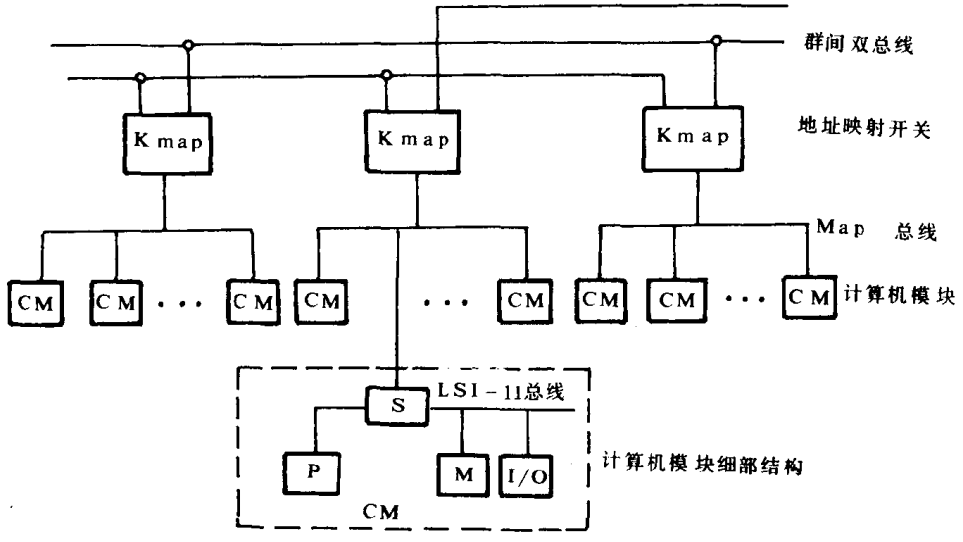


图 1.4 C_m^* 多处理机结构框图

C_m^* 采用了机群(Group)结构。目前已研制成的系统共有 50 台 DEC LSI-11 微型计算机，用三级总线(LSI-11 总线、Map 总线以及群间双总线)连接起来，其中 K_{map} 负责把各个计算模块所带有容量为 28K 字的局存组织在一起，形成统一的具有 28 位地址的虚拟存贮空间。

上面我们介绍的几种并行计算机基本上都还是在冯·诺依曼定义的框架内。目前已发展了许多非冯·诺依曼机，如**数据流计算机**(Data Flow Machine)，**归约计算机**(Reduction Machine)，**推理计算机**(Inference Machine)以及**神经网计算机**(Neural Networks Machine)等，虽然都是新一代计算机，但目前在这些机器上研究的并行算法尚未成熟，除了神经网机外，其它的不在介绍之列。

1.1.2 并行计算机分类

1. Flynn 分类法[9]

1966 年，M.J.Flynn 提出了著名的 Flynn 分类法。他按指令流及数据流将计算机系统分为四大类。

(1) **单指令流单数据流**：SISD(Single Instruction Stream Single Data Stream)计算机，这就是传统的串行计算机；

(2) **单指令流多数据流**：SIMD(Single Instruction Stream Multiple Data Stream)计算机，上一节所述的阵列机，流水线处理机均属于此类计算机。

(3) **多指令流单数据流**：MISD(Multiple Instruction Stream Single Data Stream)计算机。此类计算机是否存在尚有疑议。

(4) **多指令流多数据流**：MIMD(Multiple Instruction Stream Multiple Data Stream)计算机，上一节所述的多处理机和多计算机均属此类计算机。

2. Handler 分类法[10]

1977 年，Handler 根据计算机系统中流水线和并行度出现的级别，将一台计算机表示为三对整数。令 PCU 代表处理器控制单元，它相当于一个处理器或 CPU；ALU 代表算术逻辑运算单元，它相当于功能单元或处理单元；BLC 代表位一级电路。于是，按照 Handler 分类法，一台计算机可表示为：

$$T(C) = < K \times K', D \times D', W \times W' >$$

其中：K = PCU的数目；

K' = 能够流水执行的PCU数目；

D = 每个PCU所控制的ALU数目；

D' = 能够流水执行的ALU数目；

W = ALU或处理单元PE中的位数；

W' = 在所有ALU或单个PE中流水线段数。

上式中，如果任一对整数的第二个元素值为 1，则就略去它。在单一计算机系统中，符号"×"可用于描述不同种类处理器的连接。

如 CDC 6600 计算机，它有一个 CPU；ALU 有 10 个功能单元，它们都可以流水地执行；CDC 6600 字长 60 位；最多有 10 个外围 I/O 处理器，它们可以与 CPU 并行地工作，每个 I/O 处理器有一个字长 12 位 ALU，这样，

$$T(\text{CDC 6600}) = T\ (\text{中央处理器}) \times T(\text{I/O 处理器})$$
$$= < 1, 1 \times 10, 60 > \times < 10, 1, 12 >$$

然而，Handler 分类又过于依赖机器，目前大家都普遍遵循 Flynn 分类法。

1.2 并行计算模型

计算模型是算法的实现基础。并行算法的设计严格地依赖其计算模型。对同一问题而言，完全可能有许多不同的并行算法以适应在不同的模型上对这一问题的求解。

关于并行计算模型已有人做了大量的研究工作，在这里我们不打算给各种并行计算模型做一个系统的总结，仅仅介绍一些公认的计算模型。

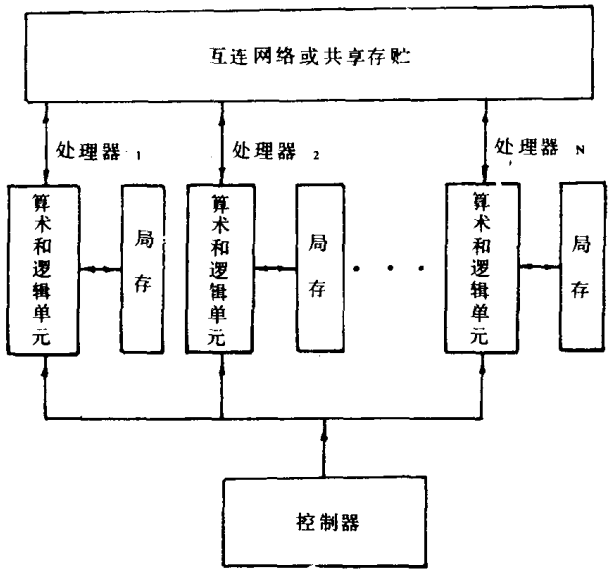


图 1.5 SIMD 计算模型

前面介绍的几种并行计算机，它们是并行算法的物质基础。但对算法的研究者而言，不能仅局限于某种具体并行机而研究并行算法，必须从算法角度，将各种并行机的基本特征加以理想化，抽象出所谓的并行计算模型，然后在此模型上研究和发展各种有效的并行算法。为此，我们在并行计算模型中对多处理器并行机作如下一些假定：

(1) **处理器数目假定**：无限和有限。前者是指算法中可使用随问题规模(大小或尺寸) n 成高阶多项式增长的处理器数；后者则是用 p 个处理器去求解问题($p \leqslant n$)；

(2) **时间假定**：每个处理器执行指令或数据操作的时间相同，且与处理器数目多少无关；

(3) **存取假定**：允许多个处理器同时读、写一个共享存贮单元，或不允许这样做；

(4) **指令集假定**：每个处理器具有普通计算机所拥有的指令集，如算术或逻辑运算、访内、输入／输出等指令。

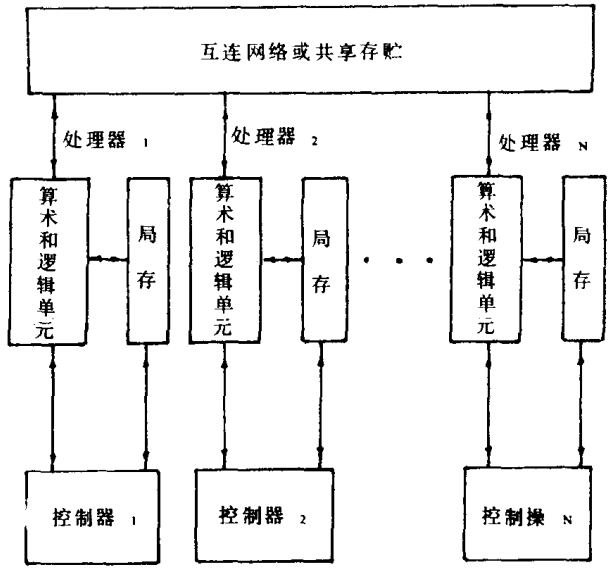


图 1.6 MIMD 计算模型

一般而言，并行计算模型可分为专用并行结构和通用并行结构。所谓**专用并行结构**是专门为有效解决某一类特定问题而设计的一种并行结构，如用于排序和选择的**比较器网络**(Comparator Network)[11]。基于 VLSI 技术的 **Systolic 阵列**[12]，这种模型是由 Kung 提出的，由于它具有结构规整，且以流水方式操作，很适合于数值计算。还有一种 VLSI 模型[13,14]，这种模型是指在一块芯片上装入成千上万个部件线路的计算模型，其中每个处理部件完成单一的逻辑功能，处理部件之间通过线路连接通讯，由于 VLSI 具有将算法渗透于硬件中的特性，近几年颇受人们重视。所谓**通用并行**

结构是为了解决广泛的应用问题而设计的一种并行结构。根据 Flynn 分类法，通用并行结构可分为 SIMD 及 MIMD 两大类。这两大类还可进一步细分为基于共享存贮的 SIMD 及 MIMD 计算模型和基于互连网络的 SIMD 及 MIMD 计算模型。它们分别如图 1.5 及图 1.6 所示。

1.2.1 SIMD 共享存贮模型

共享存贮模型是一种理想的计算模型，Fortune 等人[15]最初描述的共享存贮模型是：假定存在一个容量无限大的共享存贮器——**并行随机存取机器** PRAM(Parallel Random Access Machine)，同时有**有限**(Bound)或**无限**(Unbound)个功能相同的处理器，每个处理器拥有简单的算术运算和逻辑判断功能。在任何时刻，任何一个处理器均可通过共享存贮器的共享单元同其它任何处理器互相交换数据，但根据处理器对共享单元存取的不同约束条件又进一步分为：

(1) **不允许同时读和同时写**(Exclusive-Read and Exclusive-Write)。换句话说，每次仅允许一个处理器读或写某一个共享单元。这种计算模型简记为 SIMD-EREW PRAM；

(2) **允许同时读，但不允许同时写**(Concurrent-Read and Exclusive-Write)。换句话说，每次允许任意多个处理器同时读一个共享单元的内容，但每次仅允许一个处理器向某个共享单元写内容。这种计算模型简记为 SIMD-CREW PRAM；

(3) **允许同时读和写**(Concurrent-Read and Concurrent-Write)，即每次允许任意多个处理器同时读和同时写同一个共享存贮单元。这种计算模型简记为 SIMD-CRCW PRAM。

上述三种计算模型，根据它们对共享存贮单元的存取约束的强弱又分为最弱的计算模型 SIMD-EREW PRAM 和最强的计算模型 SIMD-CRCW PRAM。Snir 已证明了 SIMD-EREW PRAM 严格弱于 SIMD-CREW PRAM[16]。Cook 等人则又证明了 SIMD-CREW PRAM 严格弱于 SIMD-CRCW PRAM[17]。因此，若 T_M 表示一并行算法在一并行计算模型 M 上的时间，则

$$T_{EREW} \geqslant T_{CREW} \geqslant T_{CRCW}$$

在 SIMD-CRCW PRAM 上，允许许多处理器同时向一共享单元写内容，虽然这是不现实的，为了解决这种"写"引起的冲突。Kucera 曾提出三种解决"写冲突"的方法[18]，它们分别是：

(1) 仅写"1"的处理器写成功；

(2) 写入相同内容的处理器写成功；

(3) 优先权最高的处理器写成功。

而且他还证明了在以处理器为代价的前提下，这三种解决"写冲突"方法的时间复杂性是一致的。最近 Fich 等人更进一步分析了在处理器数及共享单元数目都受限制的情况下，这三种解决"写冲突"之间的关系[19]。

1979 年，Eckstein 曾采用二叉树方法来解决读写冲突问题[20]。为了解决读引起的冲突，只允许一个处理器从共享单元取内容，然后通过树向所有其它处理器广播此数据。为了解决写冲突，将树作为一种**竞赛**(Tournament)机构，确保仅有一个处理器写内容，这样，一个具有时