快速数论变换的一种算法

何永富　陈　涛　　王柏钧
（成都理工学院）　（成都气象学院）

【摘　要】

本文构造了一种快速数论变换算法，该算法是一种以数论为基础，有效高速的数论变换算法，并且这种算法易于在计算机上实现。

关键词： 数论；费马数变换；蝶形计算；码位倒读.
中图法分类： O174.22

AN ALGORITHM FOR FAST NUMBER THEORY TRANSFORM

He Yongfu　Chen Tao　　Wang Baijung
(Chengdu University of Technique)　(Chengdu Institute of Meteorology)

ABSTRACT

In this paper, an algorithm for fast number theory transform is constructed. The algorithm is based on number theory which enables us to derive an effective high speed algorithm for mumber theory transform. It is easy to carried out on thecomputer.

Key Words: *Number theory; Fermat number transform; Butterfly computation; Bit-reversal.*

1　概述

众所周知，在气象、通讯、雷达、遥感、地震、石油物探、自动控制、图象处理等方面，离散付里叶变换(DFT)得到了广泛应用。快速付里叶变换(FFT)的提出，大大减少了DFT的计算次数，乘法和加法计算次数的阶由原来的 $O(N^2)$ 减为 $O(N\log_2 N)$，从而大大节省了计算量。

但是，当数字信号序列长度 N 很大时，FFT的计算量仍然不能满足快速计算的要求。在保持循环褶积特性的前提下，快速数论变换是具有比FFT更快计算速度的快速变换算法之一。本文研究了进行快速数论变换的一种算法，对这一算法进行了严格推导，总结了它的一般规律，给出了程序步骤与框图。并在计算机上实现了计算。

本文于1992年5月3日收到.

2 数论变换

在离散正交变换的理论中，业已证明在复数域内，具有循环褶积特性的唯一变换是 DFT，所以，在复数域中不存在具循环褶积性质的更简单的离散正交变换。因此，人们提出了一种以数论为基础的具循环褶积性质的数论变换。

设正整数 M 有标准分解式 $M = p_1^{l_1}p_2^{l_2}\cdots p_s^{l_s}$（$p_i$ 为素数，l_i 为正整数），$Z_M = \{0,1,2,\cdots,M-1\}$ 是模 M 的非负最小完全剩余系，a 是模 M 的 N 阶本原单位根，且 a 对模 $p^i(i = 1,2,\cdots,s)$ 的阶数均为 N，设序列 $x(n) \in Z_M(n = 0,1,\cdots,N-1)$，则称

$$X(m) \equiv \sum_{n=0}^{N-1} x(n)a^{mn} \pmod{M} \qquad m = 0,1,\cdots,N-1 \tag{1.1}$$

为数论（正）变换，简记为 NTT。称

$$x(n) \equiv N^{-1}\sum_{m=0}^{N-1} X(m)a^{-mn} \pmod{M} \quad n = 0,1,2,\cdots,N-1 \tag{1.2}$$

为数论逆变换，简记为 INTT。式(1.2)中 N^{-1}, a^{-1} 分别是 N, a 的逆元素，即有 $NN^{-1} \equiv 1 \pmod{M}$，$aa^{-1} \equiv 1 \pmod{M}$。

根据

$$\sum_{n=0}^{N-1} a^{ln} \equiv \begin{cases} N, & l \equiv 0 \pmod{N} \\ 0, & l \not\equiv 0 \pmod{N} \end{cases} \pmod{M}$$

不难证明(1.1)式和(1.2)式确是构成一对互逆变换。为此，我们常将(1.1)式和(1.2)式所表达的一对变换简记为 $x(n) \rightleftharpoons X(m) \pmod{M}$。

当 $N = 2^t$ 时，与 DFT 类似，NTT 也有快速算法，称为快速数论变换，在数论变换中，当取模 M 为费马(Fermat)数

$$F_t = 2^b + 1 \qquad b = 2^t \quad (t = 1,2,\cdots) \tag{1.3}$$

时，这种数论变换叫做费马数变换(FNT)，在数论逆变换中取模 M 为费马数 F_t，该逆变换就称费马数逆变换(IFNT)。通常使用的数论变换，就是费马数变换。在这种变换中只需作加、减法及移位操作，不需要作乘法运算，因而极大地提高了计算速度，节省了机时。

与 DFT 类似，可以证明 NTT 也具有循环褶积特性，即如果 $x(n) \underset{N}{\rightleftharpoons} X(m) \pmod{M}$ $h(n) \underset{N}{\rightleftharpoons} H(m) \pmod{M}$，则循环褶积

$$y(n) = \sum_{l=0}^{N-1} x(l)h(<h-l>_N) \tag{1.4}$$

的 NTT 为

$$Y(m) \equiv X(m)H(m) \pmod{M} \tag{1.5}$$

在(1.4)中记号 $<n-l>_N$ 代表 $n-l$ 被 N 除所得的非负最小剩余。根据 NTT 的循环褶积特性，易知利用 NTT 计算循环褶积，与 DFT 类似，只需通过两次正变换和一次逆变换即可完成。

3 快速变换

现在构造 NTT 的快速算法，首先，给出两种特殊情形：

3.1 $N = 2^2$ 情形

假设变换参数为 $M, N = 2^2, a^N \equiv 1 \pmod M, a^{N/2} \equiv -1 \pmod M$，对 $x(n) \in Z_M (n = 0,1,2,3)$，此时变换公式(1.1)化为

$$X(m) \equiv \sum_{n=0}^{3} x(n) a^{mn} \pmod M \quad m = 0,1,2,3 \tag{2.1}$$

设 m, n 的二进制表示式为

$$m = 2m_1 + m_0 = (m_1, m_0)_2$$

$$n = 2n_1 + n_0 = (n_1, n_0)_2$$

其中，$m_i, n_i (i = 0,1)$ 等于 0 或 1，注意到 a 的假设，并令 $x_0(n) = x(n)$，有

$$\begin{aligned} X(m_1, m_0) &\equiv \sum_{n_0=0}^{1} \sum_{n_1=0}^{1} x_0(n_1, n_0) a^{(2m_1+m_0)(2n_1+n_0)} \\ &\equiv \sum_{n_0=0}^{1} \left\{ \sum_{n_1=0}^{1} x_0(n_1, n_0) a^{2n_1 m_0} \right\} a^{(2m_1+m_0)n_0} \pmod M \end{aligned} \tag{2.2}$$

于是，得以下递推关系

$$\begin{aligned} x_1(m_0, n_0) &\equiv \sum_{n_1=0}^{1} x_0(n_1, n_0) a^{2n_1 m_0} \\ &\equiv x_0(0, n_0) + x_0(1, n_0) a^{2m_0} \pmod M \end{aligned} \tag{2.3}$$

$$\begin{aligned} x_2(m_0, m_1) &\equiv \sum_{n_0=0}^{1} x_1(m_0, n_0) a^{(2m_1+m_0)n_0} \\ &\equiv x_1(m_0, 0) + x_1(m_0, 1) a^{2m_1+m_0} \pmod M \end{aligned} \tag{2.4}$$

显然，有

$$X(m_1, m_0) \equiv x_2(m_0, m_1) \pmod M \tag{2.5}$$

从(2.5)式看到，式子两端序号编码，关于二进制表示互为逆序。首先，我们给出输入为正序，输出为逆序的计算流程图。根据(2.3)式，有：

当 $n_0 = 0, m_0 = 0$ 时

$$x_1(0,0) \equiv x_0(0,0) + x_0(1,0) a^0 \pmod M \tag{2.6}$$

当 $n_0 = 0, m_0 = 1$ 时，

$$x_1(1,0) \equiv x_0(0,1) - x_0(1,0) a^0 \pmod M \tag{2.7}$$

当 $n_0 = 1, m_0 = 0$ 时，

$$x_1(0,1) \equiv x_0(0,1) + x_0(1,1) a^0 \pmod M \tag{2.8}$$

当 $n_0 = 1, m_0 = 1$ 时，

$$x_1(1,1) \equiv x_0(0,1) - x_0(1,1) a^0 \pmod M \tag{2.9}$$

以上四式构成计算流程图的第一次递推，如图 1 所示。

再根据(2.4)式构造第二次递推：

当 $m_0 = 0, m_1 = 0$ 时，

$$x_2(0,0) \equiv x_1(0,0) + x_1(0,1) a^0 \pmod M \tag{2.10}$$

当 $m_0 = 0, m_1 = 1$ 时，

$$x_2(0,1) \equiv x_1(0,0) - x_1(0,1) a^0 \pmod M \tag{2.11}$$

当 $m_0 = 1, m_1 = 0$ 时，

$$x_2(1,0) \equiv x_1(1,0) + x_1(1,1)\alpha^1 \pmod{M} \tag{2.12}$$

当 $m_0 = 1, m_1 = 1$ 时，

$$x_2(1,1) \equiv x_1(1,0) - x_1(1,1)\alpha^1 \pmod{M} \tag{2.13}$$

这四个式子构成了流程图的第二次递推。注意到输入为正序，输出为逆序，由公式(2.6)～(2.13)，我们编制出计算流程图 1。

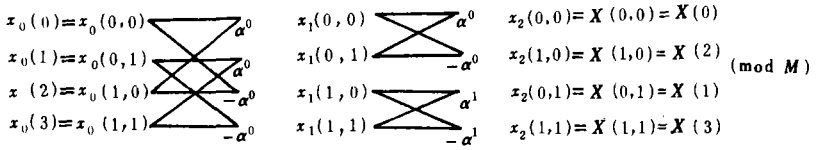

图 1　计算流程图

同样，根据(2.6)～(2.13)式，我们还可以构造如下输入为逆序，输出为正序的计算流程图 2。

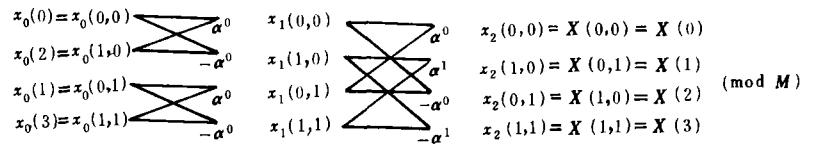

图 2　计算流程图

3.2　$N = 2^3$ 情形

假设变换参数为 $M, N = 2^3, \alpha^N \equiv 1 \pmod{M}, \alpha^{N/2} \equiv -1 \pmod{M}$，对 $x(n) \in Z_M (n = 0,1,\cdots 7)$，则变换公式(1.1)化为

$$X(m) \equiv \sum_{n=0}^{7} x(n)\alpha^{mn} \pmod{M} \quad m = 0,1,\cdots 7 \tag{2.14}$$

设 m, n 的二进制表示式为

$$m = 2^2 m_2 + 2m_1 + m_0$$

$$n = 2^2 n_2 + 2n_1 + n_0$$

其中，$m_i, n_i (i = 0,1,2)$ 等于 0 或 1，注意到 α 的假设，则有

$$\begin{aligned} X(m_2,m_1,m_0) &\equiv \sum_{n_0=0}^{1}\sum_{n_1=0}^{1}\sum_{n_2=0}^{1} x(n_2,n_1,n_0)\alpha^{(2^2m_2+2m_1+m_0)(2^2n_2+2n_1+n_0)} \\ &\equiv \sum_{n_0=0}^{1}\Big(\sum_{n_1=0}^{1}\Big(\sum_{n_2=0}^{1} x(n_2,n_1,n_0)\alpha^{4m_0n_2}\Big)\alpha^{(2m_1+m_0)2n_1}\Big)\alpha^{(2^2m_2+2m_1+m_0)n_0} \pmod{M} \end{aligned} \tag{2.15}$$

设 $x_0(n) = x(n)$，于是，得如下递推公式

$$x_1(m_0,n_1,n_0) \equiv \sum_{n_2=0}^{1} x_0(n_2,n_1,n_0)\alpha^{4m_0n_2}$$

$$\equiv x_0(0,n,n_0) + x_0(1,n_1,n_0)\alpha^{4n_0} \pmod{M} \tag{2.16}$$

$$x_2(m_0,m_1,n_0) \equiv \sum_{n_1=0}^{1} x_1(m_0,n_1,n_0)\alpha^{(2m_1+m_0)2n_1}$$

$$\equiv x_1(m_0,0,n_0) + x_1(m_0,1,n_0)\alpha^{2(2m_1+m_0)} \pmod{M} \tag{2.17}$$

$$x_3(m_0,m_1,m_2) \equiv \sum_{n_0=0}^{1} x_2(m_0,m_1,n_0)\alpha^{(2^2m_0+2m_1+m_0)n_0}$$

$$\equiv x_2(m_0,m_1,0) + x_2(m_0,m_1,1)\alpha^{2^2m_0+2m_1+m_0} \pmod{M} \tag{2.18}$$

因此有

$$X(m_2,m_1,m_0) \equiv x_3(m_0,m_1,m_2) \pmod{M} \tag{2.19}$$

根据(2.16)～(2.19),类似 $N = 2^2$ 情形的分析,我们可编出输入为正序,输出为逆序的计算流程图 3,和输入为逆序,输出为正序的计算流程图 4。

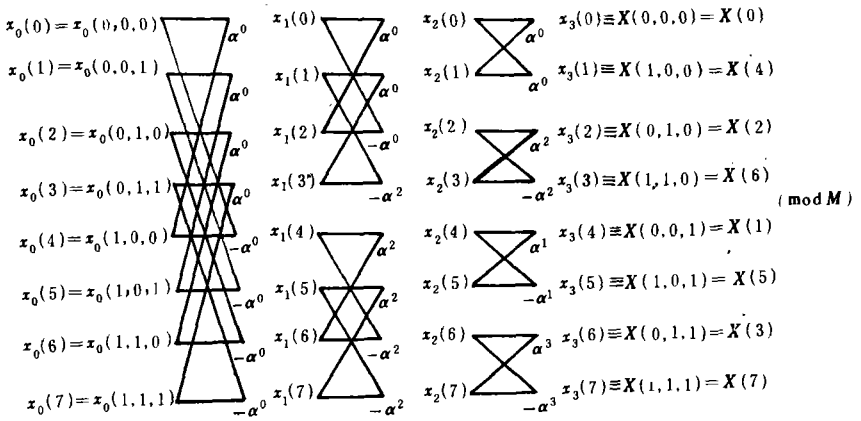

图 3 计算流程图

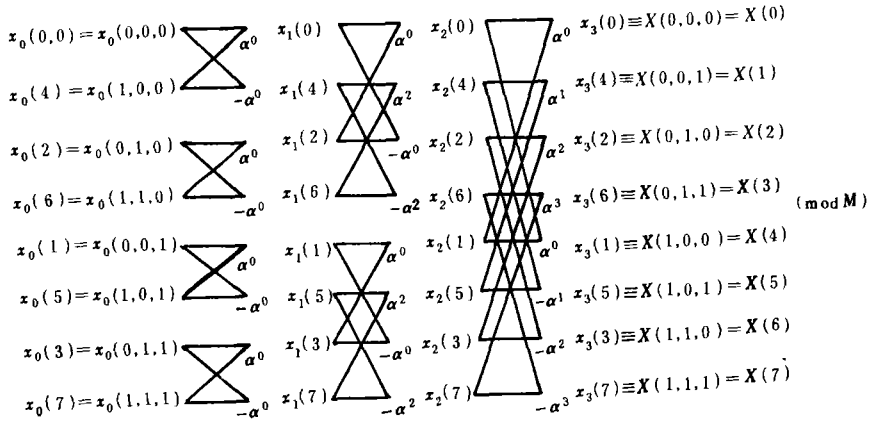

图 4 计算流程图

4 一般情形

假设变换参数为 $M, N = 2^l, a, a^N \equiv 1 \pmod M, a^{N/2} \equiv -1 \pmod M$。对于 $x(n) \in Z_M (n = 0,1,\cdots,2^l-1)$，NTT 为

$$X(m) \equiv \sum_{n=0}^{N-1} x(n) a^{mn} \pmod M \quad m = 0,1,\cdots,N-1 \tag{3.1}$$

设 m, n 的二进制表示为

$$m = \sum_{k=0}^{l-1} m_k 2^k = (m_{l-1}, m_{l-2}, \cdots m_0)_2$$

$$n = \sum_{k=0}^{l-1} n_k 2^k = (n_{l-1}, n_{l-2}, \cdots, n_0)_2$$

其中，$m_i, n_i = 0$ 或 $1 (i = 0,1,\cdots l-1)$。现研究乘积 mn，有

$$\begin{aligned} mn &= \sum_{k=0}^{l-1} m_k 2^k \sum_{k=0}^{l-1} n_k 2^k \\ &= (m_{l-1}2^{l-1} + \cdots + m_1 2^1 + m_0 2^0)(n_{l-1}2^{l-1} + \cdots + n_1 2^1 + n_0 2^0) \\ &= [n_{l-1}2^{l-1}(m_{l-1}2^{l-1} + \cdots + m_1 2^1) + \cdots + n_1 2^1 (m_{l-1}2^{l-1})] \\ &\quad + [n_{l-1}2^{l-1} m_0 + \cdots + n_0 (m_{l-1}2^{l-1} + \cdots + m_1 2^1 + m_0)] \end{aligned} \tag{3.2}$$

上式是通过对 n 进行分解后得到的。再注意到 $a^N \equiv 1 \pmod M$，由(3.2)式得

$$a^{mn} = a^{n_{l-1}2^{l-1}m_0 + \cdots + n_0(m_{l-1}2^{l-1} + \cdots + m_1 2^1 + m_0)} \tag{3.3}$$

将(3.3)式代入(3.1)式得：

$$\begin{aligned} X(m_{l-1},\cdots,m_0) &\equiv \sum_{n_0=0}^{1} \cdots \sum_{n_{l-1}=0}^{1} x(n_{l-1},\cdots n_0) a^{n_{l-1}2^{l-1}m_0 + \cdots + n_0(m_{l-1}2^{l-1} + \cdots + m_1 2 + m_0)} \\ &\equiv \sum_{n_0=0}^{1} \{\cdots (\sum_{n_{l-1}=0}^{1} x(n_{l-1},\cdots,n_0) a^{2^{l-1}m_0}) \cdots\} a^{n_0 \sum_{k=0}^{l-1} m_k 2^k} \pmod M \end{aligned} \tag{3.4}$$

令 $x(n_{l-1},\cdots,n_0) = x_0(n_{l-1},\cdots,n_0)$，由(3.4)得递推公式

$$\begin{aligned} &x_r(m_0,\cdots,m_{r-1},n_{l-r-1},\cdots n_0) \\ &\equiv \sum_{n_{l-1}=0}^{1} x_{r-1}(m_0,\cdots m_{r-2},n_{l-r},\cdots,n_0) a^{2^{l-r} n_{l-r} \sum_{k=0}^{r-1} m_k 2^k} \pmod M \quad r = 1,2,\cdots,l \end{aligned} \tag{3.5}$$

经过 l 层递推得到结果

$$X(m_{l-1},m_{l-2},\cdots,m_0) \equiv x_l(m_0,m_1,\cdots m_{l-1}) \pmod M \tag{3.6}$$

在每一层的计算中总可以找到这样的两点，它们的值仅是通过上一层中位置相同的一对结点运算得到。例如，第 r 层计算中 $x_r(m_0,\cdots,m_{r-2},0,m_{l-r-1},\cdots,m_0)$ 和 $x_{r-1}(m_0,\cdots,1,m_{l-2},1,n_{l-r-1},\cdots,n_0)$ 这对点仅是通过 $x_{r-1}(m_0,\cdots,m_{r-2},1,n_{l-r-1},\cdots n_0)$ 和 $x_{r-1}(m_0,\cdots,n_{r-2},1,n_{l-r-1},\cdots,n_0)$，这对位置相同的结点运算得到的。我们称这样的点为对偶结点。在每层递推中的任何一个点，都有对偶点组成对偶结点。

当输入为正序，输出为逆序的情况下，在第一层递推中，前 $N/2$ 个点以间距 $N/2$ 和后 $N/2$ 依次组成对偶结点，并且将第一层上的所有 N 个点看成一组。在第 r 层计算中，对偶结点间距为 $N/2^r (r = 1,2,\cdots l)$ 并将 N 个运算点分成 2^{r-1} 组，每组含 2^{l-r+1} 个点，组与组之间的组距为

$N/2^{r-1}$。

利用点的对偶性和对偶结点的间距，得到(3.5)式的一个便于计算的分式

$$x_r(m+\frac{jN}{2^{r-1}})=x_{r-1}(m+\frac{jN}{2^{r-1}})+a^p x_{r-1}(m+\frac{jN}{2^{r-1}}+\frac{N}{2^r}) \tag{3.7}$$

$$x_r(m+\frac{jN}{2^{r-1}}+\frac{N}{2^r})=x_{r-1}(m+\frac{jN}{2^{r-1}})-a^p x_{r-1}(m+\frac{jN}{2^{r-1}}+\frac{N}{2^r}) \tag{3.8}$$

$r=1,2,\cdots,l.\quad j=0,1,\cdots,2^{r-1}-1$

$m=0,1,\cdots,\frac{N}{2^r}-1,\quad p=2^{l-r}\times(2j)'$

其中，r 是递推次数，j 是分组序号，m 是半组元素排列序号，$(2j)'$ 表示 $2j$ 的二进制码位倒读。

根据(3.7)和(3.8)式，得如图 5 所示蝶形计算格式。

在图 5 中，图线含意如图 6 所示。

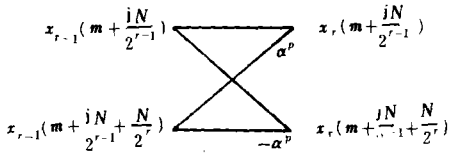

图 5 蝶形计算格式

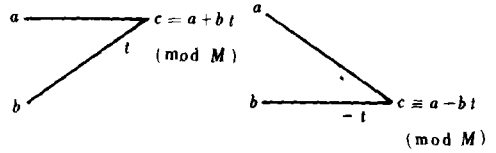

图 6 图 5 的图线含意

根据(3.5)～(3.8)式，既可构造输入为正序，输出为逆序的计算流程图，也可构造输入为逆序，输出为正序的计算流程图。

由以上讨论，下面按输入为正序，输出为逆序，总结出以下几条一般规律：

① 算出 $X(m)$ 共需 $l=\log_2 N$ 次递推。每次递推包含 2^{l-1} 个蝶形单元。第 r 次递推包含 2^{r-1} 个组，$r=0,1,\cdots,l$，而每组包含 2^{l-r+1} 个 $x_r(n)$。对偶结点间距为 2^{l-r}，组与组之间的组距为 2^{l-r-1}。

② 在第 r 次迭代时，第 j 组所乘 a 因子为 $a^{2^{l-r}\times(2j)'}$，其中，$(2j)'$ 为 $2j$ 的二进制码位倒读。在同一个组，前一半因子带正号，后一半因子带负号。比如 $N=2^3$ 时，在 $r=3,j=2$ 的情形，因 $(2j)'=1$，故所乘因子为 a^1。

③ 根据 $N=2^{l-1}$ 的计算流程图，可以按如下规律求得 $N=2^l$ 的计算流程图。第一，在 $N=2^{l-1}$ 流程图中，将每组所含 $x_r(n)$ 的个数加倍，将所有 a 因子的个数及其幂指数加倍，得到 $N=2^l$ 中前 $l-1$ 步递推的算法。第二，将新得到的第 $l-1$ 次递推中乘幂相同的 a 的个数减半，得第 l 次递推中前 2^{l-1} 个 a 因子，再将这些 a 的幂指数依次加 1，就得到后 2^{l-1} 个 a 因子。根据这个方法，我们就可以从 $N=2^2$ 的计算流程图 1，简单地构造出 $N=2^3$ 的计算流程图 3。

① 以上所述快速数论正变换的计算方法，也可以用来计算快速数论逆变换，只需在计算逆变换时，要以 a^{-1} 代替 a，并将最后递推的结果都乘以 N^{-1} 即可。

5 程序步骤与框图

输入为逆序，输出为正序的情形，具 a 的幂指数按自然顺序排列的特点，便于查找，所以这里我们仅给出这种情形程序步骤与框图。

5.1 程序步骤

① 输入变换长度 N，变换序列 $x(I)$，$I=1,N$，变换参数 M(模)，a(N 阶本原单位根)、INV(正逆变换方向)。

② 判断变换参数是否匹配，即是否满足 $a^N \equiv 1 \pmod M$。

③ 对正序输入的变换序列作反序处理。

④ 求递推次数 $MM(2^{MM}=N)$。

⑤ 作递推次数循环 $L=1,MM$。

a) 求第 L 次递推的基本 a 因子 KW。

若 INV=0，$KW=a^{N/2^L} \pmod M$

若 INV≠0，$KW=a^{-N/2^L} \pmod M$

b) 对第 L 次递推作蝶形计算 $L=1,MM$，KW0=1

作循环：

$$\begin{cases} J=1,2^{L-1} \\ \begin{cases} I=J,N,2^L \\ x(I)=x(I)+x(I+2^{L-1})\mathrm{KWO} \\ x(I+2^{L-1})=x(I)-x(I+2^{L-1})\mathrm{KW0} \end{cases} \\ \mathrm{KW0}=\mathrm{KW0}\cdot\mathrm{KW} \end{cases}$$

⑥ 若 INV=0，则返回，得正变换结果。

若 INV≠0，则求 $NINV=N^{-1} \pmod M$。然后，对序列每一个元素乘以 NINV，即得逆变换结果.

5.2 程序框图

程序框图请见图7。

5.3 算例

设 $N=2^4$　$a=2$　$M=257$，序列

$$\{x(n)\}_{n=1}^{16}=\{74,-38,45,41,76,92,-32,-18,-7,43,90,39,-57,-23,89,137\}$$

上机计算得结果

$$\{X(m)\}_{n=1}^{16}=\{37,81,86,-43,-50,-96,-55,104,5,54,-64,15,95,59,-32,-40\}$$

利用我们编的程序，对上述结果进行快速数论逆变换，即可验证上述这结果的正确性。

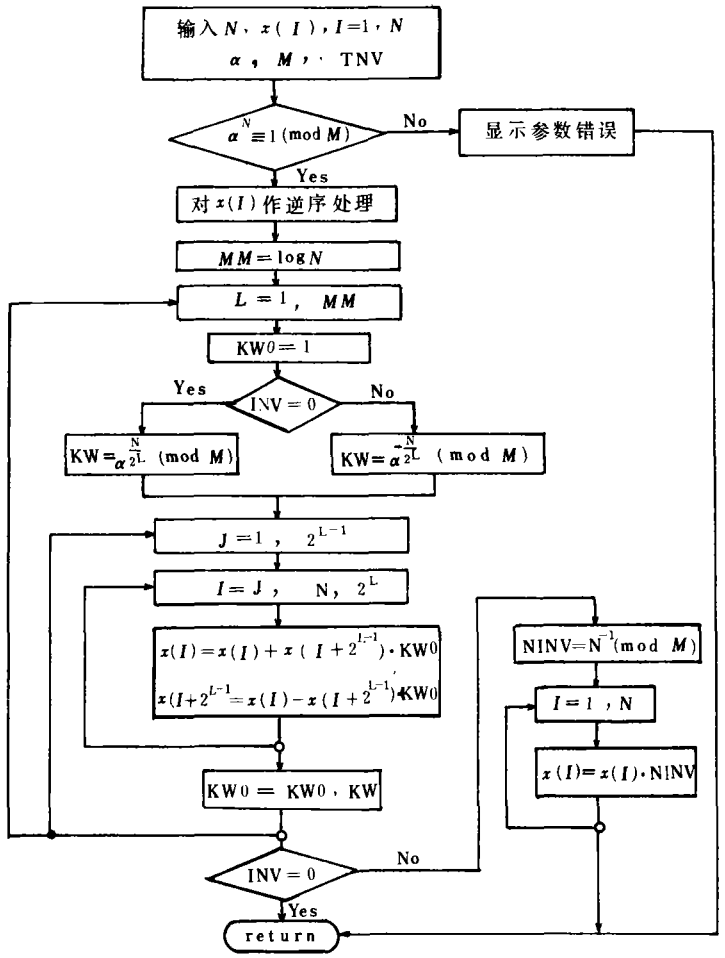

图 7 程序框图

参 考 文 献

1 孙琦，郑德勋，沈仲琦. 快速数论变换. 北京：科学出版社，1980

2 蒋增荣. 数论变换. 上海：科学技术出版社，1980

3 李金宗. 离散正交变换导论. 北京：高等教育出版社，1989

4 Meclellan J H and Rader C M. Number Theory in Digital processing. Prentice-Hall, Inc, Englewood cliffs, NJ, 1979

（编辑 沈镇芳）