一种改进的基于 FFT Pruning算法的快速实现方法

李忠慧, 曾毓敏, 尹晓琦, 吴婷婷

(南京师范大学 物理科学与技术学院, 江苏 南京 210097)

[摘要] 针对文献[5]提出的 FFT Pruning算法作了一些改进, 得到了只计算 FFT频谱中部分频点谱值的改进的快速实现方法. 根据输入输出数据的结构特点, 利用辅助矩阵和数据复制等手段, 降低了 FFT Pruning算法实现的复杂度, 提高了 FFT Pruning算法实现的灵活性. 将改进后的 FFT Pruning算法用 C语言实现并在 DSP集成开发环境 CCS下的 C5402 Device Simulator上运行. 在相同条件下, 再运行一般意义上的 FFT算法和文献[5]中算法所对应的 C程序, 统计 3种方法的运行时间并比较他们的效率. 仿真结果表明: 在相同的条件下, 改进后的算法在快速准确地得到相关频谱值的同时, 运算时间明显少于另外两种方法. 同时, 对输入输出端所取数据的长度也没有任何限制.

[关键词] FFT Pruning算法, 算法实现, 数字信号处理, 频谱分辨率

[中图分类号] TN912.3 [文献标识码] B [文章编号] 1672-1292(2005)04-0042-04

An Improved Fast Implementation Method for FFT Pruning Algorithm

LI Zhonghui, ZENG Yumin, YIN Xiaoqi, WU Tingting

(School of Physical Science and Technology, Nanjing Normal University, Jiangsu Nanjing 210097, China)

Abstract Considering the FFT Pruning algorithm proposed in literature[5], this paper improves it and obtains an improved fast implementation method for FFT pruning algorithm in which only part spectral of the whole FFT's spectral need to be calculated. According to structural characteristics of the input and output data, the improved method utilizes the data replication and assistant matrix to decrease the computational complexity and enhance the implementary flexibility of the FFT Pruning algorithm. The improved method is realized in C and run in C5402 Device Simulator of CCS which is the integrated exploitive circumstance for DSP. The C programs corresponding to the algorithm in reference[5] and basic FFT algorithm are run respectively in the same circumstance too. After the three C program files are executed, the executive time of each algorithm are recorded to compare the efficiency of them. The emulational experiments show that in the same condition, contrasted with other methods, the improved method consumes less time obviously while the corresponding spectra can be obtained quickly and correctly, and that it doesn't limit the length of the input and output data too.

Key words FFT Pruning algorithm, implementation of algorithm, digital signal processing, spectrum resolution

0 引言

离散 Fourier变换(DFT)是信号处理及线性系统分析等领域最常用、最有效的数学工具之一. 而快速 Fourier变换(FFT)的提出使 DFT理论得以推广并被广泛应用, 并对频谱分析、卷积与相关、数字滤波器设计、功率谱分析、传递函数建模等的快速计算起到了关键性作用. 但 FFT固有的频率分辨率与计算量之间的矛盾从某种程度上也限制了它的应用, 因为要提高 FFT的频率分辨率, 就要增加采样点数及计算量. DFT在时域和频域都是 N 点的周期序列, 为了提高频谱的分辨率, 常常在输入数据 $x(n)$ 后面补零, 当然这要以增加计算量为代价. 而当数据足够长时, 有时又不需要频域的全部频点, 而只要得到频谱中某一频带或一部分频点的值即可, 那么关于无关频点的计算就是多余的, 找到能够有效去除无关频点的蝶形计算以提高运算效率的 FFT Pruning算法一直是学者们所关注的. 自

收稿日期: 2005-09-28

作者简介: 李忠慧(1978-), 女, 硕士研究生, 主要从事物理电子学与语音信号处理等方面的学习与研究. E-mail: lililzh@163.com

通讯联系人: 曾毓敏(1962-), 教授, 博士, 主要从事语音信号处理方面的教学与研究. E-mail: zengyumin@njnu.edu.cn

从 Markel J D首次于 1971年提出了在输入端仅有少数非零点的简化快速 Fourier 算法 (FFT Pruning) 后[1], 不断有人提出相关的基于 Cooley-Tukey算法的 FFT Pruning算法[2~4], 但这类算法均要求输入、输出数据长度是某一整数 k 的 2^k, 且所得是频谱中某一个频带内的连续频点的谱值, 无法在简化运算量的同时得到相应频带内频点位置有间隔的谱值. 其后 Rogerio G Alves等人于 2000年在文献[5]中提出了一种新的 FFT Pruning 的实现方法, 它与一般的 FFT Pruning实现方法有很大的不同, 运算过程中它需要生成 $N \times \log_2(N)$ 大小的辅助矩阵, 然后根据辅助矩阵中的各个元素值决定对应的蝶型单元是否计算. 它的优点在于不限定输入、输出数据的长度, 并可得到频谱中某一频带内任意频点的谱值. 但是它也存在某些不足之处: 其一, 当 FFT计算长度为 N, 输入数据中仅有少数数据为非零值时, 要在输出端得到部分频点的输出谱值时, 需产生 3个 $N \times \log_2(N)$ 的辅助矩阵, 当 N 较大时, 矩阵的产生和相关运算所耗费的计算量是非常可观的; 另外, 这一方法要根据输入数据中的非零值分布情况生成一个 $N \times \log_2(N)$ 矩阵, 但实际应用中, 非零值在 N 个输入数据之中任意分布的情况并不多见, 而绝大多数的应用场合都是在少数非零数据之后添加大量的“0”值使输入数据总长度达到 N, 以提高输出端的频谱分辨率, 这种情况下的输入数据结构具有很强的规律性, 完全可以不用生成对应于输入端的矩阵. 本文根据实际常用的应用场合 (如频谱分析、卷积与相关、功率谱分析等), 结合文献[5]的实现方法, 提出了一种改进的实现方法. 该方法充分利用输入数据结构的规律性, 在 FFT运算流程的前端仅需执行简单的前后级相关数据的复制, 然后根据输出数据的需要, 仅需生成一个输出端辅助矩阵, 因而大大降低了算法实现的计算量及内存的使用.

1 改进的 FFT Pruning算法的快速实现原理

设时域信号波形的采样离散值为 $x(n)$, 假设 $x(n)$ 的长度为 NI, FFT的长度为 $N = 2^M$, NI远小于 N, 显然这是一个输入端仅包括少数非零点的 FFT 变换. 针对这一特点, 采用按时间抽选 (DIT) FFT Pruning算法, 在 $x(n)$ 之后补零, 直至 $x(n)$ 的长度为 N. 若 NI不是 2的整次幂, 由于 DIT FFT Pruning算法要求输入数据长度为 2的整次幂, 则取比 NI大的 2的整次幂数中最小的数 $NL = 2^L$, 那么此时就认为输入 $x(n)$ 中有 $NL = 2^L$ 个非零值点, 即认为 $x(n)$ 的非零值序号是从 0至 $2^L - 1$, 而不再是 0至 $NI - 1$, 零值点的个数等于 $2^M - 2^L = 2^L(2^{M-L} - 1)$, 此时, 共有前 $M - L$ 级可应用 DIT FFT Pruning简化运算. 同时, 在对补零后的 $x(n)$ 执行码位倒置时, 只需对 $NL = 2^L$ 个非零数据倒置即可, 而不需要对 N 个含有大量零值的输入数据执行码位倒置. 以下以 $N = 16$, $NL = 4$ 为例来展示 DIT FFT Pruning算法, 信号流图如图 1所示.

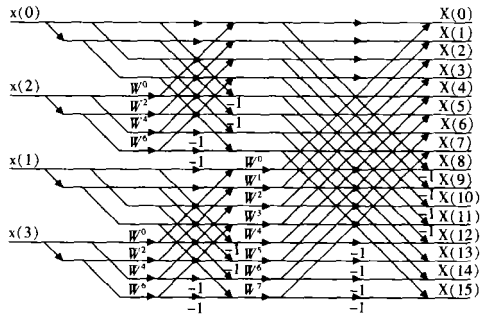

图 1 DIT FFT Pruning 信号流图

观察图 1中简化的半蝶形单元, 由于输入按时间抽取, W_N 因子乘在输入端的下部分, 且输入端的下部分等于 0, 所以前 $M - L$ 级的半蝶形运算单元既不需要乘法, 也不需要加法, 每一个非零点只需被复制到下一级的相应位置即可, 而不需要再执行 $(M - L) \times 2^{M-1}$ 个蝶形运算, 而计算一个完整的蝶形单元需要 4次实乘、6次实加, 这样, 由于输入端采用了简化算法, 计算量大大减少, 仅应用输入端简化算法就可节约 $\frac{M - L}{M} \times 100\%$ 的时间.

最后 L 级的 FFT运算, 需要生成一个 N 行、L 列的矩阵, 矩阵中的每一个元素与最后 L 级 FFT算法流图中的相应的节点一一对应, 矩阵中元素的值只取“0”或“1”. 在算每一个节点值之前先检验一下矩阵中对应的元素值, 以确定该节点是否需要计算, 只有当矩阵对应的元素值为“1”时, 才对相应的节点执行蝶形运算, 本文中用符号 $\boldsymbol{Mo}$ 表示该矩阵. 为了得到矩阵 $\boldsymbol{Mo}(N \times L)$, 先考虑一个有 N 个元素的列向量, 暂时用符合 ***outn*** 来表示该列向量, 向量中的每一个值与相应的输出信号对应, 当需要该频点谱值输出时, 将列向量对应位置的元素置“1”, 否则置“0”. 用该列向量 ***outn*** 的值作为矩阵 $\boldsymbol{Mo}$ 的最后一列, 再用最后一列列向量推算出倒数第二列, 依此类推, 直至第 $M - L + 1$ 列. 图 2将以具体的图例演示如何得到矩阵 $\boldsymbol{Mo}$, 要求 FFT的长度 $N = 16$, 输入端非零数据长度 $NL = 4$, 输出端要求得到 $X(2K)$, $K = 0, 1, 2, 3\cdots$ 的频谱值, 即只需输

出 $X(0)$、$X(2)$、$X(4)$、$X(6)$、$X(8)$ …,

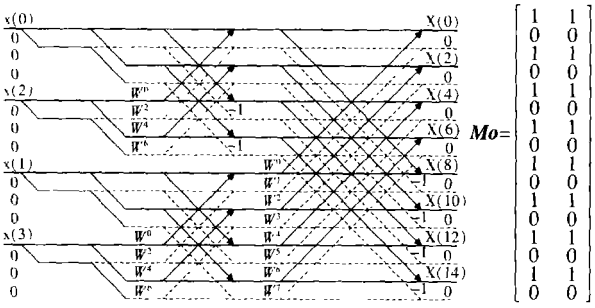

图 2　改进的 FFT Pruning 信号流图及对应的矩阵 *Mo*

如图 2所示, 图中实线表示实际需要运算的蝶形运算, 虚线表示略去的蝶形运算. 如前所述, $outn = [1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]'$, 根据此向量依此类推出矩阵 $\boldsymbol{Mo}$ 中的每一个值, 对应于改进后的 FFT Pruning信号流图, 可以看出矩阵 $\boldsymbol{Mo}$ 为一个 16行、2列的矩阵. 在 4级蝶型运算中, 前 2级运算直接复制非零点即可, 后 2级在执行每一个蝶形运算之前, 先判断矩阵 $\boldsymbol{Mo}$ 中与相应的蝶形节点对应的 2个元素是否为“1”, 2个元素中有一个为“1”就执行相关蝶形运算, 否则不予执行.

文献[5]算法不限制输入、输出数据的长度一定是 2的整次幂, 输出频点的位置可以是不连续的甚至是任意的, 应用起来很灵活. 但是它并不是在非零数据之后补零至序号 $n = N - 1$, 而是相当于按照一定的规律或是任意地在每两个非零的 $x(n)$ 之间添加 0以使数据长度最终为 N[5]. 显然, 按这种方法得到的频谱不等于在原非零输入数据之后补零再经过 FFT变换后的频谱. 如果输入、输出端都做简化运算, 总共需要用到 3个 $N \times M$ 的矩阵 $\boldsymbol{Mi}$、$\boldsymbol{Mo}$ 和 $\boldsymbol{Mio}$, 有关这 3个矩阵的运算量是不可忽略的, 这对于提高运算速度是不利的. 而本文提出的改进的 FFT Pruning实现方法则克服了以上的缺陷, 在整个运算过程中, 前 M–L 级应用了 FFT DIT Pruning算法, 无需任何的蝶形计算以及对矩阵元素的判别, 只需执行复制操作即可, 后 L 级只需要生成一个大小为 $N \times L$ 的矩阵 $\boldsymbol{Mo}$, 显然, 非零数据越少, 矩阵 $\boldsymbol{Mo}$ 就越小, 所做的蝶型运算就越少, 节约的时间也就越多.

2　实验结果及分析

对非零数据长度 $NL = 64$ 的时域信号 $x(n)$ 分别做长度 $N = 128$、256、512点的 FFT, 要求得到 $X(4K)$, $K = 0, 1, 2, 3\cdots$ 的频谱值. 在相同的条件下, 分别使用本文快速实现方法、文献[5]所述方法及一般意义的 FFT这 3种方法求频谱, 对应的 C程序运行环境为 DSP集成开发环境 CCS下的软件模拟器 C5402 Device Simulator, 并采用软件模拟器提供的剖析函数的功能对 3种方法的 C源程序进行剖析, 统计各个程序的运行时钟周期总数以比较各实现方法的优劣, 实验结果见表 1. 再分别对非零数据长度 $NL = 16$、32的 $x(n)$ 做长度 $N = 512$ 点的 FFT, 也要求得到 $X(4K)$ 的频谱值, 其他条件不变, 运用软件模拟器剖析后的实验结果如表 2所示. 其中, pfft对应于本文改进的 FFT Pruning快速实现方法的 C程序; gfft对应于文献[5]中实现方法的 C程序; ff对应于一般意义的 FFT算法的 C程序, 本文直接采用文献[6]中所附的程序. 运行时间比是指在相同条件下, 分别用 pfft、gfft对应的运行时钟周期总数除以 fft对应的时钟周期总数得到的百分比.

表 1　3种 FFT实现方法的比较 1(NL 固定, N 变化)

FFT 长度	$NL = 64, N = 128$			$NL = 64, N = 256$			$NL = 64, N = 512$		
实现方法	pfft	gfft	fft	pfft	gfft	fft	pfft	gfft	fft
时钟周期数	345660	514562	3309023	606017	1033797	7512745	1150507	2172838	18218531
运行时间比	10.45%	15.55%	—	8.07%	13.76%	—	6.32%	11.93%	—

表 2　3种 FFT实现方法的比较 2(NL 变化, N 固定)

FFT 长度	$NL = 64, N = 512$			$NL = 32, N = 512$			$NL = 16, N = 512$		
实现方法	pfft	gfft	fft	pfft	gfft	fft	pfft	gfft	fft
时钟周期数	1150507	2172838	18218531	1024684	2126486	18026605	887637	2058498	17821323
运行时间比	6.32%	11.93%	—	5.68%	11.8%	—	4.98%	11.55%	—

从表 1、表 2可以看出, 本文所改进的 Pruning FFT快速实现方法明显优于文献[5]中的方法, 可以节约的运算时间是显而易见的. 表 1中, 当非零数据长度 NL 固定不变时, 随着 N 的增大, pfft的运行时间百分比迅速从 10.45% 下降到 6.32%, 而当 $N = 512$ 时, 程序 pfft求取频谱的时间几乎只有 gfft的一半. 相似地, 在表 2中, FFT运算长度 N 固定为 512时, 随着 NL 的减少, 程序 pfft的运行时间也逐渐减少, 而 gfft的几乎不变, 当 $NL = 16$ 时, 程序 pfft的运行时间甚至少于程序 gfft的一半. 究其原因在

于,算法的实现过程中涉及产生矩阵以及与矩阵大小相关的蝶型单元的运算,N 越大,相应的矩阵越大,需要运算的蝶型单元也就越多. 改进的 FFT Pruning实现方法只需产生一个$N \times \log_2(NL)$的矩阵,而后者需要产生 3个$N \times \log_2(N)$矩阵,当N较大时,前者的矩阵比后者小得多,整个程序花费的运行时间必然远少于后者.

3 结论

本文在文献[5]提出方法的基础上,结合文献[2]提出的输入端简化算法,对其中的实现方法作了改进,得到一种改进的基于 Pruning FFT算法的快速实现方法. 当输入端仅有少数非零数据时,无需限定输入、输出数据的长度,通过去除无关频点的运算,能够快速地得到频点位置任意的输出频谱. 由于改进后的实现方法只针对某些我们感兴趣的频点做计算,略去无关频点的计算, 大大降低了运算量,提高了使用的高效性和灵活性. 同时本文用 C语言实现了文中提出的快速实现方法,并在 DSP C5402 Device Simulator软件模拟器上运行,统计其运行时钟周期,并与在相同条件下其他的 FFT Pruning实现方法的运行时钟周期相比较,实验结果证明,本文提出的快速实现方法的计算量明显少于其他方法,有较强的实用性和参考价值.

[参考文献]

[1] Markel J D. FFT pruning[J]. IEEE Trans on Audio Electroacoust, 1971, 19(4): 305-311.

[2] Skinner D P. Pruning the decimation in time FFT algorithm[J]. IEEE Trans on ASSP, 1976, 24(3): 193-194.

[3] Sreenivas T V, Rao P V S. FFT algorithm for both input and output pruning[J]. IEEE Trans on ASSP, 1979, 27(3): 291-292.

[4] Nagai K. Pruning the decimation-in-time FFT algorithm with frequency shift[J]. IEEE Trans on ASSP, 1986, 34(4): 1008-1010.

[5] Alves R G, Osorio P L, Swamy M N S. General FFT pruning[C]// Proceedings of the 43rd IEEE Midwest Symposium. Michigan: Lansing, 2000: 1192-1195.

[6] 胡广书. 数字信号处理——理论、算法与实现[M]. 2版. 北京:清华大学出版社, 2003: 187-192.

[责任编辑: 刘 健]

(上接第 10页)

[参考文献]

[1] 欧洲风能协会. 风力 12: 关于 2020年风电达到世界电力总量 12%的蓝图[M]. 绿色中国,译. 北京: 中国环境科学出版社, 2004: 20-21.

[2] Tomas Petru. Modeling of wind turbine for power system studies[D]. Sweden: Chalmers University of Technology, 2003.

[3] 张新房,徐大平,吕跃刚,等. 风力发电技术的发展和若干问题[J]. 现代电力, 2003, 20(5): 29-34.

[4] Lars Gertmar. Power electronics and wind power[D]. Sweden: Lund University, 2003.

[5] Vestas V47-660 kW brochure[EB/OL]. [2000-12-06]. http://www.vestas.com/UK/news/press/20001found 20001206 UK.html.

[6] 伍小杰, 柴建云, 王祥. 变速恒频双馈风力发电系统交流励磁综述[J]. 电力系统自动化, 2004, 28(23): 92-96.

[7] Wikie J, Leithead W E, Anderson C. Modeling of wind turbines by simple models[J]. Wind Engineering, 1990, 14(4): 247-273.

[8] Frede Blaabjerg, Zhe Chen, Soeer Baekhoej Kjaer. Power electronics as efficient interface in dispersed power generation systems[J]. IEEE Trans on Power Electr, 2004, 19(5): 1184-1194.

[9] 张桂斌, 徐政. 直流输电技术的新发展[J]. 中国电力, 2000, 33(3): 32-35.

[10] Rodríguez J M, Fernández J L, Beato D, *et al*. Incidence on power system dynamics of high penetration of fixed speed and doubly fed wind energy systems: study of spanish case[J]. IEEE Trans on Power System, 2002, 17(4): 1089-1095.

[11] 马洪飞, 徐殿国, 苗立杰. 几种变速恒频风力发电系统控制方案的对比分析[J]. 电工技术杂志, 2000(10): 1-4.

[责任编辑: 严海琳]