并行算法的设计与分析

陈国良

高等教育出版社

并行算法的设计与分析

陈 国 良

高等教育出版社

(京)112 号

内 容 提 要

本书比较系统、全面地讨论了各种专用和通用并行计算模型上的算法的设计和分析方法。书中以并行计算模型为线索，强调算法、结构和模型三者之间的密切关系，着重介绍了各种最基本、常用和典型的并行算法，同时也力图反映本学科的最新成就和发展趋势。

全书共分二十章，主要内容包括：并行算法基础；并行算法的基本设计技术；各种计算模型上的计算机领域中诸多常用计算问题的并行算法的设计和分析方法；最后还讨论了各种并行计算模型的能力、限制及其等价性以及与并行计算有关的 NC 理论问题。

本书内容丰富，取材新颖，叙述清晰，逻辑性强。可作为高等学校计算机有关专业的高年级学生和研究生的选修教材；也可供从事计算机科学理论和算法研究的科技人员阅读参考。

并行算法的设计与分析

陈 国 良

高等教育出版社出版

新华书店总店科技发行所发行

河北省香河县印刷厂印装

开本 787×1092 1/16 印张 33 字数 757 000

1994 年 5 月第 1 版 1994 年 5 月第 1 次印刷

印数 0 001— 2 135

ISBN 7-04 004612-1/TP·125

定价 12.60 元

前　言

最近几年来，由于元器件极限速度的限制，使得高性能计算机的研究热点主要集中在计算并行度的开发上，这样便促进了并行算法研究的深入和迅速发展．在此情况下，尽管目前并行算法的理论体系仍处于发展阶段，但其很多基本内容，研究方法已渐趋稳定和成熟，从而有可能将并行算法这门学科从学术论坛上搬上大学的讲台．作者从1984年以来已为计算机系的研究生和高年级学生讲授此课程多次，每次讲稿都有较大的变化，内容都不断地充实和完善．在此基础上，1992年又广泛地征求了国内同行专家的意见，同时也及时参照了国外最新同类教材，最后报请国家教育委员会高等学校计算机科学教学指导委员会审定，确定了该课程的教学大纲和书稿的基本内容．

按照审定的内容，本课程是属于算法研究的三个层次(并行算法理论，并行算法的设计和分析，并行算法的实现)的中间层次，即它不去重点研究算法理论中的某些基本问题，而是着重研究可有效并行求解问题类(即用多项式数目的处理器可在多对数时间内求解的NC-类问题)的算法的设计和分析方法，而且也略去了并行算法的具体实现环节(如并行语言、编译、执行环境与工具等)．同时本课程也是遵循计算机学科中有关算法研究的经典内容进行组织的，它严格区别于并行数值计算与分析课程(这些课程主要是研究数学计算原理、方法、精度与稳定性等问题)．所以对于本课程所涉及的一些数值计算问题，乃是按照上述原则，直接根据数值分析中的数学原理，密切结合体系结构和计算模型，详细讨论其有效并行算法的设计和分析的方法．

全书共分二十章，大体上可分为三大部分：第一部分讲述并行算法的基础知识及其基本设计技术(第一、二章)，包括并行机分类及处理器互连网络、并行计算模型、并行算法的定义、分类、表达和度量以及并行算法的基本设计方法等；第二部分讨论各种专用和通用并行计算模型上的计算机领域中诸多常用计算问题(诸如排序，选择，归并，搜索，组合运算，模式串匹配，表达式求值，语言识别和语法分析，矩阵运算，数值求解，FFT变换，计算几何，图论问题，组合搜索等)的确定性并行算法(第三章至第十七章)和非确定性并行算法(第十八章)的设计和分析方法；第三部分涉及到算法理论中的一些问题(第十九、二十章)，主要包括VLSI计算理论和并行计算模型的能力、限制及其等价性以及与并行计算有关的NC-理论问题．书中取材内容具有相当的广度，基本上包含了并行算法学科中主要的研究内容和主要的研究方面．

国家教育委员会已把并行算法的设计和分析列为高等学校计算机专业的专业选修课．学生应在学过数据结构、计算机体系结构、图论和算法设计与分析等课程之后学习本课程．全书的内容可根据不同的教学学时的规定和不同的读者对象进行不同的组合．作为必须讲授和最低60学时的教学要求，建议讲授章节及其相应的学时分配如下：

第一章(2学时)：1.1节～1.3节；　　第二章(2学时)：2.2节～2.4节；

第三章(2学时):3.1节,3.2节;　　第四章(4学时):4.1节, 4.4节;
4.5节,4.7节;

第五章(2学时):5.1节,5.3节;　　第六章(2学时):6.2节,6.4节;

第七章(2学时):7.1节,7.3节;　　第八章(2学时):8.2节,8.4节;

第九章(3学时):9.1节,~9.3节;　　第十章(2学时):10.1节.10.3节;

第十一章(2学时):11.1节,11.2节;　　第十二章(4学时):12.2~12.4节;

第十三章(3学时):13.1节,13.2节,13.4节;　　第十四章(4学时):14.1节~14.3节;

第十五章(4学时):15.2节~15.4节;　　第十六章(3学时):16.1节,16.4节,
16.6节;

第十七章(4学时):17.1节~17.4节;　　第十八章(4学时):18.2节,18.3节;
18.6节;

第十九章(4学时):19.1节,19.3节,19.5节.

书中其余部分的内容是作为学生阅读和参考用的.而带有*号的章节是任选的,它们或是预备性的知识(要求读者课前预习),或是深入研究性质的内容(鼓励面向研究的读者阅读).每章之后均附有适量的密切结合课文的习题;同时也开列了本章所引用的主要参考文献.全书最后附录了算法复杂界一览表,以便读者随时查阅.同时为了供读者不断补充某些计算问题的最新研究成果,最后还提供了一张备用表.

本书在撰写中,曾直接或间接地引用了许多专家、学者的文献,作者深表感谢;但也有很多作者的优秀论文未能被引用,作者亦深表歉意.在本书的内容取舍和章节安排方面,马绍汉教授、唐策善教授、康立山教授、张德富教授和张丽君教授等提出了很多宝贵意见;书稿付梓前,承蒙张德富教授进行了终审,提出了不少中肯的修改意见,作者谨此一并表示谢意.

中国科学技术大学的历届学生们,在听取我的讲授中,曾提出过很多可贵的建议,不断地丰富和充实了本书的内容.对于他们的辛勤劳动和良好的愿望,作者尤为感谢.

《并行算法的设计与分析》一书,在国内被推荐为高等学校试用教材,作者感到非常高兴,但也深深地意识到,限于作者水平,加之时间仓促,书中定有不少欠妥和错误之处,恳请诸位批评指正.

陈国良

于中国科学技术大学

1993年6月3日

目　录

第一章 导 论

本章首先介绍一下并行计算机体系结构中的若干问题及其分类；然后讨论各种常用的并行计算模型；最后介绍并行算法的一般概念．其中处理器互连网络是讨论并行计算模型所需的最基本的硬件知识，而并行计算模型是以后各章节所讨论的并行算法的共同基础．在介绍并行算法一般概念时也示例性地给出了几个并行算法，以期读者对并行算法的设计和分析有个初步的认识．

1.1 并行计算机的体系结构及分类

1.1.1 并行处理系统发展谱[1][2]

计算机发展的总趋势是速度越来越快，容量越来越大，体系结构越来越完善，软件越来越丰富，从而处理能力越来越强．计算机的发展历史表明，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，特别是前者的速度达到极限时，后者将变成了问题的焦点．计算机系统结构的改进，主要围绕着增加同一时间间隔内的操作数量，即所谓**并行处理技术**；为并行处理所设计的计算机统称之为**并行计算机**；在并行计算机上求解问题称之为**并行计算**；在并行计算机上实现求解问题的算法可称之为**并行算法**．

并行处理是一种有效的的强调开发计算过程中的**并发事件**(concurrent events)的信息处理方式．**并发性**(concurrency)包含**并行性**(parallelism)、**同时性**(simultaneity)和**流水线**(pipelining)．并行事件可在同一时间间隔内发生在多个资源中；同时事件可在同一时刻发生；流水线事件可在部分重叠时间内出现．开发问题中的并行性包括开发计算并行性，搜索并行性和逻辑并行性．其中计算并行性与算法的设计与分析有密切关系．

并行处理的发展主要是为满足某些含大量计算的应用领域的要求：例如气象预报，海洋学的研究，空气动力学计算，天体物理的计算模拟，卫星图象处理，核反应堆实验，生物系统模拟，社会经济模型构造，国防尖端技术，人工智能的研究等等．为了达到上述高性能的要求，于是就从单处理机开始沿着时间重叠，资源重复和资源共享这三大计算机结构学派生出不同的多处理机系统．图 1.1 示出了从**单处理机**(uniprocessor)和**多计算机**(multicomputer)两个极端向现代并行处理系统发展的过程．在单处理机内部增设流水线处理部件、多功能单元，关联存储器等只能开发指令级的并行性，而为了进一步提高并行度，必然导致向开发程序和任务级的并行性的多计算机系统发展．把多台计算机连接起来，相互配合，各尽其能，沿着功能专门化，多机群和网络化这三种基本技术措施向**异构型多处理系统**，**同构型多处理系统**和**分布处理系统**发展．

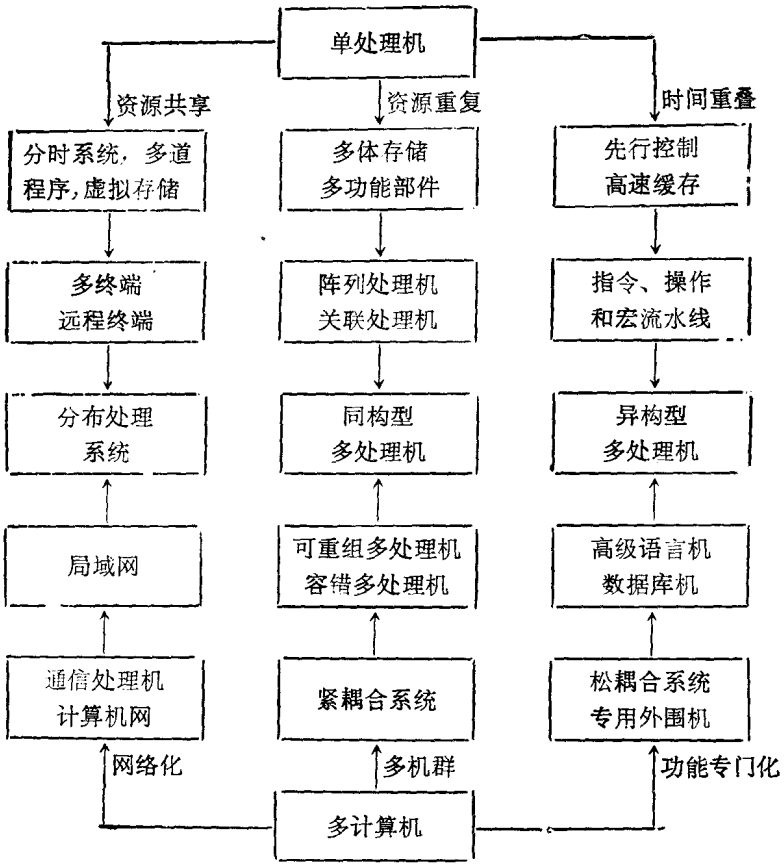


图 1.1 并行处理系统发展谱

1.1.2 并行计算机的分类

1. Flynn 分类法[3]

1966 年美国的 M. J. Flynn 按照指令流和数据流将计算机系统分为四类.

① **单指令流单数据流** SISD(Single Instruction stream Single Data stream)计算机. 这就是传统的顺序处理机，有时称单处理机.

② **单指令流多数据流** SIMD(Single Instruction stream Multiple Data stream) 计算机. 如阵列处理机，关联处理机和流水线处理机均属此类计算机.

③ **多指令流单数据流** MISD (Multiple Instruction stream Single Data stream) 计算机. 实际上以何种机器为代表尚存疑议.

④ **多指令流多数据流** MIMD(Multiple Instruction stream Multiple Data stream)计算机. 如多处理机和多计算机均属此类计算机.

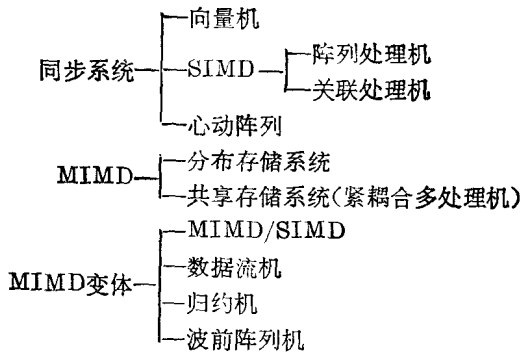
2. Händler 分类法[4]

1977 年 Händler 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，ALU 代表算术逻辑单元，BLC 代表位一级电路. 于是一台计算机可表示如下：

$$T(C)=\langle K\times K',\ D\times D',\ W\times W'\rangle \tag{1.1}$$

其中，K=PCU 数目，K'=能够执行流水线的 PCU 数目，D=每个 PCU 所控制的 ALU 数目，D'=能够执行流水线的 ALU 数目，W=ALU 或处理单元 PE 的位数，W'=所有 ALU 或单一 PE 中流水线段数.

3. 按体系结构分类[5]



① **同步结构**：同步并行计算机通过全局时钟、中央控制单元或向量单元控制器实现并发操作. 其中，**流水线向量机**(pipelined vector processors)直接支持向量和矩阵等高速运算；**阵列处理机**(array processors)是近代大规模并行机(massively parallel computers)的前驱的主要机型，直接支持大型科学计算；**关联处理机**(associative processors)是一种特殊的阵列机，直接支持非数值计算方面的诸如按内容存取、施行逻辑比较等操作；**心动阵列**(systolic array)是一种专用结构的计算机，采用同步流水线操作，能直接支持信号处理等计算.

② **MIMD 结构**：这种结构由各按局部数据独自执行指令的处理器所组成，直接支持可大量自治求解的问题计算. 其中，**分布存储系统**(distributed memory system)的各处理器节点的进程之间，通过**互连网络**(如环、网孔、树、超立方等)用**传递消息**(passing message)的方式进行同步；**共享存储系统**(shared memory system)的处理器之间通过公共总线、交叉开关或多级互连网络，使用一个均可访问的全局共享变量进行通信与同步，这样的系统也叫做**紧耦合多处理机**(tightly coupled multiprocessor).

③ **MIMD 变体结构**：其中，**MIMD/SIMD** 是一种混合结构，能直接支持图象处理 专家系统等方面的应用；**数据流结构**(data flow architecture)是一种数据驱动原理的新型计算机，能有效地开拓大规模并行度；**归约结构**(reduction architecture)是一种需求驱动原理的新型计算机，直接支持应用(函数)式程序设计语言；**波前阵列结构**(wavefront array architecture)是同步的心动结构与异步的数据流结构的结合，在处理稀疏矩阵运算方面特别有效.

*1.1.3 处理器互连网络

在并行机中如何将各个处理器或处理器与存储器互连起来是个关键问题. 本节引入几种典型的在并行计算模型中经常使用的互连网络[6].

1. 一维线性连接

一维连接又称为线性阵列(linear array)，简记之为 LA. 其连接方式是并行机中最简单、最基本的互连方式. 其中每个处理器只与其左右近邻相连，所以也叫作二近邻连接. 这种连接方式是心动结构的最基本形式. 当首尾处理器相连时可构成循环移位连接，在拓扑结构上等同于环.

约定处理器的数目 $n=2^m$ (m 为正整数) 令其地址的二进制表示式为 $P=p_{m-1}p_{m-2}\cdots p_0$ (下同)，则线性阵列的连接函数 LC 可定义如下：

$$\left.\begin{aligned} LC_{-1}(P)&=P-1,\ 1\leqslant P\leqslant n-1\\ LC_{+1}(P)&=P+1,\ 0\leqslant P\leqslant n-2\end{aligned}\right\} \tag{1.2}$$

其中，LC_{-1} 和 LC_{+1} 分别表示左和右连接.

2. 网孔连接

在网孔连接(mesh-connected)中，各处理器置于 q-维网格的格点上. 当 $q=2$ 时，就是四近邻连接，简记之为 MC^2，其中每个处理器只与其上下左右近邻相连，连接函数 MC^2 可定义如下：

$$\left.\begin{aligned} MC^2_{-}(P)&=P-1\\ MC^2_{+1}(P)&=P+1\\ MC^2_{-\sqrt{n}}(P)&=P-\sqrt{n}\\ MC^2_{+\sqrt{n}}(P)&=P+\sqrt{n}\end{aligned}\right\} \bmod n,\ 0\leqslant P\leqslant n-1 \tag{1.3}$$

上述连接函数也可用置换轮换表示之：

$$\left.\begin{aligned} MC^2_{-1}&=(n-1,\ \cdots,\ 1,\ 0)\\ MC^2_{+1}&=(0,\ 1,\ \cdots,\ n-1)\\ MC^2_{-\sqrt{n}}&=\prod_{j=0}^{\sqrt{n}-1}(j+n-\sqrt{n},\ \cdots,\ j+2\sqrt{n},\ j+\sqrt{n},\ j)\\ MC^2_{+\sqrt{n}}&=\prod_{j=0}^{\sqrt{n}-1}(j,\ j+\sqrt{n},\ j+2\sqrt{n},\ \cdots,\ j+n-\sqrt{n})\end{aligned}\right\} \tag{1.4}$$

例 1.1 对于如图 1.2 所示的 $n=16$ 的二维网孔，$MC^2_{-1}=(15,\ 14,\ \cdots,\ 1,\ 0)$，$MC^2_{+1}=(0,\ 1,\ \cdots,\ 14,\ 15)$，$MC^2_{-4}=(15,\ 11,\ 7,\ 3)(14,\ 10,\ 6,\ 2)(13,\ 9,\ 5,\ 1)(12,\ 8,\ 4,\ 0)$ 和 $MC^2_{+4}=(0,\ 4,\ 8,\ 12)(1,\ 5,\ 9,\ 13)(2,\ 6,\ 10,\ 14)(3,\ 7,\ 11,\ 15)$. □

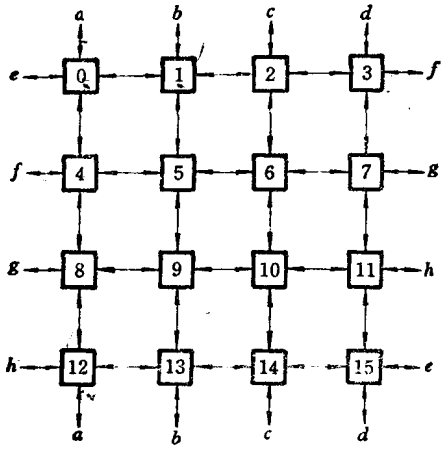


图 1.2 $n=16$ 的二维网孔

在 MC^2 模型上已设计出许多有效的并行算法，但 MC^2 的通信功能较差，在最坏情况下，任意两处理器之间的信息交换至少需要 $\sqrt{n}-1$ 步[11].

3. 树形连接

二叉树是大家非常熟悉的数据结构. **树连接** (tree-connected)，简记为 TC，除了根节点和叶节点外，每个内节点都与其父节点和两个子节点相连，因此二叉树连接可视为三近邻连接. 假定二叉树有 d 级(也称为 d 层，编号自根至叶为 0 到 $d-1$)，则共

有 $n=2^d-1$ 个节点. 图 1.3 示出了 $d=4$ 的二叉树. 树的典型用法是, 根和叶节点具有 I/O 功能, 且叶节点执行并行运算, 而内节点仅负责叶节点间的通信, 最长通信路径为树高. 显然, 根易成通信瓶颈. 为此可使用 **X-树**, 使得同级的兄弟节点彼此相连[7].

就通信功能而言, 树优于网孔. 两者的结合就产生了树网连接.

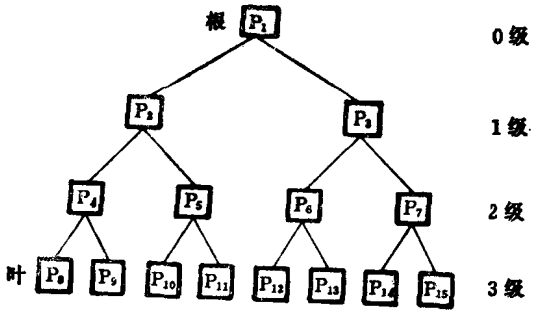


图 1.3 四级二叉树

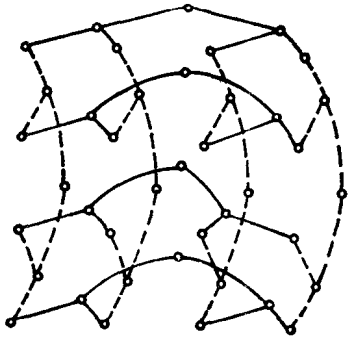


图 1.4 4×4 树网

4. 树网连接

树网(mesh of tree), 简记之为 MT. 如果去掉原网孔中的水平和垂直连接, 而各行各列代之以二叉树就可构成此种结构. 如此处理, 如图 1.4 所示, 原网孔格点上的处理器就变成了二叉树的叶节点处理器, 它们通过相应的行树和列树相互通信. 数据的入出经过树根. 这种结构, 既具有良好的通信功能, 又能继承网孔上很多有效算法, 同时也非常适合于 VLSI 集成化, 所以在其上已经发展了很多并行算法.

5. 金字塔连接

金字塔(pyramid)结构也是树与网孔的结合, 它在图象处理中颇为有用. 一个规模为 p 的金字塔是一棵高度为 $\log_4 p$ 的完全四叉树, 其中, $p=k^2$ 是二维网孔形式的金字塔之底. 金字塔的层数自底向上从 0 开始递增编号, 顶部只有一个节点, 第 k 层的每个节点与九个其他节点相连, 它们分别是 $k-1$ 层的一个父节点, $k+1$ 层的四个子节点(若存在的话)和同层四个近邻节点(若存在的话). 因此一个高为 $\log_4 p$ 的金字塔共有 $(4p-1)/3$ 个处理器节点. 图 1.5 是 $p=16$ 的金字塔.

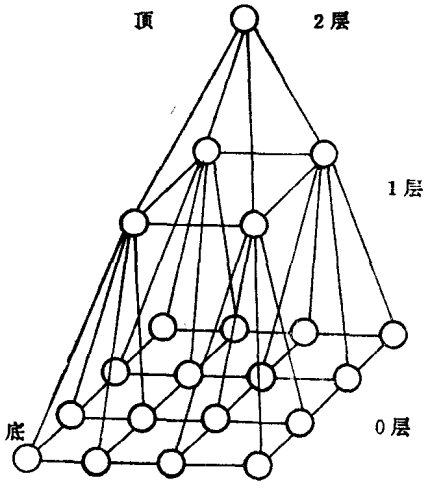


图 1.5 $p=16$ 的金字塔

6. 超立方连接

一个 $n=2^q$ 个节点的网络可以组织成一个 q-维**超立方**(hypercube)连接. 当 $q=3$ 时就是**立方连接**(cube-connected), 简记之为 CC. 一个 $q=4$ 的超立方连接示于图 1.6.

在超立方连接中, 如果顶点采用合适的标记方法, 使得每维中处理器编号的二进制表示仅一位不同, 则其连接函数 CC 可定义如下:

$$CC_i(p_{m-1}\cdots p_{i+1}p_ip_{i-1}\cdots p_0)=p_{m-1}\cdots p_{i+1}\bar{p}_ip_{i-1}\cdots p_0 \qquad (1.5$$

其中 $0 \leqslant i < m$, $\bar{p}_i = 1 - p_i$.

CC_i 也可用置换轮换表示之:

$$CC_i = \prod_{j=0}^{n-1} (j,\ CC_i(j)), \quad 0 \leqslant i < m \qquad (1.6)$$
(j 的第 i 位 $= 0$)

例 1.2 对于 $n=8$ 的立方连接, $CC_0=(0,1)(2,3)(4,5)(6,7)$, $CC_1=(0,2)(1,3)(4,6)(5,7)$ 和 $CC_2=(0,4)(1,5)(2,6)(3,7)$. □

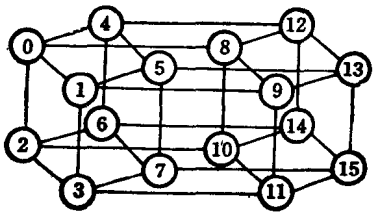


图 1.6 4-维超立方

立方连接是一种非常通用的互连结构. 但由于 q-维超立方中每个节点将与 q 个近邻相连, 当问题的规模增大时, 将导致工程技术实现的困难, 从而在某种意义上限制了超立方网络的应用.

7. 立方环连接

立方环(cube-connected-cycles), 简记之为 CCC, 是由 Preparata 等 1981 年提出的一种通用互连结构[8]. 它结合了环网与立方连接的优点及性质, 在立方网络的每个顶点代之以一个环, 故得名为带环的立方网络, 简称为立方环. 在此网络中, 每个顶点的度 $\leqslant 3$, 从而与问题的规模无关. 这一点对 VLSI 的实现极为重要.

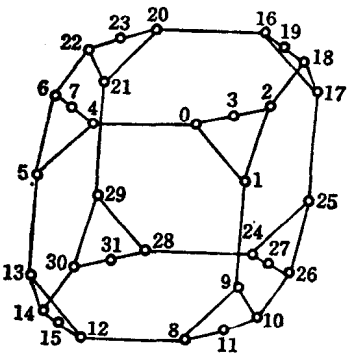


图 1.7 $n=32$ 的 CCC 结构

CCC 的构造如下: 令 $n=2^q$, 其中 $q=r+2^r$, r 取满足 $r+2^r \geqslant q$ 的最小整数. 在环内: 编号为 j 的顶点与编号为 $(j+1) \bmod 2^r$ 的顶点相连, 其中, $0 \leqslant j < n$; 在环间: 编号为 $(l2^r+j)$ 的顶点与编号为 $(l2^r+j+(1-2l_j)2^j)$ 的顶点相连, 其中, $0 \leqslant l < 2^{2^r}$, $0 \leqslant j < 2^r$, 且 l_j 是 l 的二进制表示的第 j 位(最右为第 0 位)的值, $(1-2l_j)2^j$ 表示将 l 的二进制第 j 位取反. 可见每个环是由 2^r 根连线连接而成的. 每个环的下标为 l(共有 2^{q-r} 个环). 若将每个环上的顶点集视为一个超顶点, 则 CCC 就蜕化为一个 $q-r$ 维的立方网络了. 图 1.7 示出了一个 $n=32(q=5, r=2)$ 的 CCC 网络. 它共有 2^{5-2} 个环, 每个环内共有 $2^r=4$ 个顶点.

若每个顶点上的处理器的编号 i 的二进制表示有 q 位, 它可分解为一对整数 (l, p), 其中, l 占 $q-r$ 位, p 占 r 位, 且满足 $l2^r+p=i$, $0 \leqslant i < n$. 于是每个处理器有三个与其他处理器相连的连接函数 $F(forward)$、$B(backward)$ 和 $L(lateral)$, 定义如下:

$$\left.\begin{array}{ll} \text{环内} & F(l,p) \text{连向} B(l,\ (p+1) \bmod 2^r) \\ & B(l,p) \text{连向} F(l,\ (p-1) \bmod 2^r) \\ \text{环间:} & L(l,p) \text{连向} F(l+s2^p,\ p) \end{array}\right\} \qquad (1.7)$$

其中, $s=1-2l_p$.

Galil 和 Paul 研究表明, CCC 能够有效地模拟许多网络, 所以是一种非常通用的互连网络[9].