文章序号: 1001-2486 (2000) 02-0041-04

多维离散变换与小波的并行算法及其实现*

曾泳泓[1], 孟祥杰[2], 何丽君, 李晓梅[3]

(1. 国防科技大学, 长沙, 410073; 2. 装甲兵指挥学院, 北京; 3. 总装备部指挥技术学院, 北京)

摘　要: 讨论了多维 DCT 和多维 DWT 的并行行列算法和并行多项式变换算法, 并用 LogP 模型对算法进行了分析。在仔细分析一维和二维离散小波变换与小波包变换计算结构的基础上, 提出了它们的并行算法。算法只需相对较少的通信时间; 适合大规模并行机 (MPP) 和工作站或微机机群系统; 方法也适合信号处理中的各种塔式分解过程。用 Fortran 语言和 PVM 编制了算法的程序。在机群系统和大规模并行机上的实验表明, 算法是正确的且具有较高的加速比。

关键词: 并行算法; 离散变换; 小波; 塔式算法; 信号处理

分类号: TN911.7　　文献标识码: A

Parallel Algorithms For Discrete Transforms and Wavelet Transforms with their Applications

ZENG Yong-hong, MENG Xiang-jie, HE Li-jun, LI Xiao-mei

(College of science, National Univ of Defense Technology, Changsha 410073, China)

Abstract: The parallel row-column method and the parallel polynomial transform algorithm are discussed. The algorithms are analyzed with the LogP model. After detailed analysis of the structure of one-and two-dimensional discrete wavelet transform, the paper proposes parallel algorithms for them. The algorithms need relatively small amount of communication time and are suitable for MPP or workstation clusters. Programs are made for them. Experiments on MPPs show that the algorithms are correct and high speedups are achieved. The methods can also be used for any kind of pyramid a Algorithm in signal processing.

Key words: Parallel algorithm; Discrete Wavelet Transform; Pyramid Algorithm; Signal Processing

近年来, 并行与分布式信号处理成了国际上的研究热点。尤其是利用机群系统 (工作站集群或微机网络) 完成实时信号处理已成为一个主要发展方向, 发挥 MPP 或机群系统效率的关键在于设计合适的并行算法。目前小波分析与离散变换应用最成功的领域是信号处理。在信号处理中, 经常需要对信号进行离散变换或进行塔式分解, 对分解后得到的变换系数进行分析、处理、压缩。在计算机视觉、高清晰度电视(HDTV)以及可视电话等领域, 要对运动图象进行分析和处理, 通常称为多帧检测(Multi-Frame Detection, 简称 MFD), 这时需要用三维或更高维的离散变换。在信号与图像分析、地震信号处理、计算机视觉、语言合成与分析、信号的异性检测和去噪、生物医学工程以及复杂流体等方面, 小波变换有着广泛的应用[7~9]。在有些应用中, 实时性非常重要, 目前尚达不到要求。因此如何快速并行实现离散变换与小波变换是极其重要的。本文面向机群系统或 MPP 提出了计算离散变换、小波变换与小波包变换的并行算法, 并分析了算法的通信时间。编制了算法的 Fortran 程序, 并在基于 PVM 并行平台的微机机群系统上做了大量的数值实验, 证明了算法的正确性和并行效果。算法也可以用于信号处理中的各种塔式分解过程或其他领域。

1　多维 DCT 的并行算法及其实现

二维 DCT-Ⅱ为

* 基金项目: 国家自然科学基金资助项目 (19601012)
作者简介: 曾泳泓 (1962-), 男, 副教授, 博士, 现兼任新加坡南洋理工大学研究员。

$$X(k,l) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} x(n,m)\cos\frac{(2n+1)k\pi}{(2N)}\cos\frac{(2m+1)k\pi}{(2M)},$$
$$k = 0,1,\cdots,N-1;\ l = 0,1,\cdots,M-1.$$

令

$$y(k,m) = \sum_{n=0}^{N-1} x(n,m)\cos\frac{(2n+1)k\pi}{(2N)},$$

则有

$$X(k,l) = \sum_{m=0}^{M-1} y(k,m)\cos\frac{(2m+1)k\pi}{(2M)},$$

这相当于先对输入序列$x(n,m)$的每一列作长度为N的一维DCT-Ⅱ,得到$y(k,m)$;然后,对序列$y(k,m)$的每一行作长度为M的一维DCT-Ⅱ,得到$X(k,l)$。因此,要做N次长度为M的一维DCT-Ⅱ和M次长度为N的一维DCT-Ⅱ。

考虑二维DCT的行列算法的并行计算,显然M个长度为N的一维DCT-Ⅱ是完全独立的,因而可以并行计算,同样N个长度为M的一维DCT-Ⅱ也可以同时计算。但计算过程中必须在处理机之间交换数据。设处理机个数为P,为简单起见,设P/M, P/N。开始时可以将输入数据$x(n,m)$按列成块分配给各处理机,即每个处理机依次存放矩阵的M/P列。并行行列算法如下:

算法**1** 并行行列算法

第一步. 每个处理机对局部存储内的每一列数据进行一维DCT。得到的结果仍然为一个$N\times M$的矩阵$y = (y(n,m))$。每个处理机依次存放了该矩阵的M/P列。

第二步. 在处理机之间重新分配数据,目的是让处理机按行存放矩阵y的元素,每个处理机依次存放该矩阵的N/P行。为此,需要对矩阵y进行转置。

第三步. 每个处理机对局部存储内的矩阵y的每一行进行一维DCT。

若并行机是共享存储结构, 输入数据应放在共享存储器内, 这时算法不需进行第二步。在分布存储的并行机上, 算法的第二步需要仔细考虑。整个算法的数据通信主要在这一步。文献[1]中论述了在分布存储的并行机上的矩阵转置问题。算法的时间复杂性为

$$T_{re} = \frac{N}{P}M\log M\tau_1 + \frac{M}{P}N\log N\tau_1 + T_{\text{trans}}.$$

这里T_{trans}表示第二步所需的通信时间(参考文献[1, 5])。τ_1表示计算一次乘法和加法的时间。这个算法不难推广到多维DCT的计算。

下表给出在银河Ⅲ计算机上行列并行算法的计算时间。

表1 并行二维DCT算法在银河Ⅲ上的计算时间(秒)

Tab.1 The runtime for 2D-DCT on YH-Ⅱ computer

问题规模	4处理机		8处理机		16处理机	
	计算时间	加速比	计算时间	加速比	计算时间	加速比
$2^9\times2^{10}$	9.2634	2.5708	6.1643	3.8746	5.6005	4.2444
$2^{10}\times2^{10}$	19.3780	2.5941	10.3172	4.8720	6.9106	7.2807
$2^{12}\times2^8$	19.2038	2.6248	10.1300	4.9766	6.7504	7.4722

多维离散W变换(DWT)经过适当的变换也可以采用行列算法计算(参考文献[1, 3]),因而上节的并行算法也可以用于计算多维DWT。用多项式变换计算多维DWT是目前运算量最少的算法。作者在论文[4]中详细考虑了这个算法。由于多项式变换的并行计算在专著[1]中有详细论述,因此容易得到多维DWT的并行多项式变换算法,限于篇幅,此处不做论述。

2 离散小波变换的并行算法

为了节省篇幅,这里只考虑二维周期小波变换。设尺度系数为h_{k_1,k_2}、小波系数为$g^1_{k_1,k_2}, g^2_{k_1,k_2}, g^3_{k_1,k_2}$,对输入的二维离散信号$x = x_{k_1,k_2}(k_1 = 0,1,\cdots,N-1;\ k_2 = 0,1,\cdots,M-1)$,其二维周期离散小波变

换为

$$C_{j,m_1,m_2}=\sum_{k_1,k_2}h_{k_1,k_2}C_{j-1,<k_1+2m_1>,<k_2+2m_2>},\ D^1_{j,m_1,m_2}=\sum_{k_1,k_2}g^1_{k_1,k_2}C_{j-1,<k_1+2m_1>,<k_2+2m_2>},$$

$$D^2_{j,m_1,m_2}=\sum_{k_1,k_2}g^2_{k_1,k_2}C_{j-1,<k_1+2m_1>,<k_2+2m_2>},\ D^3_{j,m_1,m_2}=\sum_{k_1,k_2}g^3_{k_1,k_2}C_{j-1,<k_1+2m_1>,<k_2+2m_2>},$$

$$j=1,2,\cdots,r.$$

其中r 是分解的步数, 视实际需要而定。$C_{0,k_1,k_2}=x_{k_1,k_2}$.

实际应用中, 一般而言输入数据的尺寸很大, 而h,g^2 的非 0 项很少(集中在下标为 0 的附近) , 因此, 计算输出序列在点(m_1,m_2) 处的值只需要输入序列在点$(2m_1,2m_2)$ 附近点的值, 这说明计算基本上只依赖局部数据。若将数据适当地分成大小相等的若干块, 让每台处理机负责其中一块的分解, 则在处理块边界点的分解时需要处理机之间的数据交换。若分块时让数据在边界上有一定的重叠(根据h,g^i 的非 0 点的数目确定要重叠的数据的多少) , 则可以减少处理机之间需要交换的数据量。

为方便起见, 设N,M 为 2 的幂。设处理机台数为P。将输入数据$x_{k_1,k_2}(k_1=0,1,\cdots,N-1;\ k_2=0,1,\cdots,M-1)$ 按列块分配给P 台处理机, 处理机i 得到的数据为x_{k_1,k_2} 的第iM/P 列至第$(i+1)M/P-1$ 列, 让处理机i 负责C_{m_1,m_2} 和$D^1_{m_1,m_2},D^2_{m_1,m_2},D^3_{m_1,m_2}(m_2=iM/(2P),\cdots,(i+1)M/(2P)-1)$ 的计算, 则不难看出, 处理机i 基本上只要用到局部数据, 只有$N\times L/2$ 个点的计算要用到处理机$i+1$ 中的数据。这时做一步并行数据发送: 将处理机$i+1$ 中的前$L-1$ 个数据发送给处理机i, 则各处理机的计算不再需要数据交换。算法步骤如下:

算法 2　二维离散小波变换的并行算法

$j=0$;

第一步: 将数据$C_{j,m_1,m_2}(m_1=0,1,\cdots,N/2^j-1;\ m_2=0,1,\cdots,M/2^j-1)$ 按列块分配给P 台处理机;

第二步: 将处理机$i+1$ 中的前$L-1$ 个列的数据发送给处理机i;

第三步: 处理机i 负责C_{j+1,m_1,m_2} 和$D^1_{j+1,m_1,m_2},D^2_{j+1,m_1,m_2},D^3_{j+1,m_1,m_2}\left(m_2=i\frac{M}{P2^{j+1}},\cdots,(i+1)\frac{M}{P2^{j+1}}-1\right)$ 的计算;

第四步: $j=j+1$, 若$j+1<r$ 转到步 2。

注意, 每一步分解后数据C_{j+1,m_1,m_2} 和$D^1_{j+1,m_1,m_2},D^2_{j+1,m_1,m_2},D^3_{j+1,m_1,m_2}$ 已经是按块存储在P 台处理机上, 因此算法的第一步除了$j=0$ 时需要做外, 以后这一步根本就不需要做了(数据已经到位) 。因此, 按 Log$P$ 模型, 算法的总的通信时间为: $2(LN\max(o,g)+l)$, 远小于计算时间$O(NM)$。

对于一维离散小波或三维以上的离散小波, 可以类似地构造并行算法。同样, 也可以类似地构造逆离散小波的并行算法。

分别用 PVM 和 MPI (FORTRAN) 语言编制了一维和二维周期小波的并行计算程序。并在日立 SR2201 并行计算机和银河- 3 上进行了计算, 其中二维小波是采用一维张量积生成的。对二维情形我们考虑了通信和计算重叠的技巧, 既在数据进入通信网络传送的同时处理机就进行计算。表 2 为在 SR2201 上计算二维小波的计算时间和加速比 (基于 MPI 的程序), 其中时间 1 和加速比 1 表示考虑了通信和计算重叠时的时间和加速比, 而时间 2 和加速比 2 则表示未考虑通信和计算重叠时的时间和加速比。时间单位为秒。

由于 SR2201 的通信延迟相对于消息打包的时间而言比较小, 因此通信和计算重叠的技巧意义不大, 实际的计算时间基本相同。计算中出现超线性加速比, 主要原因是并行计算机高速缓存 (cache) 起了作用。

表2 二维小波的并行计算结果

Tab.2 Computational results for parallel z- D wavelets

数据大小	处理机台数	时间1	时间2	加速比1	加速比2
256×256	1	0.4585	0.4585	1.0000	1.0000
256×256	2	0.2321	0.2353	1.9754	1.9486
256×256	4	0.1187	0.1282	3.8627	3.5764
512×256	1	0.9091	0.9091	1.0000	1.0000
512×256	2	0.4590	0.4599	1.9806	1.9767
512×256	4	0.2326	0.2393	3.9084	3.7990
2048×256	1	6.6633	6.6633	1.0000	1
2048×256	2	3.0401	3.0316	2.1918	2.1979
2048×256	4	0.9121	0.9186	7.3054	7.2538

3 小波包变换的并行算法

小波包可以看成是小波的推广，近年来在数据压缩、图像处理等领域得到广泛应用，它比小波有更大的选择余地，从而可以得到更好的结果。小波包算法实际上是一大类算法的集合，在实际应用中必须根据需要选取合适的小波包算法。在数据压缩等领域，主要目的是尽可能多地去除数据中的冗余，因而选择最优小波包基的一种标准为：分解后的数据的熵最小。

下面以二维情形为例考虑完全小波包分解的并行算法。

对大小为 $N \times M$ 的输入数据 x_{k_1,k_2}，将它分解为4个序列：

$$C_{m_1,m_2} = \sum_{k_1,k_2} h_{k_1-2m_1,k_2-2m_2} x_{k_1,k_2},\quad D^1_{m_1,m_2} = \sum_{k_1,k_2} g^1_{k_1-2m_1,k_2-2m_2} x_{k_1,k_2}$$

$$D^2_{m_1,m_2} = \sum_{k_1,k_2} g^2_{k_1-2m_1,k_2-2m_2} x_{k_1,k_2},\quad D^3_{m_1,m_2} = \sum_{k_1,k_2} g^3_{k_1-2m_1,k_2-2m_2} x_{k_1,k_2}$$

其中 h, g^i 分别表示尺度系数和小波系数。下一步是对这4个序列的每一个都进行这样的分解，得到16个序列；然后再对这16个序列的每一个进行分解，…。一共进行 J 步，$J \le \log N, \log M$。实际应用中，一般而言，输入数据的尺寸很大，而 h, g^i 的非零项很少（集中在下标为0的附近），因此，计算输出序列在点 (m_1, m_2) 处的值只需要输入序列在点 $(2m_1, 2m_2)$ 附近点的值，这说明计算基本上只依赖局部数据。若将数据适当地分成大小相等的若干块，让每台处理机负责其中一块的分解，则在处理块内部点的分解时需要处理机之间的数据交换。若分块时让数据在边界上有一定的重叠（根据h，g^i 的非0点的数目确定要重叠的数据的多少），则可以减少处理机之间需要交换的数据量。每进行一步分解后需要对处理机之间的数据进行适当的重新分配。这时也要进行数据交换。在分解过程进行了一定的步数以后，以下的分解是对多个不同的序列进行完全独立的分解。当序列的个数已经达到或超过处理机的台数时，只要让每台处理机负责其中一个或多个序列的分解，这时，处理机之间不再需要进行数据交换。下表给出在银河III计算机上并行算法的计算时间。

表3 二维小波包算法的并行计算结果

Tab.3 Computational results for parallel 2- D wavelet packets

数据大小	处理机台数	时间	加速比	效率
512×128	4	2.094956	3.239984	80.99960%
	16	0.992840	6.836574	42.72859%
256×256	1	6.988105	1.000000	100%
	4	3.504525	3.009219	75.23048%
	16	1.357171	7.770490	48.56556%
1024×128	1	22.082481	1.000000	100%
	4	6.062259	3.642616	91.06540%
	16	2.243121	9.844534	61.52834%
512×256	1	13.667693	1.000000	100%
	4	6.001922	3.662109	91.55272
	16	2.234734	9.835311	61.47070%

致谢：本文得到国防科技大学数学技术实验室和并行与分布式计算国家重点实验室以及中科院并

行软件研究开发中心的支持。博士研究生吴建平参与了部分程序的调试。

参考文献:

[1] 曾泳泓, 成礼智, 周敏. 数字信号处理的并行算法 [M], 长沙: 国防科技大学出版社, 1999. 5.

[2] 蒋增荣, 曾泳泓, 余品能, 快速算法 [M] . 长沙: 国防科技大学出版社, 1994.

[3] 曾泳泓, 张小水, 二维离散 W 变换的快速算法及其应用 [J], 数值计算和计算机应用, 1997 年第一期.

[4] Zeng Y H, LI X M. Multi-dimensional PT algorithms for multi-dimensional DWT [A]. IEEE Trans. Signal Processing, 1999 (7) .

[5] 曾泳泓. 离散变换的快速算法和并行算法 [D], 长沙: 国防科技大学, 1997.

[6] C. Calvin. Implementation of parallel FFT algorithms on distributed memory machines with a minimum overhead of communication [T], Parallel Computing, 1996, 22: 1255～1279.

[7] 龙瑞麟. 高维小波分析 [M] . 北京: 世界图书出版公司, 1995.

[8] The special issue on wavelets. Proceedings of IEEE, 1996, 84 (4) .

[9] The special issue on parallel architecture for image processing [A]. Proceedings of IEEE, 1996, 84 (7) .