可并行存取的共享 RAM 设计方案*

黄千平** 戴大为
武汉大学计算机学院,武汉 430072

摘要 给出一种有广泛适应性的可并行存取的共享 RAM 设计方案. 该方案由数据的无冲突存取存放方法及处理机(或处理单元)与共享 RAM 之间的互联网络结构两部分组成. 详细分析和论证了该方法和该网络的主要特点和性能. 并说明了该方案的实现方法与其他方案相比较的优点.

关键词 共享 RAM 无冲突存取 动态路径网 性能分析 实现方法

在各种具有共享 RAM 的并行计算机系统中,共享 RAM 往往成为系统运行速度的"瓶颈". 为了提高共享 RAM 的平均存取速度,人们设计了由多个可独立存取的存贮模块组成的多体交叉并行存贮器[1]. 然而这种存贮器要真正实现并行存取和充分发挥效率,必须解决好两个问题:一是数据的无冲突存取存放方法,这种方法要有广泛的适应性,以满足不同算法或同一算法不同运行时刻的需要;二是要设计一种无路径冲突的、处理机(或处理单元)与存贮模块之间的对应关系动态可变的互联网络,以使处理机能同时地和(对应关系)正确地通过网络存取各自的数据.

对这一课题的研究,在国际上已有 20 多年的历史,许多学者给出了他们的设计方案[2~6]. 其中,最早和较著名的是 Batcher 在文献[2]中提出的"散落式"存贮的思想,它不要求共享数据中相邻数据所在存储模块之间有某种线性连续关系,而是按照一种规则让数据"随机地散落"在各模块内. 他的这一思想后来由 Budnik 和 Kuck 作了进一步的发展和完善,并在文献[4]中给出了一种"线性斜交"方案,指出了在给定数据集合上实现无冲突存取的充分必要条件和确定存贮模块数最小值的方法.

进入 20 世纪 90 年代以来,随着 Cache 存贮器的广泛使用和分布式软件技术的发展,人们提出了分布式共享存贮器[7,8]和虚拟共享存贮器[9,10]的概念和实施方案,以求另辟途径解决无冲突存取问题. 但是,用 Cache 实现分布式共享存贮器容易产生不好解决的一致性问题[7],而用软件实现的虚拟共享存贮器在实现数据共享时的时空开销较大[9]. 因此,这些方案的总体效果还是不及集中式共享存贮器好.

本文将要给出的方案,由两部分组成:一是数据的无冲突存取存放方法,二是处理机(或处理单元)与共享 RAM 之间的互联网络结构.

2001-03-05 收稿,2001-06-04 收修改稿
* 湖北省自然科学基金资助课题(批准号:99j026)
** E-mail: hgp@whu.edu.cn

1 若干定义

不失一般性,本文假定系统内有 N 个处理机(或处理单元);共享 RAM 由 N 个可独立存取的存贮模块组成,每个模块有 N 个单元;$N=2^n$,n 是大于 2 的偶数;被存放和加工的数据用一个二维数组 $A=\{A(i,j)\mid i,\ j=0,1,\cdots,N-1\}$ 表示,它具有 N^2 个元素.

定义 1 令 NI 是 N 个整数 $0,1,2,\cdots,N-1$ 的集合,$A=\{A(i,j)\mid i,j\in NI\}$ 是 N 行、N 列的二维数组;令 $r=\sqrt{N}$,$N=2^n$,n 是大于 2 的偶数. 则 A 的第 i 行 $Row\ i$,第 i 列 $Col\ i$,第 i 个"方块" $Sb\ i$ 和第 i 个"分布块" $Db\ i$ 的定义如下

$$Row\ i=\{A(i,j)\mid 0\leqslant j\leqslant N-1\},$$

$$Col\ i=\{A(j,i)\mid 0\leqslant j\leqslant N-1\}$$

$$Sb\ i=\{A(\lfloor\frac{i}{r}\rfloor * r+a,(i \bmod r) * r+b)\mid 0\leqslant a,b\leqslant r-1\},$$

$$Db\ i=\{A(\lfloor\frac{i}{r}\rfloor+a * r,(i \bmod r)+b * r)\mid 0\leqslant a,b\leqslant r-1\}.$$

根据上述定义,可直接得到以下引理.

引理 1 设 $A(i,j)$ 是 A 中的一个元素,令 $x=(i \bmod r) * r+(j \bmod r)$,$y=\lfloor i/r\rfloor * r+\lfloor j/r\rfloor$,则在行主编号下,$A(i,j)$ 同时是 $Sb\ y$ 中的第 x 个元素和 $Db\ x$ 中的第 y 个元素.

定义 2 令 $x=x_0x_1\cdots x_{\frac{n}{2}-1}x_{\frac{n}{2}}\cdots x_{n-1}$ 是 $x\in NI$ 的二进制表示,则 x 的"半移"定义为,$Q(x)=Q(x_0x_1\cdots x_{\frac{n}{2}-1}x_{\frac{n}{2}}\cdots x_{n-1})=x_{\frac{n}{2}}x_{\frac{n}{2}+1}\cdots x_{n-1}x_0x_1\cdots x_{\frac{n}{2}-1}$,($x_0$ 是最高位,x_{n-1} 是最低位).

此定义表明,一个数的"半移",即将此数的二进制表示的高半部分和低半部分交换,它可以通过循环移位 $n/2$ 位实现.

定义 3 令 D 和 l 是 NI 中的任意两个整数,则 A 的某一"分割"定义为 $P_D^l(A)=\{A(Q(l\oplus\overline{Dk}),l\oplus Dk)\mid k\in NI\}$,且有以下引理.

引理 2 (a) 对于任一 $l\in NI$,有 $|P_D^l(A)|=N$;

(b) 对于所有 $l_1,l_2\in NI$ 和 $l_1\neq l_2$,有 $P_D^{l_1}(A)\cap P_D^{l_2}(A)=\varphi$;

(c) $\bigcup_{l=0}^{N-1}P_D^l(A)=A$.

基于定义 3,引理 2 的正确性是显然的.

上述定义及引理表明,D 给出一种对 A 的分割方法(例如,按行分割或按列分割)而 l 则给出在此分割方法下的某一具体"分割"(例如,某一行或某一列). 由于 D,l 均取值 $0,1,2,\cdots,N-1$,故共有 N 种不同的对 A 的分割方法,每种方法下均产生 N 个"分割",每个"分割"含有 N 个元素,它们之和完整地覆盖 A.

2 存储方案

在上节给出的假定和定义的前提下,本文给出的存储方案是:存放元素 $A(i,j)$ 于共享 RAM 中第 $Q(i)\oplus j$ 个模块的第 j 个单元内;$i,\ j\in NI$.

定理 1　本存储方案可保证无存放单元冲突，即 A 中的任意两个不同的元素不会存放在同一模块的同一单元内.

定理 2　对于任意给定的 $D, l \in NI$, $P_D^l(A)$ 中的元素均是可并行无冲突地存取的，并有以下推论.

推论 1　在本存储方案中，A 中的任一行均是可并行存取的.

推论 2　在本存储方案下，A 中的任一列均是可并行存取的.

推论 3　在本存储方案下，任一“方块”(定义 1)均是可并行存取的.

推论 4　在本存储方案下，任一“分布块”(定义 1)均是可并行存取的.

分别令 $D=[1^n], D=[0^n], D=[0^{\frac{n}{2}}1^{\frac{n}{2}}], D=[1^{\frac{n}{2}}0^{\frac{n}{2}}]$，可以很容易地证明上述 4 个推论.

以上仅是 $P_D^l(A)$ 的 4 个特例. 事实上，按照定义 3，可对 A 进行 N 种不同的分割，而且，根据定理 2，每种分割方法下的每一个“分割”中的所有元素都是可并行无冲突地存取的.

在上述方案下，存储器寻址，就是在给定的 D, l 下确定每个处理机 $PE_k(k \in NI)$ 应访问 RAM 中的哪个模块以及该模块中的哪个单元. 根据上述方案，元素 $A(Q(l \oplus \bar{D}k), l \oplus Dk)$ 存放在第 k 个模块中的第 $l \oplus Dk$ 个单元内. 这就是说，对给定的 D, l，每个存储模块 k 的局部地址都是 $l \oplus Dk$；至于每个 PE_k 应访问哪个模块，才能保证处理机编号与行主元素之间的对应，则由互联网络根据 l 决定.

3　互联网络

互联网络的功能是通过“动态地”实现处理机与存储模块之间的对应，以达到处理机与被处理数据之间的“行主对应”. 所谓“动态”，是说处理机与存储器模块之间的对应是随 l 取值的不同而变化的；而“行主对应”则是应将处理机 PE_k 与 A 的某一“分割”按行主编号的第 k 个元素相对应. 这种对应是许多并行算法所需要的.

3.1　网络实现

从原理上讲是对 *Omega* 网[6]的改进，不妨把它叫做“动态路径网”DPN. 它由 4×4 开关元件和级间连线组成. 在控制变量的控制下，可实现入端到出端之间的 8 种不同的转换. 若将 4 个端用一个两位的二进制数 $t = t_0t_1$ 表示，而控制变量用一个三位的二进制数表示：$CV = c_0c_1c_2$，则这 8 种不同的转换规则可由下式给出

$$\xi_{cv}(t) = \begin{cases} t_0 \oplus c_1 \# t_1 \oplus c_2, & c_0 = 0; \\ t_1 \oplus c_1 \# t_0 \oplus c_2, & c_0 = 1; \end{cases}$$

记号 $a \# b$ (a, b 均为一位二进制数)表示 a 与 b 的拼接. 如 $a=1, b=0$，则 $a \# b = 10$；$x \oplus a \# b \oplus y$ 表示将 x 与 $a \# b$ 的高位(左边位)按位加，将 y 与 $a \# b$ 的低位(右边位)按位加.

在 $N=2^n$ (n 为大于 2 的偶数)的情况下，动态路径网 DPN 总共有 $n/2$ 级，编号为 $0,1,2,\cdots,n/2-1$. 由于每个开关元件有 4 个入端(出端)，故每级有 $N/4=2^{n-2}$ 个开关元件，编号为 $0,1,2,\cdots 2^{n-2}-1$. 我们假定，同一级的诸开关均用同一控制变量控制，而且，在网络实现时，实际上是用 l 中的诸二进制位作各级的控制变量.

3.2　该网络级间连续规划

令开关元件 4 个端的编号用二进制数 $t = t_0t_1$ 表示，同一级各开关元件的编号用二进制数

$q = q_0q_1\cdots q_{i-1}q_i\cdots q_{n-3}$表示. 由于网络的每一级总共有 N 个入端(出端),故它们的编号用一个 n 位的二进制数 $z = z_0z_1\cdots z_{n-1}$表示. 于是,本网络的连线规则是:对第 $i(i=0,1,2,\cdots,n/2-1)$级而言,其上一级(若 $i=0$,则上一级为网络的输入)的输出端 $z = q_0q_1\cdots q_{i-1}t_0q_i\cdots q_{\frac{n}{2}+i-2}t_1q_{\frac{n}{2}+i-1}\cdots q_{n-3}$应连到本级(第 i 级)的第 $q = q_0q_1\cdots q_{i-1}q_i\cdots q_{n-3}$个开关元件的第 $t = t_0t_1$ 个输入端上.

定理 3 DPN 可实现双向映射 $k \Leftrightarrow Q(k)\oplus l; l,k\in NI$.

证明 由归纳法,令 $k = k_0k_1\cdots k_{n-1}$和 $l = l_0l_1\cdots l_{n-1}$分别是 k 和 l 的二进制表示,令 $CV = 1l_il_{\frac{n}{2}+i}$为网的第 i 级的控制变量,并规定同一级的诸开关元件用同一控制变量 CV 来控制.

对 $i(i=0,l\cdots,n-1)$进行归纳,即经过第 i 级之后,网络输入 k 将变为$(k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{\frac{n}{2}+i}\oplus l_i)k_{i+1}k_{i+2}\cdots k_{\frac{n}{2}-1}(k_0\oplus l_{\frac{n}{2}})(k_1\oplus l_{\frac{n}{2}+1})\cdots(k_i\oplus l_{\frac{n}{2}+i})k_{\frac{n}{2}+i+1}\cdots k_{n-1}$,对于所有的 $i=0,1,2,\cdots,n-1$成立. 于是,当 $i=n-1$ 时,网络输入 k 便变为

$$(k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{n-1}\oplus l_{\frac{n}{2}-1})(k_0\oplus l_{\frac{n}{2}})(k_1\oplus l_{\frac{n}{2}+1})\cdots(k_{\frac{n}{2}-1}\oplus l_{n-1}) = Q(k)\oplus l.$$

(1) 当 $i=0$,根据前述连线规则,$k = k_0k_1\cdots k_{\frac{n}{2}-1}k_{\frac{n}{2}}\cdots k_{n-1}$应连到网络第 0 级的第 $q = k_1k_2\cdots k_{\frac{n}{2}-1}k_{\frac{n}{2}+1}\cdots k_{n-1}$个开关的第 $t = k_0k_{\frac{n}{2}}$ 个输入端上,而此级的控制变量 $CV = 1l_0l_{\frac{n}{2}}$,根据上述转换规则,$t = k_0k_{\frac{n}{2}}$应转换到输出端 $\xi_{cv}(t) = k_{\frac{n}{2}}\oplus l_0 \# k_0\oplus l_{\frac{n}{2}}$也就是说,经过第 0 级之后,网络输入 k 已联到$(k_{\frac{n}{2}}\oplus l_0)k_1k_2\cdots k_{\frac{n}{2}-1}(k_0\oplus l_{\frac{n}{2}})k_{\frac{n}{2}+1}\cdots k_{n-1}$,即,归纳假设对 $i=0$ 是成立的.

(2) 假设 $i=j-1$ 时归纳假设是成立的,即,经过第 $j-1$ 级之后,网络输入 k 将连到$(k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{\frac{n}{2}+j-1}\oplus l_{j-1})k_j\cdots k_{\frac{n}{2}-1}(k_0\oplus l_{\frac{n}{2}})\cdots(k_{j-1}\oplus l_{\frac{n}{2}+j-1})k_{\frac{n}{2}+j}\cdots k_{n-1}$. 现要证明,当 $i=j$ 时,网络输入 k 将连到$(k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{\frac{n}{2}+j-1}\oplus l_{j-1})(k_{\frac{n}{2}+j}\oplus l_j)k_{j+1}\cdots k_{\frac{n}{2}-1}(k_0\oplus l_{\frac{n}{2}})(k_1\oplus l_{\frac{n}{2}+1})\cdots(k_j\oplus l_{\frac{n}{2}+j})k_{\frac{n}{2}+j+1}\cdots k_{n-1}$. 由于网络输入 k 经第 $j-1$ 级之后的输出端应连到第 j 级的第 $q = (k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{\frac{n}{2}+j-1}\oplus l_{j-1})k_{j+1}\cdots k_{\frac{n}{2}-1}(k_0\oplus l_{\frac{n}{2}})\cdots(k_{j-1}\oplus l_{\frac{n}{2}+j-1})k_{\frac{n}{2}+j+1}\cdots k_{n-1}$个开关元件的第 $t = k_jk_{\frac{n}{2}+j}$ 个输入端,而第 j 级的控制变量 $CV = 1l_jl_{\frac{n}{2}+j}$,于是,输入端 $t = k_jk_{\frac{n}{2}+j}$应转换到 $\xi_{cv}(t) = k_{\frac{n}{2}+j}\oplus l_j \# k_j\oplus l_{\frac{n}{2}+j}$,也就是说,经过第 j 级之后,网络输入 k 应连到$(k_{\frac{n}{2}}\oplus l_0)(k_{\frac{n}{2}+1}\oplus l_1)\cdots(k_{\frac{n}{2}+j}\oplus l_j)k_{j+1}\cdots k_{\frac{n}{2}-1}(k_0\oplus l_n)(k_1\oplus l_{\frac{n}{2}+1})\cdots(k_j\oplus l_{\frac{n}{2}+j})k_{\frac{n}{2}+j+1}\cdots k_{n-1}$. 于是,$k\to Q(k)+l$ 成立.

令 $l = Q(l)$,用上面同样的方法可证明映射 $Q(k)\oplus l\to k$. 证毕.

定理 4 DPN 可实现双向映射 $k \Leftrightarrow k\oplus l; k,l\in NI$.

4 并行存取的实现

定理 5 对任意的 $D,l\in NI$,系统允许 $PE_k(k\in NI)$存取 $P_D^l(A)$中的元素 $A(Q(Dl\oplus\bar{D}k),\bar{D}l\oplus Dk)$.

定理 6　对任意的 $D, l \in NI$，系统允许 $PE_k(k \in NI)$ 存取 $P_D^l(A)$ 中的元素 $A(Q(Dl \oplus \bar{D}Q(K)), \bar{D}l \oplus DQ(k))$.

推论 5　本方案允许：

(1) PE_k 并行地存取 A 中任意给定行的第 k 个元素($k \in NI$)；

(2) PE_k 并行地存取 A 中任意给定列的第 k 个元素($k \in NI$)；

(3) PE_k 并行地存取 A 中任意给定“方块”中的第 k 个行主元素($k \in NI$)；

(4) PE_k 并行地存取 A 中任意给定“分布块”中的第 k 个行主元素($k \in NI$).

分别令 $D=[1^n]$, $D=[0^n]$, $D=[0^{\frac{n}{2}}1^{\frac{n}{2}}]$, $D=[1^{\frac{n}{2}}0^{\frac{n}{2}}]$，应用定理 5，定理 6 和引理 1，可以很容易地证明上述 4 个结论.

上面的引理只是对 D 的 4 个特殊值所对应的 4 种情况作了说明. 事实上. 对于 D 的其他值，我们也可证明 PE_k 亦可按行主顺序并行存取 A 中的相应元素.

5　结束语

本文给出的方案，从基本思想上讲，还是 Batcher 的“散落式”原理，只是“散落”的方法更加巧妙和有效，因而其适应性和可行性都比较好. 与其他方案相比有以下一些优点：(1) 可以无冲突存取的数据集合的结构类型多，在数组为 $N \times N = N^2$ 的情况下，共有 N 种不同的分割方法，每种方法下有 N 个具体的“分割”，每个“分割”里的 N 个数据都是可并行存取的；(2) 寻径和寻址的方式都比较简单，均由 D, l 通过简单的运算获得；(3) 互联网络设计独特，不仅处理机和存贮模块之间的对应关系动态可变，而且在“行主对应”下无路径冲突；(4) 整个方案完全由硬件实现，不需要软件的辅助，因而延时短，速度快.

本系统在实际工作时，能自动地将处理机给出的二维地址转变为相应的 D, l：D 指明对 A 进行分割的方法，而 l 指明在此分割方法下的某一具体“分割”. 这种“指明”，是通过把 D 和 l 用于第 $k(k \in NI)$ 个存储模块的局部地址寻址($l \oplus Dk$)和把 l 用于 DPN 的控制变量实现的. 在 D, l 的作用下，一次可并行存取 N 个元素，处理机与其存取的元素之间有一种行主对应关系.

参　考　文　献

1 Hwang K, et al. Computer Architecture and Parallel Processing. McCrow-Hill Book Comp, 1984

2 Batcher K E. The multidimensional access memory in STARAN. IEEE Trans on Comp, 1977, 26(2): 247

3 Stenstrom P. Reducing contention in shared-memory multiprocessors. Computer, 1988, 21(11): 20

4 Budnik P, et al. The organization and use of parallel memories. IEEE Trans on Comp, 1991, 20(12): 1566

5 Padmanabhan K. Efficient architecture for data access in a shared memory hierarchy. Jour of Parallel and Distributed Computing, 1991, 11(4): 314

6 Kumar V P, et al. Augmented shuffle——exchange multistage interconnection networks. Computer, 1987, 20(6): 30

7 Fong C C F, et al. Distributed caching and broadcast in a wireless mobile computing environment. Computer, 1999, 42(6): 455

8 Martin K M, et al. Bounds and techniques for efficient redistribution of secret shares to new access structures. Computer, 1999, 42(8): 675

9 Hallnor E G, et al. A fully associative software-managed cache design. Computer Architecture News, 2000, 28(2): 107

10 Rixner S, et al. Memory access scheduling. Computer Architecture News, 2000, 28(2): 128