(18) 96-107

实序列斜圆卷积的实值变换计算法*

A REAL-VALUED TRANSFORM APPROACH FOR SKEW-CIRCULAR CONVOLUTION

TP301

周六丁
Zhou Liuding
(重庆大学电子信息工程学院)

摘 要 实序列斜圆卷积是二维卷积多项式变换计算法中的核心计算。本文利用实值变换的快速性及斜圆卷积的特殊性，导出一种计算 $N(N=2^M)$ 点实序列斜圆卷积的新算法。它完成该计算仅需 $N\cdot(\log_2 N+1)$ 次实乘、$3N\cdot(\log_2 N-\frac{1}{3})$ 次实加，这分别仅约为 FFT 计算法所需的 1/4、1/2。如将它与多项式变换法结合计算 $N\times N(N=2^M)$ 二维实圆卷积，则仅需 $N^2\cdot\log_2 N$ 次实乘、$4N^2\cdot\log_2 N$ 次实加，这分别仅约为 FFT 计算法所需的 1/8、1/3。

关键词 算法设计；算法分析；卷积；变换 斜圆

中国图书资料分类法分类号 TP301

ABSTRACT A new algorithm for skew-circular convolution of real sequences is proposed in this paper, which employs the feature of fast computation on discrete W transform and the speciality of the skew-circular convolution. Its real multiplication and addition requirements are $N\cdot(\log_2 N+1)$ and $3N(\log_2 N-\frac{1}{3})$ respectively in evaluating skew-circular convolution of $N(N=2^M)$ real-valued data.

KEY WORDS algorithm design; algorithm analysis; convolution; transform

0 引 言

设 $x_n(n=0,1,\cdots,N-1)$，$h_m(m=0,1,\cdots,N-1)$ 是两个实数序列，其斜圆卷积是指：

$$y_l=\sum_{n=0}^{l}x_n\cdot h_{l-n}-\sum_{n=l+1}^{N-1}x_n\cdot h_{N+l-n}(l=0,1,\cdots,N-1) \tag{1}$$

若令 $X(z)$、$H(z)$、$Y(z)$ 分别为由序列 $x_n(n=0,1,\cdots,N-1)$、$h_m(m=0,1,\cdots,N-1)$、$y_l(l=0,1,\cdots,N-1)$ 为系数构成的多项式，即：

$$X(z)=\sum_{n=0}^{N-1}x_n\cdot z^n;H(z)=\sum_{m=0}^{N-1}h_m\cdot Z^m;Y(z)=\sum_{l=0}^{N-1}y_l z^l$$

则由多项式乘法及 $Z^N\equiv -1\bmod(Z^N+1)$ 可得：

* 收文日期 1991-05-07
由国家基金项目和重庆市科委项目资助

$$Y(z) = X(z) \cdot H(z) \bmod (Z^N + 1) \tag{2}$$

容易验证式(1)与式(2)的计算等价。这两种基本运算在信号处理、图象处理中常用到[1]。特别是在二维实圆卷积的多项式变换计算法中[1,2]，二维圆卷积映射为多组如式(2)的多项式积，因此式(1)的计算速度直接支配二维实圆卷积的速度。

常规实序列斜圆卷积的计算法过程如下[2]：

(Ⅰ)由输入序列 $x_n(n = 0,1,\cdots,N-1)$，$h_m(n = 0,1,\cdots,N-1)$ 构成序列：

$$x_n' = x_n \cdot W_{2N}^n; h_m' = h_m \cdot W_{2N}^m; (m,n = 0,1,\cdots,N-1; W_{2N} = e^{-j\pi/N})$$

(Ⅱ)计算 $\{x_n'\}$，$\{h_m\}$ 序列的圆卷积，即：

$$y_l' = \sum_{n=0}^{N-1} x_n' \cdot h_{(l-n)_N}'(l = 0,1,\cdots,N-1; (\,)_N \text{取最小非负剩余})$$

(Ⅲ)由 $y_l'(l = 0,1,\cdots,N-1)$ 序列计算斜圆卷积输出：

$$y_l = y_l' \cdot W_{2N}^l(l = 0,1,\cdots,N-1; W_{2N} = e^{-j\pi/N})$$

其中第(Ⅱ)步利用 DFT 的圆卷积特性(借助 FFT)来计算[3]，即 $\{y_l'\} = \mathrm{IDFT}\{\mathrm{DFT}\{x_n'\} \cdot \mathrm{DFT}\{h_m'\}\}$。

在应用中 $\{h_m\}$ 预先确定，则 $\mathrm{DFT}\{h_m'\}$ 可预先计算好，因此完成第Ⅱ步仅需两次 DFT(正/逆各一次)的计算及点乘 $Y_k = X_k \cdot H_k = \mathrm{DFT}\{x_n'\} \cdot \mathrm{DFT}\{h_m'\}(k = 0,1,\cdots,N-1)$ 的计算。用常规 *FFT* 算法计算 DFT(假定序长 $N = 2^M$)，完成步Ⅱ共需 $2[(N/2) \cdot (\log_2 N - 3) + 2] + N$ 次复乘及 $2N\log_2 N$ 次复加。完成步Ⅰ及步Ⅲ仅需 $4N-4$ 次实乘和 $2N-2$ 次实加(因是特殊复乘)。通常一次复乘用 4 次实乘 2 次实加实现，一次复加用 2 次实加实现。因而用上述常规 FFT 计算法计算 $N(N = 2^M)$ 点实序列斜圆卷积共需的实数乘、加法次数分别为：

$$M_2^{\mathrm{FFT}}(N) = 4\{2[(N/2)(\log_2 N - 3) + 2] + N\} + 4N - 4 = 4N(\log_2 N - 1) + 12$$

$$A_2^{\mathrm{FFT}}(N) = 4N\log_2 N + 2\{2[(N/2)(\log_2 N - 3) + 2] + N\} + 2N - 2$$
$$= 6N(\log_2 N - 1/3) + 6$$

这种沿用数十年的计算法看来不可能改进(除用 Rader-Brenner FFT 计算步Ⅱ时略有改进外)，但一旦引入实域变换(离散 W 变换)将大幅度减少计算实序列斜圆卷积或模 $Z^N + 1$ 多项式积的计算量。本文首先建立 DFT 与 W 变换间关系，然后给出新算法的推导及证明，最后给出新算法的描述及其运算量的分析。分析结果表明新算法完成 $N(N = 2^M)$ 点实序列斜圆卷积仅需 $N(\log_2 N + 1)$ 次实乘、$3N(\log_2 N - 13)$ 次实加及 $2N$ 个实数单元。当 N 较大时，其实数乘、加次数分别约为常规 FFT 计算法所需的1/4、1/2。在此需要指出，如将该算法与多项式变换法配合使用来计算二维实圆卷积，其所需实数乘、加次数分别仅约为常规 FFT(行列法)计算法所需的1/8、1/3(见讨论)。且本文中的思想及证明都是新的。

1　DFT 与离散 W 变换间的关系

1.1　一维 DFT 的定义及 W 变换的定义与计算

定义1.1　设序列 $x_n(n = 0,1,\cdots,N-1)$，变换：

$$X_k = \sum_{n=0}^{N-1} x_n \cdot W_N^{kn}(k = 0,1,\cdots,N-1; W_N = e^{-j2\pi/N})$$

称为序列 $\{x_n\}$ 的 DFT，$\{X_k\}$ 序列的逆变换 IDFT 为：

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot W_N^{-kn} (n = 0,1,\cdots,N-1)$$

我国著名学者王中德提出的离散 W 变换是一种定义在实域上的变换[4,5]。他定义了 W^{I}、W^{II}、W^{III} 及 W^{IV} 四种类型的 W 变换，本文中用到 W^{I}、W^{II}、W^{III} 型变换，定义如下：

定义 1.2 给定一个实数序列 $x_n (n = 0,1,\cdots,N-1)$，变换：

$$X_k^{\mathrm{I}} = \sqrt{\frac{2}{N}} \cdot \sum_{n=0}^{N-1} x_n \cdot \sin\left(\frac{\pi}{4} + k \cdot n\frac{2\pi}{N}\right)$$

$$= \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x_n \cdot [\cos(2\pi k \cdot n/N) + \sin(2\pi k \cdot n/N)]$$

$$(k = 0,1,\cdots,N-1)$$

$$X_k^{\mathrm{II}} = \sqrt{\frac{2}{N}}\sum_{n=0}^{N-1} x_n \cdot \sin\left[\frac{\pi}{4} + k \cdot \left(n + \frac{1}{2}\right)\frac{2\pi}{N}\right]$$

$$= \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x_n \cdot \{\cos[2\pi k(n + 1/2)/N] + \sin[2\pi kn(n + \frac{1}{2})/N]\}$$

$$(k = 0,1,\cdots,N-1)$$

及：

$$X_k^{\mathrm{III}} = \sqrt{\frac{2}{N}}\sum_{n=0}^{N-1} x_n \cdot \sin\left[\frac{\pi}{4} + \left(k + \frac{1}{2}\right)n \cdot \frac{2\pi}{N}\right]$$

$$= \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x_n \cdot \{\cos[2\pi(k + \frac{1}{2}) \cdot n/N] + \sin[2\pi(k + \frac{1}{2}) \cdot n/N]\}$$

$$(k = 0,1,\cdots,N-1)$$

分别称为序列$\{x_n\}$的 WI 型变换、$W\,\mathrm{II}$ 型变换和 $W\,\mathrm{III}$ 型变换。现有多种算法(及程序)可快速计算上述变换，为讨论方便，以文献[5]中算法(及程序)为标准。

结论1.1 当给定实序列 $\{x_n\}$ 的长度 $N = 2^M$，用文献[5]中算法完成该序列的 W^{I} 型变换、W^{II} 型变换及 W^{III} 型变换所需实数乘、加次数分别为：

$$M_N^{W^{\mathrm{I}}}(N) = \frac{N}{2}(\log_2 N - 3) + 2; A_N^{W^{\mathrm{I}}}(N) = \frac{N}{2}(3\log_2 N - 5) + 6$$

$$M_N^{W^{\mathrm{II}}}(N) = M_N^{W^{\mathrm{III}}}(N) = \frac{N}{2}(\log_2 N - 1); A_N^{W^{\mathrm{II}}}(N) = A_N^{W^{\mathrm{III}}}(N) = \frac{3}{2}N(\log_2 N - 1)$$

1.2 用 W^{III} 型变换计算 DFT$\{x_n \cdot W_{2N}^n\}$

序列 $x_n' = x_n \cdot W_{2N}^{2n} (x_n \in R; n = 0,1,\cdots,N-1; W_{2N} = e^{-j2\pi/2N})$ 的 DFT 为：

$$X_k = \sum_{n=0}^{N-1}(x_n \cdot W_{2N}^n)W_N^{kn} = \sum_{n=0}^{N-1} x_n \cdot W_N^{(k+\frac{1}{2})n}$$

$$= \sum_{n=0}^{N-1} x_n \cdot \cos\left[2\pi\left(k + \frac{1}{2}\right)n/N\right] - j\sum_{n=0}^{N-1} x_n \cdot \sin\left[2\pi\left(k + \frac{1}{2}\right)n/N\right] \tag{3}$$

$$(k = 0,1,\cdots,N-1; W_N = e^{-j2\pi/N})$$

而序列 $x_n (n = 0,1,\cdots,N-1)$ 的 $W\,\mathrm{III}$ 型变换为：

$$X_k^{\mathrm{III}} = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x_n \cdot \{\cos[2\pi(k + \frac{1}{2})n/N] + \sin[2\pi(k + \frac{1}{2}) \cdot n/N]\} \tag{4}$$

$$(k = 0,1,\cdots,N-1)$$

在式(4)中以 $N-1-k$ 代 k 化简有：

$$X^{\mathrm{II}}_{N-1-k}=\frac{1}{\sqrt{N}}\sum_{n=0}^{N-1}x_n\cdot\{\cos[2\pi(k+\frac{1}{2})n/N]+\sin[2\pi(k+\frac{1}{2})\cdot n/N]\}\qquad(5)$$

$$(k=0,1,\cdots,N-1)$$

用式(5)分别加、减式(4)有：

$$X^{\mathrm{II}}_k+X^{\mathrm{II}}_{N-1-k}=\frac{2}{\sqrt{N}}\sum_{n=0}^{N-1}x_n\cdot\cos[2\pi(k+\frac{1}{2})n/N]$$

$$X^{\mathrm{II}}_{N-1-k}-X^{\mathrm{II}}_k=\frac{-2}{\sqrt{N}}\sum_{n=0}^{N-1}x_n\cdot\sin[2\pi(k+\frac{1}{2})n/N]$$

则：
$$X_k=\frac{\sqrt{N}}{2}(X^{\mathrm{II}}_k+X^{\mathrm{II}}_{N-1-k})+j\frac{\sqrt{N}}{2}(X^{\mathrm{II}}_{N-1-k}-X^{\mathrm{II}}_k)(k=0,1,\cdots,N-1)\qquad(6)$$

1.3 用 W^{I} 型变换计算 IDFT$\{T_k\}$（$\{T_k\}$ 为实序列）

给定一实序列 $T_k(k=0,1,\cdots,N-1)$，其 IDFT 为：

$$t_l=\frac{1}{N}\sum_{k=0}^{N-1}T_k\cdot W_N^{-kl}(l=0,1,\cdots,N-1;W_N=e^{-j2\pi/N})$$

$$=\frac{1}{N}\sum_{k=0}^{N-1}T_k\cdot\cos(2\pi l\cdot k/N)+j\frac{1}{N}\sum_{k=0}^{N-1}T_k\cdot\sin(2\pi l\cdot k/N)$$

而 $\{T_k\}$ 序列的 W^{I} 型变换为：

$$t'_l=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot[\cos(2\pi l\cdot k/N)+\sin(2\pi l\cdot k/N)]\qquad(l=0,1,\cdots,N-1)\qquad(7)$$

显然 $t_0=\frac{1}{\sqrt{N}}t'_0$，当 $l\neq 0$ 时在式(7)中以 $N-l$ 代 l 有：

$$t'_{N-l}=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot[\cos(2\pi l\cdot k/N)-\sin(2\pi l\cdot k/N)]\qquad(l=0,1,\cdots,N-1)\qquad(8)$$

用式(7)分别加、减式(8)有：

$$t'_l+t'_{N-l}=\frac{2}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot\cos(2\pi l\cdot k/N);t'_l-t'_{N-l}=\frac{2}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot\sin(2\pi l\cdot k/N);$$

从而有：
$$t_0=\frac{1}{\sqrt{N}}t'_0\qquad(9)$$

$$t_l=\frac{1}{2\sqrt{N}}(t'_l+t'_{N-l})+j\frac{1}{2\sqrt{N}}(t'_l-t'_{N-l})(l=1,2,\cdots,N-1)\qquad(10)$$

由于 $\{T_k\}$ 序列的 W^{I} 型变换为：

$$t'_l=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot\{\cos[2\pi l\cdot(k+\frac{1}{2})/N]+\sin[2\pi l\cdot(k+\frac{1}{2})/N]$$

$$=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}T_k\cdot\{\cos(2\pi l\cdot k/N)\cdot\cos(\pi l/N)-\sin(2\pi l\cdot k/N)\cdot\sin(\pi l/N)$$

$$+\sin(2\pi l\cdot k/N)\cdot\cos(\pi l/N)+\cos(2\pi l\cdot k/N)\cdot\sin(\pi l/N)\}$$

$$=\frac{1}{\sqrt{N}}\cos(\pi l/N)\cdot\sum_{k=0}^{N-1}T_k\cdot[\cos(2\pi l\cdot k/N)+\sin(2\pi l\cdot k/N)]$$

$$+\frac{1}{\sqrt{N}}\sin(\pi l/N)\cdot\sum_{k=0}^{N-1}T_k[\cos(2\pi l\cdot k/N)-\sin(2\pi l\cdot k/N)]$$

则：
$$t'_l=\cos(\pi l/N)\cdot t'_l+\sin(\pi l/N)\cdot t'_{N-l}\qquad(11)$$

在式(11)中以 $N-l$ 代 l 有：

$$t^{\mathrm{I}}_{N-l} = -\cos(\pi l/N)\cdot t^{\mathrm{I}}_{N-l} + \sin(\pi l/N)\cdot t^{\mathrm{I}}_{l} \tag{12}$$

当 $l=0$ 时显然有 $t_0 = t^{\mathrm{I}}_0 = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} T_k$，$l \neq 0$ 时由式(11)、式(12)可解得：

$$t^{\mathrm{I}}_{l} = t^{\mathrm{I}}_{l}\cos(\pi l/N) + t^{\mathrm{I}}_{N-l}\cdot\sin(\pi l/N) \tag{13}$$

在式(13)中以 $N-l$ 代 l 有：

$$t^{\mathrm{I}}_{N-l} = -t^{\mathrm{I}}_{N-l}\cdot\cos(\pi l/N) + t^{\mathrm{I}}_{l}\cdot\sin(\pi l/N) \tag{14}$$

显然，

$$t_0 = \frac{1}{\sqrt{N}}t^{\mathrm{I}}_0 = \frac{1}{\sqrt{N}}t^{\mathrm{I}}_0 \tag{15}$$

$l \neq 0$ 时将式(13)及式(14)代入式(10)中化简后有：

$$t_l = \frac{1}{2\sqrt{N}}\left[(1+j)\cdot t^{\mathrm{I}}_l \cdot W^{l}_{2N} - (1-j)t^{\mathrm{I}}_{N-l}\cdot u^{l}_{2N}\right] \tag{16}$$

则：

$$t_{N-l} = \frac{1}{2\sqrt{N}}\left[(1+j)\cdot t^{\mathrm{I}}_{N-l}\cdot W^{N-l}_{2N} - (1-j)t^{\mathrm{I}}_{l}\cdot u^{N-l}_{2N}\right] \tag{17}$$

请注意 $W^{l}_{2N} = e^{-\pi l/N} = \cos(\pi l/N) - j\sin(\pi l/N)$。

上面证明过程说明了 $\{T_k\}$ 序列的 $IDFT$ 可通过求 $\{T_k\}$ 序列的 $W\,\mathrm{II}$ 型变换并利用式(15)、(16)、(17)来求得。

2 新算法的推导及证明

新算法的基本思想是，在概念上仍利用 DFT 圆卷积特性，但计算时全采用实域变换实现。

现重新来研究两实序列 $x_n(n=0,1,\cdots,N-1)$，$h_m(m=0,1,\cdots,N-1)$ 的斜圆卷积：

$$y_l = \sum_{n=0}^{l} x_n\cdot h_{l-n} - \sum_{n=l+1}^{N-1} x_n\cdot h_{N+l-n},(l=0,1,\cdots,N-1) \tag{18}$$

的计算。这里 $h_m(m=0,1,\cdots,N-1)$ 预先固定，涉及有关该序列的所有运算可预先完成。

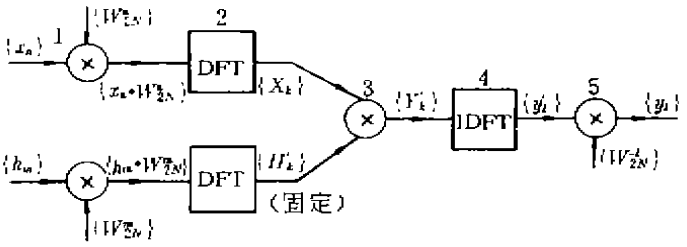

图 1 两实序列斜圆卷积的 DFT 计算法(其中 ⊗ 表示两序列对应点相乘，$W_{2N} = e^{-j\pi/N}$)

由引言可知实序列斜圆卷积(式(18))的常规算法如图 1 所示，其中 DFT 和 IDFT 均用 FFT 计算。由 1、2 小节可知实序列 $x_n(n=0,1,\cdots,N-1)$ 可由 $\{x_n\}$ 序列的 $W\,\mathrm{II}$ 型变换序列 $\{X^{\mathrm{II}}_k\}$ 简单构成。图 1 中第 3 步的计算(即 $Y_k = X_k\cdot H_k(k=0,1,\cdots\cdots,N-1)$ 也很简单，剩下最难的问题是如何用 W 变换完成图 1 中的第 4、5 步的计算，即如何由 $\{Y_k\}$ 序列快速求得

斜圆卷积输出序列$\{Y_k\}$，下面我们将证明：$T_k=\frac{1}{\sqrt{N}}\{\mathrm{Re}[Y_k]-\mathrm{Im}[Y_k]\}(k=0,1,\cdots,N-1$；Re[ ]，Im[ ]为取复数实、虚部）序列的W Ⅱ型变换序列$t_l(l=0,1,\cdots,N-1)$。就是待求的斜圆卷积输出序列$y_l(l=0,1,\cdots,N-1)$下面证明过程较复杂，基本思路是首先找到$\{T_k\}$序列的IDFT变换序列$\{t_l\}$与$\{y_l\}$间关系，从而确定$\{y_l\}$序列如何由$\{t_l\}$序列构成，又因$\{t_l\}$序列可由$\{t_l^1\}$序列构成，最后确定$\{y_l\}$序列与$\{t_l^1\}$序列关系。

由图1及DFT定义可知：

$$X_k=\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{n})\cdot W_N^{kn};\qquad H_k=\sum_{n=0}^{N-1}(h_m\cdot W_{2N}^{m})\cdot W_N^{km}$$

$$(k=0,1,\cdots,N-1;W_N=e^{-j2\pi/N};W_{2N}=e^{-j2\pi/2N}=e^{-j\pi/N})$$

$$Y_k=X_k\cdot H_k=\Big[\sum_{n=0}^{N-1}x_nW_{2N}^{n}\cdot W_N^{kn}\Big]\Big[\sum_{m=0}^{N-1}h_m\cdot W_{2N}^{m}\cdot W_N^{km}\Big]\qquad(k=0,1,\cdots,N-1)$$

$$=\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot W_N^{(m+n)(\frac{1}{2}-k)}\qquad(\because W_{2N}^{2}=W_N^{(1/2)z})$$

$$=\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\cos\Big[2\pi(m+n)(\frac{1}{2}+k)/N\Big]$$

$$-j\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\sin\Big[2\pi(m+n)(\frac{1}{2}+k)/N\Big]$$

$$T_k=\frac{1}{\sqrt{N}}\cdot\{\mathrm{Re}[Y_k]-\mathrm{Im}[Y_k]\}\qquad(k=0,1,\cdots,N-1)$$

$$=\frac{1}{\sqrt{N}}\cdot\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\{\cos[2\pi(m+n)(\frac{1}{2}+k)/N]+\sin[2\pi(m+n)(\frac{1}{2}+k)/N]\}$$

因为：

$$\cos(2\pi x/N)=\frac{1}{2}[e^{j2\pi x/N}+e^{-j2\pi x/N}]=\frac{1}{2}[W_N^{-x}+W_N^{x}]$$

$$\sin(2\pi x/N)=\frac{1}{2j}[e^{j2\pi x/N}-e^{-j2\pi x/N}]=\frac{j}{2}[W_N^{x}-W_N^{-x}]$$

$$T_k=\frac{1}{2\sqrt{N}}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m[W_N^{(m+n)(\frac{1}{2}+k)}+W_N^{-(m-n)(\frac{1}{2}+k)}+jW_N^{(m+n)(\frac{1}{2}+k)}-jW_N^{-(m+n)(\frac{1}{2}+k)}]$$

$$=\frac{1}{2\sqrt{N}}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot[(1+j)W_N^{(m-n)(\frac{1}{2}+k)}+(1-j)W_N^{-(m+n)(\frac{1}{2}+k)}]$$

$T_k(k=0,1,\cdots,N-1)$序列的IDFT为：

$$t_l=\frac{1}{N}\sum_{k=0}^{N-1}T_k\cdot W_N^{-lk}(l=0,1,\cdots N-1)$$

$$=\frac{1}{N}\sum_{k=0}^{N-1}\frac{1}{2\sqrt{N}}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot[(1+j)W_N^{(m+n)(\frac{1}{2}+k)}+(1-j)\cdot W_N^{-(m+n)(\frac{1}{2}+k)}]W_N^{-lk}$$

$$=\frac{1}{2\sqrt{N}}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\frac{1}{N}\sum_{k=0}^{N-1}(1+j)W_N^{(1/2)(m+n)}\cdot W_N^{(m+n)k}\cdot W_N^{-lk}$$

$$+\frac{1}{2\sqrt{N}}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\frac{1}{N}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}x_n\cdot h_m\cdot\frac{1}{N}\sum_{k=0}^{N-1}(1-j)W_N^{-(1/2)(m+n)}\cdot W_N^{-(m+n)k}\cdot W_N^{-lk}$$

$$=\frac{1+j}{2\sqrt{N}}\cdot\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}(x_n\cdot W_{2N}^{n})(h_m\cdot W_{2N}^{m})\frac{1}{N}\sum_{k=0}^{N-1}W_N^{(m+n-l)k}$$

$$+\frac{1-j}{2\sqrt{N}}\cdot\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}(x_n\cdot W_{2N}^{n})(h_m\cdot W^{-}m_{2N})\cdot\frac{1}{N}\sum_{k=0}^{N-1}W_N^{-(m+n+l)k}$$

因为：
$$\frac{1}{N}\sum_{k=0}^{N-1}W_N^{(m+n-l)k}=\begin{cases}1 & m+n-l\equiv 0\bmod N \text{ 或 } m\equiv l-n\bmod N\\ 0 & m,n,l \text{ 其它取值组合}\end{cases}$$

$$\frac{1}{N}\sum_{k=0}^{N-1}W_N^{(m-n-l)k}=\begin{cases}1 & m+n+l\equiv 0\bmod N \text{ 或 } m\equiv -l-n\bmod N\\ 0 & m,n,l \text{ 其它取值组合}\end{cases}$$

则：
$$t_l=\frac{1+j}{2\sqrt{N}}\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{n})\cdot(h_{(l-n)_N}\cdot W_{2N}^{(l-n)n})$$

$$+\frac{1-j}{2\sqrt{N}}\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{-n})\cdot(h_{(-l-n)_N}\cdot W_{2N}^{-(-l-n)n})(l=0,1,\cdots,N-1)$$

则：
$$t_0=\frac{1+j}{2\sqrt{N}}[x_0\cdot h_0+\sum_{n=1}^{N-1}x_n\cdot W_{2N}^{n}\cdot h_{N-n}\cdot W_{2N}^{N-n}]$$

$$+\frac{1-j}{2\sqrt{N}}[x_0\cdot h_0+\sum_{n=1}^{N-1}x_n\cdot W_{2N}^{-n}\cdot h_{N-n}\cdot W_{2N}^{-(N-n)}]$$

$$=\frac{1+j}{2\sqrt{N}}[x_0\cdot h_0-\sum_{n=1}^{N-1}x_n\cdot h_{N-n}]+\frac{1-j}{2\sqrt{N}}[x_0h_0-\sum_{n=1}^{N-1}x_n\cdot h_{N-n}]$$

$$t_0=\frac{1}{\sqrt{N}}y_0(\text{因 } y_0=x_0h_0-\sum_{n=1}^{N-1}x_n\cdot h_{N-n};W_{2N}^{N}=W_{2N}^{-N}=-1) \tag{19}$$

当 $l\neq 0$ 时，令 $t_l=\frac{1+j}{2\sqrt{N}}t_l^1+\frac{1-j}{2\sqrt{N}}t_l^2(l=1,2,\cdots,N-1)$，其中：

$$t_l^1=\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{n})\cdot(h_{(l-n)_N}\cdot W_{2N}^{(l-n)n})(l=1,2,\cdots,N-1)$$

$$t_l^2=\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{-n})\cdot(h_{(-l-n)_N}\cdot W_{2N}^{-(-l-n)n})(l=1,2,\cdots,,N-1) \tag{20}$$

因：
$$t_l^1=\sum_{n=0}^{l}x_n\cdot W_{2N}^{n}\cdot h_{2N}^{n}\cdot h_{l-n}\cdot W_{2N}^{l-n}+\sum_{n=l+1}^{N-1}x_n\cdot W_{2N}^{n}\cdot W_{N+l-n}\cdot W_{2N}^{N+l-n}$$

$$=W_{2N}^{l}(\sum_{n=0}^{l}x_n\cdot h_{l-n}-\sum_{n=l+1}^{N-1}x_n\cdot h_{N+(l-n)})=W_{2N}^{l}\cdot y_l$$

在式(20)中以 $N-l$ 代 l 有：

$$t_{N-l}^2=\sum_{n=0}^{N-1}(x_n\cdot W_{2N}^{-n})\cdot(h_{(l-n)_N}\cdot W_{2N}^{-(l-n)n})$$

$$=\sum_{n=0}^{l}x_n\cdot W_{2N}^{-n}\cdot h_{l-n}\cdot W_{2N}^{-(l-n)}+\sum_{n=l+1}^{N-1}x_n\cdot W_{2N}^{-n}\cdot h_{N+l-n}\cdot W_{2N}^{-(N+l-n)}$$

$$=W_{2N}^{-l}[\sum_{n=0}^{l}x_n\cdot h_{l-n}-\sum_{n=l+1}^{N-1}x_n\cdot h_{N+1-n}](\text{因 } W_{2N}^{-N}=-1)$$

$$=W_{2N}^{-l}y_l$$

则：
$$t_l^2=W_{2N}^{-(N-l)}\cdot y_{N-l}=-W_{2N}^{l}\cdot y_{N-l}$$

从而，当 $l\neq 0$ 时有：

$$t_l=\frac{1+j}{2\sqrt{N}}\cdot W_{2N}^{l}\cdot y_l-\frac{1-j}{2\sqrt{N}}W_{2N}^{l}\cdot y_{N-l}=\frac{1}{2\sqrt{N}}W_{2N}^{l}[(1+j)y_l-(1-j)y_{N-l}]$$

$$t_{N-l}=\frac{1}{2\sqrt{N}}W_{2N}^{N-l}[(1+j)\cdot y_{N-l}-(1-j)y_l]$$

则：

$$(1+j)y_l-(1-j)y_{N-l}=2\sqrt{N}W_{2N}^{l}\cdot t_l \tag{21}$$

$$(1+j)y_{N-l}-(1-j)y_l=2\sqrt{N}W_{2N}^{-(N-l)}\cdot t_{N-l} \tag{22}$$

由式(21)、(22)及式(19)有：

$$\begin{cases} y_l=\dfrac{\sqrt{N}}{1+j}[W_{2N}^{l}t_l-jW_{2N}^{-(N-l)}\cdot t_{N-l}] \\ (l=1,2,\cdots,N-1) \\ y_0=\sqrt{N}t_0 \end{cases} \tag{23}$$

由1.3小节已知$\{T_k\}$序列的IDFT序列$\{t_l\}$可由$\{T_k\}$序列的$W\mathrm{I}$型变换序列$\{t_l^{\mathrm{I}}\}$按下三式构成，即：

$$t_0=\frac{1}{\sqrt{N}}t_0^{\mathrm{I}} \tag{24}$$

$$t_l=\frac{1}{2\sqrt{N}}\cdot W_{2N}^{l}[(1+j)\cdot t_l^{\mathrm{I}}-(1-j)\cdot t_{N-l}^{\mathrm{I}}] \tag{25}$$

$$t_{N-l}=\frac{1}{2\sqrt{N}}W_{2N}^{N-l}[(1+j)t_{N-l}^{\mathrm{I}}-(1-j)\cdot t_l^{\mathrm{I}} \tag{26}$$

$$(l=1,2,\cdots,N-1)$$

显然$y_0=t_0^{\mathrm{I}}$；当$l\neq 0$时将式(25)、(26)代入式(23)中有：

$$y_l=\frac{1}{2(1+j)}[(1+j)t_l^{\mathrm{I}}-(1-j)t_{N-l}^{\mathrm{I}}-j[(1+j)t_{N-l}^{\mathrm{I}}-(1-j)t_l^{\mathrm{I}}]\}=t_l^{\mathrm{I}}$$

$$(l=1,2,\cdots,N-1)$$

到此，新算法中的最关键的证明已完结。

3　新算法的描述及运算量分析

由上两节的讨论，便易于给出用实域变换计算$N(N=2^M)$点实序列斜圆卷积的一种算法(图2)。

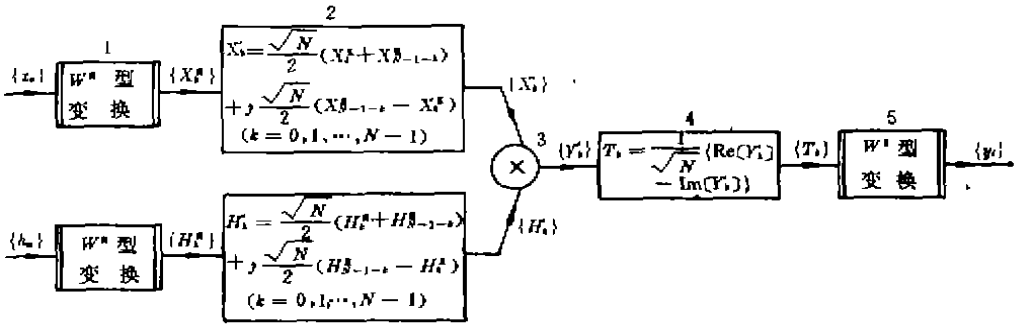

图2　用W变换计算$N(N=2^M)$点实序列斜圆卷积的一种算法

图2中算法由5阶段构成，其中第1、2阶段的计算与图1中算法的第1、2阶段的计算等价，

而第4、5阶段计算与图1中算法的第4、5阶段等价，由第1、2节的证明知道它是正确的。该算法简单易行且其运算量已比常规FFT计算法所需运算量有大幅度减少。但还可对它做下述三方面改进。一是$\{T_k\}$序列可直接由$\{x_k^R\}$、$\{H_k^R\}$序列构成；二是利用$\{h_m\}$序列预先固定的特点将乘常数运算预先合并到$\{H_k^R\}$序列中；三是利用$\{T_k\}$序列由$\{x_k^R\}$、$\{H_k^R\}$序列构成的过程具有同址运算特点能大幅度减少算法的内存需求。

因为$Y_k = X_k \cdot H_k (k = 0,1,\cdots,N-1)$且$X'_k, H'_k$分别由$\{X_k^R\}$、$\{H_k^R\}$序列构成，则有：

$$\begin{aligned} Y_k &= \left[\frac{\sqrt{N}}{2}(X_k^R + X_{N-1-k}^R) + j\frac{\sqrt{N}}{2}(X_{N-1-k}^I - X_k^I)\right] \\ &\quad \left[\frac{\sqrt{N}}{2}(H_k^R + H_{N-1-k}^R) + j\frac{\sqrt{N}}{2}(H_{N-1-k}^I - H_k^I)\right] \\ &= \frac{N}{4}[(X_k^R + X_{N-1-k}^R)(H_k^R + H_{N-1-k}^R) - (X_{N-1-k}^I - X_k^I)(H_{N-1-k}^I - H_k^I) \\ &\quad + j(X_k^R + X_{N-1-k}^R)(H_{N-1-k}^I - H_k^I) + j(X_{N-1-k}^I - X_k^I)(H_k^R + H_{N-1-k}^R)] \end{aligned}$$

$$T_k = \frac{1}{N}[\mathrm{Re}[Y_k] - \mathrm{Im}[Y_k]](k = 0,1,\cdots,N-1)$$

$$= \frac{\sqrt{N}}{2}[(X_k^R + X_{N-1-k}^R)H_k^R + (X_k^I - X_{N-1-k}^I) \cdot H_{N-1-k}^I]$$

令$\overline{H}_k^R = \frac{\sqrt{N}}{2} \cdot H_k^R (k = 0,1,\cdots,N-1)$，则

$$T_k = (X_k^R + X_{N-1-k}^R) \cdot \overline{H}_k^R + (X_k^I - X_{N-1-k}^I) \cdot \overline{H}_{N-1-k}^I \tag{27}$$

在式(27)中以$N-1-k$代k有：

$$T_{N-1-k} = (X_k^R + X_{N-1-k}^R) \cdot \overline{H}_{N-1-k}^R - (X_k^I - X_{N-1-k}^I) \cdot \overline{H}_k^I \tag{28}$$

这里应注意两点：一是当$\{X_k^R\}$、$\{\overline{H}_k^R\}$已知时，完成$T_k(k = 0,1,\cdots,N-1)$序列的构成仅需$2N$次实乘和$2N$次实加；二是这构成过程具有同址运算特点，即构成后序列$\{T_k\}$可放在$\{X_k^R\}$序列原有空间有。

由上面讨论，便可得到新算法的描述，如图2。它计算$N(N = 2^M)$点实序列斜圆卷积($\{h_m\}$固定)所需的运算量如表1所示。

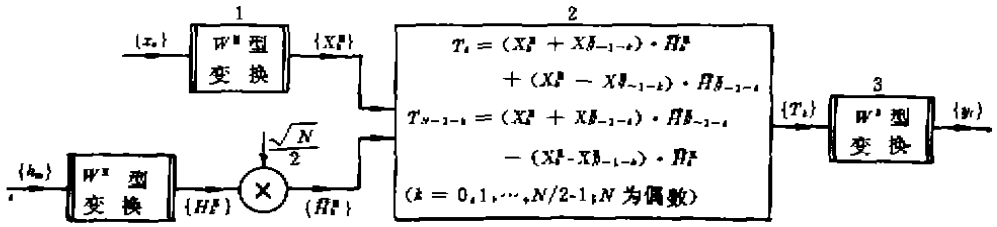

图3 用W变换计算$N(N = 2^M)$点实序列斜圆卷积的新算法

表1　新算法计算 $N(N=2^M)$ 点实序列斜圆卷积的运算量

运算量阶段	实数乘法次数	实数加法次数	参照
1	$\frac{N}{2}(\log_2 N-1)$	$\frac{3}{2}N\cdot(\log_2 N-1)$	结论 2.1
2	$2N$	$2N$	
3	$\frac{N}{2}(\log_2 N-1)$	$\frac{3}{2}N\cdot(\log_2 N-1)$	结论 2.1
总计	$N\cdot(\log_2 N+1)$	$3N(\log_2 N-1/3)$	

由前述，知道常规 FFT 计算法完成 $N(N=2^M)$ 点实序列斜圆卷积所需实数乘、加次数分别为：

$$M_{\mathrm{d}}^{FFT}(N)=4N(\log_2 N-1)+12;\mathrm{A}_{\mathrm{d}}^{FFT}(\mathrm{N})=6\mathrm{N}(log_2\mathrm{N}-\frac{1}{2})+6 \tag{29}$$

比较可知新算法计算 $N(N=2^M$ 较大时)点实序列斜圆卷积所需实数乘、加次数及所需存储单元数分别仅约为常规 FFT 计算法所需的 $\frac{1}{4}$、$\frac{1}{2}$、$\frac{1}{2}$。并且它极易实现，目前它将用于图象重建。

4　讨　　论

在本节讨论两个问题：一是对新算法给予验证，这对工程人员有益；二是讨论该新算法在二维实圆卷积快速计算中的重要作用。

4.1　新算法的验证

这里考虑两个4点实序列 $\{x_n\}=\{h_m\}=(1,2,3,4)$ 的斜圆卷积的计算，按定义计算有 $\{y_l\}=(-24,-20,-6,+20)$。

当 $N=4$ 时 W^{I} 型、W^{II} 型变换矩阵(定义1.2)为：

$$W_4^{\mathrm{I}}=\frac{1}{\sqrt{2}}\begin{bmatrix}\frac{\sqrt{2}}{2} & 1 & \frac{\sqrt{2}}{2} & 0\\ \frac{\sqrt{2}}{2} & 0 & -\frac{\sqrt{2}}{2} & 1\\ \frac{\sqrt{2}}{2} & -1 & \frac{\sqrt{2}}{2} & 0\\ \frac{\sqrt{2}}{2} & 0 & -\frac{\sqrt{2}}{2} & -1\end{bmatrix}\qquad W_4^{\mathrm{II}}=\frac{1}{\sqrt{2}}\begin{bmatrix}\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2}\\ 1 & 0 & -1 & 0\\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2}\\ 0 & 1 & 0 & -1\end{bmatrix}$$

$$\{X_l^{\mathrm{I}}\}=\{\overline{H}_l^{\mathrm{II}}\}=W_4^{\mathrm{I}}\cdot(1,2,3,4)^T$$

$$=\frac{1}{\sqrt{2}}(2+\sqrt{2},4-\sqrt{2},-2+\sqrt{2},-4-\sqrt{2})\left(\because\frac{\sqrt{4}}{2}=1\right)$$

$$\{T_n\}=(-10\sqrt{2}-15,-9+10\sqrt{2},-15+10\sqrt{2},-9-10\sqrt{2})$$

$\{T_k\}$ 序列的 W^1 型变换为：

$$W_4^1 \cdot (-10\sqrt{2}-15, -9+10\sqrt{2}, -15+10\sqrt{2}, -9-10\sqrt{2})^T$$
$$= (-24, -20, -6, 20) = \{y_l\}$$

从而验证算法正确。

4.2 新算法在二维实圆卷积中的重要作用

一个 $N \times N (N = 2^t)$ 的二维实圆卷积：

$$y_{n,l} = \sum_{m=0}^{N-1}\sum_{m=0}^{N-1} x_{k,m} \cdot h_{\langle l-m\rangle_N,\langle l-m\rangle_N} \quad (n,l = 0,1,\cdots,N-1;\{h_{r,s}\}\text{ 固定}) \tag{30}$$

用常规 FFT 计算所需复数乘、加次数为[2]：

$$M_n^{FFTC}(N) = 2N^2(\log_2 N - \frac{5}{2}) + 8N; A_l^{FFTC}(N) = 4 \cdot N^2\log_2 N$$

折合成实数乘、加次数为：

$$M_n^{FFTR}(N) = 8N^2(\log_2 N - \frac{5}{2}) + 64N;$$

$$A_l^{FFTR}(N) = 12N^2(\log_2 N - \frac{5}{6}) + 16N \tag{31}$$

利用快速多项式变换(FPT)计算式(30)，二维圆卷积通过中国剩余定理及多项式变换归结为求许多模 Z^M+1（M 为高复合数）的多项式积，由文献[1]中有关章节可得出 FPT 法计算式(30)所需实数乘、加次数 $M_n^{FPT}(N)$、A_l^{FPT} 的递推方程：

$$\begin{cases} M_n^{FPT}(N) = \text{计算}\frac{3}{2}N\text{ 个模 }Z^{N/2}+1\text{ 的多项式积所需乘法} \\ \quad (\text{每个多项式积中一个固定}) + M(N/2) \\ M_n^{FPT}(8) = 196 \end{cases}$$

$$\begin{cases} A_l^{FPT}(N) = \text{计算}\frac{3}{2}N\text{ 个模 }Z^{N/2}+1\text{ 的多项式积所需加法} \\ \quad + 2N^2 + \frac{N}{2}\cdot N\log_2 N + N^2 + \frac{N}{2}\cdot(\frac{N}{2}\cdot\log_2\frac{N}{2}) + A(N/2) \\ \quad = \text{计算}\frac{3}{2}N\text{ 个模 }Z^{N/2}+1\text{ 的多项式积所需加法} + \frac{3N^2}{4}(\log_2 N + \frac{11}{3}) + A(N/2) \\ A_l^{FPT}(8) = 1012 \end{cases}$$

上面初始条件为 *Agarwal-Cooly* 算法计算 8×8 实圆卷积所需乘、加次数[1]。

在引言中我们曾提到 N 点实序列斜圆卷积与模 Z^N+1 的多项式积实际上是同一物的两种提法，它们的计算是等价的。如将新算法的结果代入上面两组递推方程中有：

$$\begin{cases} M_n^{FPT}(N) = \frac{3}{2}N\cdot\frac{N}{2}(\log_2\frac{N}{2}+1) + M(N/2) = \frac{3N^2}{4}\log_2 N + M(N/2) \\ M_n^{FPT}(8) = 196 \end{cases} \tag{32}$$

$$\begin{cases} A_l^{FPT}(N) = \frac{3}{2}N\cdot\frac{3N}{2}(\log_2\frac{N}{2} - \frac{1}{3}) + \frac{3N^2}{4}(\log_2 N + \frac{11}{3}) + A(N/2) \\ \quad = 3N^2\cdot\log_2 N - \frac{1}{4}N^2 + A(N/2) \\ A(8) = 1012 \end{cases} \tag{33}$$

解方程可得：

$$M_1^{FPT}(N) < N^2\log_2 N + 196; A_1^{FPT}(N) < 4N^2\log_2 N + 1012 \quad (34)$$

比较式(34)与式(31)可看到；当N较大时，新算法与多项式变换法结合使用以计算二维实圆卷积所需实数乘、加次数分别约仅为常规FFT计算法所需实数乘、加次数的$\frac{1}{8}$和$\frac{1}{3}$，对于$N \times M(N = 2^{l_1}, M = 2^{l_2})$二维实圆卷积可得出同样的比较结果。该算法与多项式变换法结合使用可广泛用于需高速计算二维实圆卷积的领域，如卫星图象处理、由投影重建图象(*CT*)及遥感等方面。此外，即使从高度并行处理角度看，这种结合计算法也具有同样的优势，因为多项式变换计算法与常规*FFT*计算法同样具有高度的任务级并行性，多项式变换本身也易于并行[7]。

参 考 文 献

1 Nussbaumer H J. Fast Fourier Transform and Convolution Algorithms, Springer-Verlag. 1981.

2 蒋增荣. 二维数字卷积FPT算法及其在计算机上实现. 高等学校计算数学学报，1985，(3)：254～266

3 蒋增荣. 数论变换. 上海：上海科技出版社，1980

4 王中德. 快速变换的历史与现状. 电子学报，1989，(5)：103～111

5 王中德. 快速W变换—算法与程序. 中国科学(A辑)，1988，(5)：549～560

6 Kwong, C P. Shiu, K. P, Structured fast hartley transform, IEEE Trans. on ASSP., 1986, 34, 1000-1002

7 周六丁. 程代杰. 一种计算多项式变换的MIMD并行算法. 电子学报，1989，(6)：7～12