

基于 DHT 快速插值并行算模的研究

李中年, 张 朋, 谢杨华

(武汉理工大学 自动化学院, 湖北 武汉 430070)

摘 要:文中所研究的这种快速插值并行算模,是一种基于 DHT(Discrete Hartley Transform)的流水型模块式(即把若干个插值模块形序列全部变换为一条“流水线长龙状”序列)算模。这种算模的计算过程既不需要数据记录设施,亦不需要缓冲暂存环节,而且插值运算时间的复杂性同插值模块因子的复杂性独立无关。这种快速插值并行算模的每个计算周期持续时间相当于执行一个累积运算(加法运算和乘法运算)时间,运算简便迅速,因此对于实施高速计算应用非常有用。

关键词:快速插值;DHT;并行算模;“全流水”;模块

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2007)05-0242-03

Research on Fast Interpolation Parallel Algorithm Mode Based on DHT

LI Zhong-nian, ZHANG Peng, XIE Yang-hua

(Automation School, Wuhan University of Technology, Wuhan 430070, China)

Abstract: Introduces a new method of fast interpolation parallel algorithm mode, which is pipelining algorithm module based on DHT(Discrete Hartley Transform). In the mode, some interpolation module sequences are all converted into a “pipeline” sequence. The computational process need neither data record facilities, nor buffer scratch segment. And time complexity has independence on that of factor. In every computational cycle, the time duration is equivalent to cumulative computation(addition and multiplying) time. Besides these, it has advantage of convenience and rapidity, so it's valuable to real-time, high speed computation.

Key words: fast interpolation; DHT; parallel algorithm mode; pipeline; module

0 引 言

插值法应用日益广泛(诸如:数字电波形成;合成语言重构;多种波形编码;多媒体系统;数模转换等等),然而基于经典插值公式的方法,不仅计算量繁重,而且均方误差亦大,因此基于无关联正交变换的图表法和离散快速插值法应运而生,并且这些方法及其派生方法好似雨后春笋,层出不穷。由于在许多实际工程中,涉及的信号是实数值,所以 DHT(Discrete Hartley Transform)优于 DFT(Discrete Fourier Transform)^[1],并且工程实用中通常以 DHT 替换 DFT。尤其有趣的是:正向 DHT 等同于反向 DHT(即两者一样),因此仅用一种编码程序便可存储正向和反向的 DHT。

而文中研究的基于 DHT 快速插值并行算模算法比 DHT 一般插值并行算法更有效更优越,前者不但比

后者省用存储空间,并且前者比后者运行速率要高。文中研究的这种算模思想是:把要处理的若干个插值模块形序列全部变换成一条“流水线长龙状”序列,进行快速传输和运算处理(文中称之为:“全流水”处理),这样运作既不需要进行数据记录,也不需要进行数据暂存,从而省用存储空间和提高运行速率。

1 算 模

1.1 算模架构

用于 1-D 实数数据插值的基本架构主要包括 DHT 的计算阵列和中间矩阵计算模块。

采用一对 DHT 因子 $X(k)$ 和 $X(N-k)$,便可计算函数 $S(m, k)$ 和 $S(m, N-K)$,因此,对于“全流水”型计算,将 DHT 模块改进成能受理形如 $\{x(n), x(N-n)\}$ 的输入对和形如 $\{X(k), X(N-k)\}$ 的输出对,并且保持这种入/出模态,递推关系便可用于 DHT 模块中,推导如下:

序列 $\{x(n)\}$ 的 DHT 为:

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) [\cos(2\pi kn/N) +$$

收稿日期:2006-07-27

作者简介:李中年(1949-),男,湖北应城人,教授,研究方向为智能电气与软件工程。

$$\sin(2\pi kn/N)] \quad (1)$$

式中 $n = 0, 1, 2, \dots, N-1; k = 0, 1, 2, \dots, N-1$; 并且此式中 DHT 的 k 参量表达式可改写为:

$$X(k) = Y_1(k) + Y_2(k) \quad (2)$$

$$X(N-k) = Y_1(k) - Y_2(k) \quad (3)$$

式中 $k = 0, 1, 2, \dots, (N-1)/2$; 并且设 $X(N) = X(0)$; 还有:

$$Y_1(k) = [u_1(0)C_{0k} + u_1(1)C_{1k} + u_1(2)C_{2k} + \dots + u_1(P)C_{Pk}] \quad (4)$$

$$Y_2(k) = [u_2(0)S_{0k} + u_2(1)S_{1k} + u_2(2)S_{2k} + \dots + u_2(P)S_{Pk}] \quad (5)$$

而且式(5)中:

$$C_{nk} = \cos(2\pi kn/N) \quad (6)$$

$$S_{nk} = \sin(2\pi kn/N) \quad (7)$$

$$u_1(n) = [x(n) + x(N-n)] \quad (8)$$

$$u_2(n) = [x(n) - x(N-n)] \quad (9)$$

式(6)至式(9)中: $n = 0, 1, 2, \dots, P$; 并且设 $x(N) = 0$; 由正弦函数和余弦函数的性质可递推得到:

$$S_{(n+1)k} = S_{nk}C_k + C_{nk}S_k \quad (10)$$

$$C_{(n+1)k} = C_{nk}C_k - S_{nk}S_k \quad (11)$$

对于计算长度为 N 的 DHT, 由式(4), 式(5), 式(10), 式(11)便可形成包含 $(P+1)$ 个处理单元的基本架构, 其中每个 DHT 阵列包含两个输出加法器, 以分别计算所要求的输入(参见式(8)和式(9))以及 DHT 输出(参见式(2)和(3))。由式(4), 式(5), 式(10), 式(11)可见, 在每个计算周期, 要求 DHT 模块的每个处理单元进行 6 次乘法和 4 次加法。所有乘法和加法能同时进行, 以便高速运行。因此, 每个处理单元应当包含 6 个乘法器和 4 个加法器, 以便同时计算, 所以, 计算周期持续时间 T 可缩减为: $T = T_m + T_a$ (T_m 和 T_a 分别表示乘法时间与加法时间)。

1.2 算模算法

在第一个输入到达后, 与之相应的一个 DHT 模块的第一对输出获取 $(P+3)$ 个计算周期, DHT 模块后续各对输出将依次获取 P 个计算周期, 从而使 DHT 序列在每个 $(P+1)$ 周期得到“全流水”处理。DHT 模块处理单元的计算功能如下:

1NO: $CT \leftarrow 0$

$S \leftarrow 0$

$C \leftarrow 1$

2NO: $Y_1 \leftarrow Y_1 + u_{1in}C$

$Y_2 \leftarrow Y_2 + u_{2in}S$

$S \leftarrow SC_k + CS_k$

$C \leftarrow CC_k - SS_k$

$u_{1out} \leftarrow u_{1in}$

$u_{2out} \leftarrow u_{2in}$

$CT = CT + 1$

if($CT = P + 1$)

$Y_{1out} \leftarrow Y_1$

$Y_{2out} \leftarrow Y_2$

go to 1NO

其中 $C_k = \cos(2\pi k/N)$ 和 $S_k = \sin(2\pi k/N)$ 存储在 DHT 模块的第 $(k+1)$ 个处理单元; u_{1in} 和 u_{2in} 以及 u_{1out} 和 u_{2out} 分别为 DHT 模块处理单元的两输入和两输出; 而 Y_{1out} 和 Y_{2out} 亦为 DHT 模块处理单元的两输出; CT 为处理单元的计数; 其它参量含义如前所述。

当中间矩阵计算模块接收到输入侧 DHT 模块的输出时, 便依据下式进行计算:

$$S(m, k) = X(k)\cos(2\pi mk/K) - X(N-k)\sin(2\pi mk/K) \quad (12)$$

$$S(m, N-k) = X(N-k)\cos(2\pi mk/K) + X(k)\sin(2\pi mk/K) \quad (13)$$

对应于中间矩阵 $(L-1)$ 行的中间矩阵模块含有 $(L-1)$ 个单元。这种中间矩阵模块单元的计算功能如下:

3NO: if ($CT = 0$) then

$S \leftarrow 0$

$C \leftarrow 1$

$X_{1out} \leftarrow X_{1in}$

$X_{2out} \leftarrow 0$

else

4NO: $X_{1out} \leftarrow X_{1in}C - X_{1in}S$

$X_{2out} \leftarrow X_{2in} + X_{2in}S$

$C \leftarrow CC_{in} - SS_{in}$

$S \leftarrow SC_{in} + CS_{in}$

$CT \leftarrow CT + 1$

endif

if($CT = P + 1$)then

$CT \leftarrow 0$

go to 3NO

else

go to 4NO

endif

其中 $C_k = \cos(2\pi k/N)$ 和 $S_k = \sin(2\pi k/N)$ 存储在中间矩阵模块单元; 而 X_{1in} 和 X_{2in} 以及 X_{1out} 和 X_{2out} 分别为中间矩阵模块单元的两输入和两输出; CT 为中间矩阵模块单元的计数; 其它参量含义如前

所述。每个中间矩阵模块单元将其输出送给一个输出 DHT 模块,并且轮流输出所要求的插值信号。在一个计算周期里,每个中间矩阵模块单元依据式(12)和式(13)计算 $S(m, k)$ 和 $S(m, N - k)$, 并且依据式(10)和式(11)计算正弦值与余弦值。因此,在每个循环周期,需要进行 8 次乘法和 4 次加法运算,并且这两种运算可以同时进行。为了使中间矩阵模块单元的计算周期与 DHT 模块处理单元周期相匹配,每个中间矩阵模块单元应当含有 8 个乘法器和 4 个加法器,以便进行“全流水”处理。

1.3 算模流量

对 N 点序列到 NL 点序列 $1 - D$ 插值,文中提出的架构需要 $(L - 1)$ 个中间矩阵模块单元和 L 个线性阵列,以进行 DHT 计算,其中每个阵列含有 $(N + 1)/2$ 个处理单元。此架构总共需要 $L(N + 1)/2$ 个处理单元和 $(L - 1)$ 个中间矩阵模块单元,其中每个处理单元含有 6 个乘法器和 4 个加法器,而每个中间矩阵单元含有 8 个乘法器和 4 个加法器。在第一对输入激励架构生效后,插值输出的第一个“ L ”便从架构 $(N + 6)$ 个计算周期获得。在下一个 $(N - 1)/2$ 周期,架构将产生其余输出。 NL 点插值序列的第一个样品在 $(3N + 1)/2$ 计算周期里计算,在每 $(N + 1)/2$ 个计算周期 T ,计算 $N(L - 1)$ 个插值样品。因此,架构的流量(吞吐量)为 $(2NL)/(N + 1)T$ 。

2 范 例

任何 $2 - D$ 有限连续信号 $x_a(t_1, t_2)$,虽然可从其样本 $x(n_1, n_2)$ 用直接插值公式(3)进行插值处理,但功效较低^[2],而按文中研究的方法进行同样处理,则快捷得多,其处理过程分为两步进行。

第一步:用 $1 - D$ 插值算法“插入” $[x(n_1, n_2)]$ 的第一行,即 $\{x(0, n_2)\}$ 这一行,其中 $n_2 = 0, 1, \dots, N_2 - 1$;从而得到序列 $\{z(0, m_2)\}$,其中 $m_2 = 0, 1, \dots, N_2L - 1$;同理,“插值”于 $[x(n_1, n_2)]$ 的 $(N_1 - 1)$ 行,可得 $(N_1 - 1)$ 序列 $\{Z(i, m_2)\}$,其中 $m_2 = 0, 1, \dots, N_2L - 1$; $i = 1, 2, \dots, N_1 - 1$,于是得到大小为 $N_1 * N_2L$ 的中间矩阵 $[Z(n_1, m_2)]$ 。

第二步:对 $[Z(n_1, m_2)]$ 的第 1 列 $\{z(n_1, 0)\}$ 插值,其中 $n_1 = 0, 1, \dots, N_1 - 1$;可用 $1 - D$ 插值算法实

施,得到序列 $\{y(m_1, 0)\}$,其中 $m_1 = 0, 1, \dots, N_1L - 1$;同理,对 $[Z(n_1, m_2)]$ 的 $(N_2L - 1)$ 列重复进行“插值”,从而可得 $(N_2L - 1)$ 列其余各个序列 $\{y(m_1, m_2)\}$,其中 $m_1 = 0, 1, \dots, N_1L - 1$; $m_2 = 1, 2, \dots, N_2L - 1$;于是得到所求插值矩阵 $[y(n_1, n_2)]$ 。对此 $2 - D$ 插值^[3]的第一步中需要 N 个 $1 - D$ 插值,第二步中需要 NL 个 $1 - D$ 插值,并且需要进行下列“求实运算”(R0)^[4],即

$$R0 = LN_1[DHT(N_2)] + L^2N_2[DHT(N_1)] + [2(L^2 - 1)N_1N_2 - 3(L - 1)(N_1 + N_2L)]RM + [(L^2 - 1)N_1N_2 - 2(L - 1)(N_1 + N_2L)]RA \quad (14)$$

当 $DHT(N_i)$ 是计算 N_i 点 DHT 的架构复杂性时,式中 RM 和 RA 分别表征实数乘法和实数加法。

3 结 论

研发了一种基于插值算法的实数数据并行 DHT 算模,此算模基本计算处理思想似同“流水型模块式”运作。此算模将长度为 N 的序列“插值”成长度为 NL 的序列,只需运行 $(N + 1)/2$ 个计算周期^[5];并在每个计算周期(即相应的乘法与加法时间)产生 $2L$ 个输出^[6];而且插值处理时间复杂性与插值模块因子复杂性互不相关;计算处理过程中既不需要数据记录,也不需要数据暂存,这为设计实现高速计算、存储共享的并行处理机硬件芯片提供了理论参考依据。

参考文献:

- [1] Maharana G S, Meher P K. Algorithm for Interpolation of Real - Valued Signal Using Discrete Hartley Transform[J]. Computers and Elect Eng, 1997, 23(3): 129 - 134.
- [2] Chan S C, Ho K L, Kok C W. Interpolation of 2 - D Signal by Subsequence FFT[J]. IEEE Trans CAS - II, Analog and Digital Proc, 1993, 40(2): 115 - 118.
- [3] Satyanarayana P, Reddy P S, Swamy M N S. Interpolation of 2 - D Signals[J]. IEEE Trans CAS, 1990, 37(5): 623 - 625.
- [4] 李中年. M^2E^2 算模的研究[J]. 计算技术与自动化, 2005, 24(2): 33 - 36.
- [5] LI Zhongnian. Study on the N^2C^2 Problem[C] // Proc. of IEEE ICIPS'97. Beijing: [s. n.], 1997: 159 - 162.
- [6] LI Zhongnian. A New Method for the Model Processing of the ULS[C] // Proc. of ICSP'96 3RD. Beijing: [s. n.], 1996: 58 - 61.

《计算机技术与发展》欢迎投稿, 欢迎订阅