Fast Algorithms for 2-D Circular Convolutions and Number Theoretic Transforms Based on Polynomial Transforms Over Finite Rings

Xiaonong Ran and K. J. Ray Liu, *Senior Member, IEEE*

***Abstract*—In this paper, we develop new fast algorithms for 2-D integer circular convolutions and 2-D number theoretic transforms (NTT). These new algorithms, which offer improved computational complexity, are constructed based on polynomial transforms over Z_p; these transforms are Fourier-like transforms over $Z_p[x]$, which is the integral domain of polynomial forms over Z_p. Having defined such polynomial transforms over Z_p, we prove several necessary and sufficient conditions for their existence. We then apply the existence conditions to recognize two applicable polynomial transforms over Z_p: One is for p equal to Mersenne numbers and the other for Fermat numbers. Based on these two transforms, referred to as Mersenne number polynomial transforms (MNPT) and Fermat number polynomial transforms (FNPT), we develop fast algorithms for 2-D integer circular convolutions, 2-D Mersenne number transforms, and 2-D Fermat number transforms. As compared to the conventional row–column computation of 2-D NTT for 2-D integer circular convolutions and 2-D NTT, the new algorithms give rise to reduced computational complexities by saving more than 25 or 42% in numbers of operations for multiplying 2^i, $i \geq 1$; these percentages of savings also grow with the size of the 2-D integer circular convolutions or the 2-D NTT.**

I. Introduction

In THE applications of image/video processing and coding, the computation of 2-D (linear) convolutions are of considerable importance [1], [2]. For instance, in the well-known subband image coding scheme, a full-band image is split into subbands by means of 2-D filter bank before the encoding operations [3]. Due to the huge number of data samples associated with these applications, direct-computation of 2-D convolutions is obviously impossible, and, thus, various fast algorithms are used [4], [5]. A popular technique for computing 2-D convolution is to convert the original 2-D convolution into a 2-D circular convolution, and to utilize fast transform algorithms to compute the 2-D circular convolution [4]. In this paper, we will develop new transform-based fast algorithms for the computation of 2-D circular convolutions.

In the digital signal processing applications, the values of signal samples are typically represented using a finite alphabet. Because of this finite resolution, the number system of finite rings, e.g., $Z_p = \{0, 1, \cdots, p-1\}$ [6], can be used for the calculations in digital signal processing. More precisely, in the case of a convolution[1]:

$$y_i = \sum_k h_k x_{i-k} \tag{1}$$

where h_k and x_i assume integer values, the output y_i also assume integer values. Supposing the absolute values of y_i are upper bounded by a positive number M, then if $M \leq p/2$, the output, $v_i = \sum_k h_k x_{i-k} \bmod p$, of the same convolution operated on Z_p can be translated uniquely to y_i: $y_i = v_i$ if $v_i \leq p/2$; $y_i = v_i - p$ if $v_i \geq p/2$. Performing the calculations such as convolutions on Z_p rather than on the ordinary integer set has several advantages:

i) The residue arithmetic associated with the operations on Z_p can be implemented relatively cheaply and performed very efficiently, especially in parallel and pipeline systems [7].
ii) There are no round-off errors.
iii) There exist very efficient algorithms (e.g., number theoretic transforms (NTT's)) [4], [5], [8], and more efficient algorithms, such as the ones for 2-D circular convolutions presented in this paper, can be developed.

In this paper, we focus on the development of polynomial transforms over a specific finite ring, namely, Z_p, [6] whose definition will be stated later on, and the fast algorithms associated with these transforms for 2-D circular convolutions. These polynomial transforms over Z_p are Fourier-like transforms over $Z_p[x]$ of polynomial forms [6]. An existence theorem for such Fourier-like transforms over arbitrary finite commutative rings with unity is given by Dubois and Venetsanopoulos [9] with applications for 1-D circular convolutions. However, the condition of the existence theorem in [9] is in general difficult to apply, due to the computational difficulty of various parameters. Maher later pointed out in [10] that, in most practical cases of concern, the rings may be characterized as algebraic extensions of finite rings, and that in these cases there is an existence theorem which is much easier to apply as compared to that of [9]. Maher also mentioned the computation of 2-D circular convolutions as an application for such Fourier-like transforms, motivated by polynomial transforms [11]; but the algorithm is very briefly described based on a special case,

Manuscript received August 26, 1992; revised May 31, 1994. This work was supported by the NSF NYI Award MIP9457397 and the ONR grant N00014-93-10566.

X. Ran is with Systems Technology, National Semiconductor Corp., Santa Clara, CA 95052 USA.

K. J. R. Liu is with the Electrical Engineering Department and Systems Research Center, University of Maryland, College Park, MD 20742 USA.

IEEE Log Number 9408213.

[1] The computational operations needed in a convolution are additions, subtractions, and multiplications.

and it is not clear whether or how much one can save in terms of computational complexity by using the new technique as compared with the existing algorithms such as 2-D NTT. In this paper, we will follow the general direction of above while concentrating on the application of 2-D circular convolutions. We will define a group of Fourier-like transforms over $Z_p[x]$, which are called polynomial transforms over Z_p, and will give several necessary and sufficient conditions for their existence. These conditions will be applied in a rather straight forward manner to obtain two specific groups of transforms, namely, Mersenne number polynomial transforms (MNPT's) and Fermat number polynomial transforms (FNPT's), which are of direct applications to the computation of 2-D circular convolutions. The complete algorithms will be provided along with the computational complexity analysis and comparisons against 2-D NTT. New fast algorithms for 2-D NTT are also developed based on MNPT and FNPT. The results of the complexity analysis show that new fast algorithms offer reduced complexities in terms of numbers of computational operations. More precisely, new fast algorithms give rise to savings on the numbers of the operation for multiplying a number by 2^i, $i \geq 1$; these savings are more than 25 or 42% (and are growing with the size of 2-D circular convolution or 2-D NTT) of the numbers of such operations in the conventional row–column computation of 2-D NTT. These complexity savings of the new algorithms are also confirmed by a simulation experiment on the actual computing time.

The paper is organized as follows. In Section II, polynomial transforms over Z_p are defined, their necessary and sufficient conditions of existence are stated, and then the MNPT and FNPT are introduced. Applications of the introduced transforms to the computations of 2-D circular convolutions and 2-D NTT are included in Sections III and IV. The analysis and comparisons of computational complexity are presented in Section V. Finally, Section VI contains a summary and conclusions.

II. Polynomial Transforms Over Finite Rings

In this section, we introduce the notations, define polynomial transforms over Z_p and provide several necessary and sufficient conditions for their existence.

We denote the set of all integers by Z, and the commutative ring: $\{0, 1, \cdots, p-1\}$ with addition and multiplication modulo the integer $p \geq 2$ by Z_p [6]. The equality of two numbers a and b in Z_p are denoted by $a \equiv b \bmod p$. We also use the notation $\langle a \rangle_p$ for the modulo p arithmetic on a, for example, $\langle 4 \rangle_3 = 1$. A polynomial $f(x)$ with its coefficients in Z_p is called a polynomial over Z_p. The set of all polynomials over Z_p is denoted by $Z_p[x]$ (which is an integral domain containing Z_p [6]). The equality of two polynomials $f(x)$ and $g(x)$ in $Z_p[x]$ are denoted by $f(x) \equiv g(x) \bmod p$. If $f(x), g(x) \in Z_p[x]$ are congruent modulo $h(x) \in Z_p[x]$, i.e., $f(x) - g(x) \equiv h(x)m(x) \bmod p$ for some $m(x) \in Z_p[x]$, we write

$$f(x) \equiv g(x) \bmod p, h(x). \tag{2}$$

We will use "|" to denote the divisibility, e.g., $a \mid b$ means $b = ac$ for integers a, b and c; and $a|b \bmod p$ indicates $b \equiv ac \bmod p$ for a, b and c in Z_p.

Definition II.1: Let $M(x), g(x) \in Z_p[x]$, and $\{H_m(x)\}_{m=0}^{N-1} \subset Z_p[x]$. We call

$$\bar{H}_k(x) \equiv \sum_{m=0}^{N-1} H_m(x)[g(x)]^{mk} \bmod p, M(x), \quad k = 0, 1, \cdots, N-1 \tag{3}$$

a polynomial transform over Z_p (or simply, a polynomial transform) of $\{H_m(x)\}_{m=0}^{N-1} \bmod p, M(x)$. Its inverse transform is defined by

$$H_l(x) \equiv N^{-1} \sum_{k=0}^{N-1} \bar{H}_k(x)[g(x)]^{-kl} \bmod p, M(x), \quad l = 0, 1, \cdots, N-1 \tag{4}$$

where $N^{-1}N \equiv 1 \bmod p$ and $[g(x)]^{-kl}[g(x)]^{kl} \equiv 1 \bmod p, M(x)$. We denote such a polynomial transform by $(N,\ g(x),\ p,\ M(x))$.

The following are three necessary and sufficient conditions for polynomial transforms over Z_p. Their proofs can be found in the Appendix.

Theorem II.2: Assuming p is prime, polynomial transform $(N,\ g(x),\ p,\ M(x))$ exists if and only if

$$S(q) \equiv \sum_{m=0}^{N-1} [g(x)]^{mq} = \begin{cases} N \bmod p, M(x) & \text{if } q \equiv 0 \bmod N \\ 0 \bmod p, M(x) & \text{if } q \not\equiv 0 \bmod N \end{cases} \tag{5}$$

and $(p, N) = 1$. Moreover, when the leading coefficient of $M(x)$ is the unity in Z_p, the above statement is also true for p not being prime.

Theorem II.3: Let $M(x) \equiv cb_1^{l_1}(x)b_2^{l_2}(x)\cdots b_s^{l_s}(x)$, $b_i(x)$, $1 \leq i \leq s$, be distinct irreducible polynomials with unity leading coefficients, and c be a nonzero constant. Then, $(N, g(x), p, M(x))$ exists if and only if (i) $[g(x)]^N \equiv 1 \bmod p, M(x)$, (ii) the order of $g(x) \bmod p, b_i(x)$ is N, for $1 \leq i \leq s$.

The next theorem is also a necessary and sufficient condition like the above two, but only dealing with a special case of Theorem II.3, when $M(x)$ is a product of distinct irreducible polynomials mod p.

Theorem II.4: Let $M(x) = b_1(x)b_2(x)\cdots b_s(x) \bmod p$, and $b_i(x)$ are distinct irreducible polynomials. Then, $(N, g(x), p, M(x))$ exists for some $g(x)$, if and only if N divides the greatest common divisor of $p^{n_1} - 1, p^{n_2} - 1, \cdots, p^{n_s} - 1$, where n_i is the degree of $b_i(x), 1 \leq i \leq s$.

The above necessary and sufficient conditions only tell us that if a given set of numbers and polynomials: $N, g(x), p$ and $M(x)$, satisfies certain conditions, then polynomial transform $(N, g(x), p, M(x))$ exists; they do not describe what these numbers and polynomials are. In the next two subsections, we introduce two groups of such numbers and polynomials, and then use the above conditions to show that they form polynomial transforms over Z_p.

A. Mersenne Number Polynomial Transforms

Theorem II.5: There is $(N, x, M_N, (x^N-1)/(x-1))$, for each prime number N and Mersenne number $M_N = 2^N - 1$.

Proof: Since $(x^N-1)/(x-1)$ is a factor of x^N-1, $S(q) \equiv \sum_{m=0}^{N-1} x^{mq} \equiv N \bmod M_N$, $(x^N-1)/(x-1)$, when $q \equiv 0 \bmod N$. On the other hand, when $q \not\equiv 0 \bmod N$, $\{\langle mq\rangle_N\}_{m=0}^{N-1}$ is a permutation of $\{m\}_{m=0}^{N-1}$, thus $S(q) \equiv 0 \bmod M_N, (x^N-1)/(x-1)$. We have $(M_N, N) = 1$ since $2^N - 2 \equiv 0 \bmod N$ by Fermat theorem [13], i.e., $N|(M_N-1)$. Thus, based on Theorem II.2, we have $(N, x, M_N, (x^N-1)/(x-1))$. □

The above polynomial transforms $(N, x, M_N, (x^N-1)/(x-1))$ are MNPT's whose applications will be described in the next two sections.

Example 1: For $N = 3$, we have $(3, x, 7, (x^3-1)/(x-1))$. Given polynomial sequence $H_0(x) = x+1$, $H_1(x) = x-1$ and $H_2(x) = x$, the corresponding MNPT are

$$\bar{H}_k(x) \equiv \sum_{m=0}^{2} H_m(x) x^{mk} \bmod 7, x^2+x+1 \tag{6}$$

for $k = 0, 1, 2$. The results are $\bar{H}_0(x) = 3x$, $\bar{H}_1(x) = 6x+1$ and $\bar{H}_2(x) = x+2$. It can be easily verified that

$$H_l(x) \equiv 3^{-1} \sum_{k=0}^{2} \bar{H}_k(x) x^{-kl} \bmod 7, x^2+x+1 \tag{7}$$

for $l = 0,1,2$, where $3^{-1} \equiv 5 \bmod 7$ and $x^{-1} \equiv (6x+6) \bmod 7, x^2+x+1$.

B. Fermat Number Polynomial Transforms

Theorem II.6: There are $(2^{t-i+1}, x, F_t, x^{2^{t-i}}+1)$, $0 \le i \le t$, and $(2^{t-i}, x^2, F_t, x^{2^{t-i}}+1)$, $0 \le i \le t-1$, for each positive number t and Fermat number $F_t = 2^{2^t}+1$.

Proof: We prove the case for $(2^{t-i+1}, x, F_t, x^{2^{t-i}}+1)$, $0 \le i \le t$; the other case can be similarly proved. When $q \equiv 0 \bmod 2^{t-i+1}$ since $(x^{2^{t-i+1}}-1) = (x^{2^{t-i}}+1)(x^{2^{t-i}}-1)$, we have

$$S(q) \equiv \sum_{m=0}^{2^{t-i+1}-1} x^{mq} \equiv 2^{t-i+1} \bmod F_t, x^{2^{t-i}}+1 \tag{8}$$

for $0 \le i \le t$. When $q \not\equiv 0 \bmod 2^{t-i+1}$, $S(q) \equiv (x^{2^{t-i+1}q}-1)/(x^q-1)$. Since $x^{2^{t-i+1}q}-1$ has factors x^q-1 and $x^{2^{t-i+1}}-1$, if we can show that x^q-1 and $x^{2^{t-i}}+1$ are coprime, then $S \equiv 0 \bmod F_t, x^{2^{t-i}}+1$. Indeed, we have $(q, 2^{t-i+1}) \le 2^{t-i}$ in this case. Thus, $(x^q-1, x^{2^{t-i+1}}-1) = x^{2^a}-1$ with $a \le t-i$ [13]. Therefore, x^q-1 and $x^{2^{t-i}}+1$ are coprime because $x^{2^a}-1$, $a \le t-i$, and $x^{2^{t-i}}+1$ are coprime (see cor. C.5 of [21]). In addition, we have $(2^{t-i+1})^{-1} \equiv 2^{2^{t+1}-(t-i+1)} \bmod F_t$. Based on Theorem II.2, $(2^{t-i+1}, x, F_t, x^{2^{t-i}}+1)$ exists for $0 \le i \le t$. □

The above polynomial transforms $(2^{t-i+1}, x, F_t, x^{2^{t-i}}+1)$, $0 \le i \le t$, and $(2^{t-i}, x^2, F_t, x^{2^{t-i}}+1)$, $0 \le i \le t-1$ are called Fermat FNPT's; their applications will be investigated in the next two sections.

Example 2: For $t = 1$ and $i = 0$, we have $(4, x, 5, x^2+1)$. Given polynomial sequence $H_0(x) = x+1$, $H_1(x) = x-1$, $H_2(x) = x$ and $H_3(x) = 1$, the corresponding FNPT are

$$\bar{H}_k(x) \equiv \sum_{m=0}^{3} H_m(x) x^{mk} \bmod 5, x^2+1 \tag{9}$$

for $k = 0, 1, 2, 3$. The results are $\bar{H}_0(x) = 3x+1$, $\bar{H}_1(x) = 3x$, $\bar{H}_2(x) = x+1$ and $\bar{H}_3(x) = 2x+2$. It can be easily verified that

$$H_l(x) \equiv 4^{-1} \sum_{k=0}^{3} \bar{H}_k(x) x^{-kl} \bmod 5, x^2+1 \tag{10}$$

for $l = 0, 1, 2, 3$, where $4^{-1} \equiv 4 \bmod 5$ and $x^{-1} \equiv 4x \bmod 5, x^2+1$.

III. Fast Algorithms for 2-D Circular Convolutions

We now use the polynomial transforms over Z_p to develop new fast algorithms for the computation of 2-D circular convolutions (CC). Let us consider a 2-D $N \times N$ CC

$$y_{l,u} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} h_{m,n} q_{\langle l-m\rangle_N, \langle u-n\rangle_N}, \quad l, u = 0, 1, \cdots, N-1 \tag{11}$$

of 2-D data $\{h_{i,j}\}$ and $\{q_{i,j}\}$, $i, j = 0, 1, \cdots, N-1$, which are assumed to be *integers* without loss of generality in digital signal processing practice. Apparently, this integer 2-D CC can be performed on Z_p if p is large enough; choices of p are given by

$$\frac{p}{2} > \min\left\{ |h_{m,n}|_{\max} \sum_{r=0}^{N-1} \sum_{s=0}^{N-1} |q_{r,s}|, \; |q_{r,s}|_{\max} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} |h_{m,n}| \right\}. \tag{12}$$

This 2-D integer CC can be also written as a 1-D polynomial CC

$$Y_l(x) \equiv \sum_{m=0}^{N-1} H_m(x) Q_{\langle l-m\rangle_N}(x) \bmod p, x^N-1 \tag{13}$$

where

$$\begin{aligned} H_m(x) &= \sum_{n=0}^{N-1} h_{m,n} x^n, \quad m = 0, 1, \cdots, N-1, \\ Q_r(x) &= \sum_{s=0}^{N-1} q_{r,s} x^s, \quad r = 0, 1, \cdots, N-1, \\ Y_l(x) &= \sum_{u=0}^{N-1} y_{l,u} x^u, \quad l = 0, 1, \cdots, N-1. \end{aligned} \tag{14}$$

The above conversion from 2-D CC to 1-D polynomial CC can be easily verified [12]. Thus, we can perform 2-D CC by evaluating the corresponding 1-D polynomial CC, which, in turn, can be computed using polynomials transforms over Z_p as described in the following.

A. Circular Convolution Property of Polynomial Transforms Over Finite Rings

Consider a 1-D polynomial CC as defined in (13) with $x^N - 1$ being substituted by $M(x)$:

$$Y_l(x) \equiv \sum_{m=0}^{N-1} H_m(x) Q_{\langle l-m\rangle_N}(x) \bmod p, M(x) \tag{15}$$

and assume that $N, g(x), p$ and $M(x)$ form a polynomial transform $(N, g(x), p, M(x))$. Define the corresponding transformed polynomials of $\{Y_i(x)\}, \{H_i(x)\}$ and $\{Q_i(x)\}$ by $\{\bar{Y}_i(x)\}, \{\bar{H}_i(x)\}$ and $\{\bar{Q}_i(x)\}$, respectively. Then

$$\begin{aligned}\bar{Y}_k(x) &\equiv \sum_{l=0}^{N-1} Y_l(x)[g(x)]^{lk} \bmod p, M(x)\\ &\equiv \sum_{l=0}^{N-1}\sum_{m=0}^{N-1} H_m(x) Q_{\langle l-m\rangle_N}(x)[g(x)]^{(l-m)k}\\ &\quad \times [g(x)]^{mk} \bmod p, M(x)\\ &\equiv \sum_{m=0}^{N-1} H_m(x)[g(x)]^{mk}\\ &\quad \times \sum_{l=0}^{N-1} Q_{\langle l-m\rangle_N}(x)[g(x)]^{(l-m)k} \bmod p, M(x)\\ &\equiv \bar{H}_k(x)\bar{Q}_k(x) \bmod p, M(x)\end{aligned} \tag{16}$$

where the fact $[g(x)]^N \equiv 1 \bmod p, M(x)$ from Theorem II.2 (see (A.1)) is used to recognize the summation $\sum_{l=0}^{N-1} Q_{\langle l-m\rangle_N}(x)[g(x)]^{(l-m)k}$ as $\bar{Q}_k(x)$, for $k = 0, 1, \cdots, N-1$. Thus, in this case, the 1-D polynomial CC in (15) can be computed by evaluating

i) two polynomial transforms for $\{\bar{H}_i(x)\}$ and $\{\bar{Q}_i(x)\}$
ii) N polynomial products $\{\bar{H}_i(x)\}\{\bar{Q}_i(x)\} \bmod p, M(x)$
iii) one inverse polynomial transform for $\{Y_i(x)\}$ from $\{\bar{Y}_i(x)\}$.

Using the above circular convolution property, we develop two classes of fast algorithms for 2-D integer CC, or equivalently for 1-D polynomial CC, in the next two subsections. The first class is related to MNPT, and the second to FNPT.

B. Fast Algorithms for 2-D Circular Convolutions Based on MNPT

For a Mersenne number M_N, we consider a 1-D polynomial CC of length N and modulo $M_N, x^N - 1$ as given in (13):

$$Y_l(x) \equiv \sum_{m=0}^{N-1} H_m(x) Q_{\langle l-m\rangle_N}(x) \bmod M_N, x^N - 1. \tag{17}$$

Because we do not have polynomial transform for this polynomial CC, we decompose it into two 1-D polynomial CC's as follows:

$$\begin{aligned}Y_{1,l}(x) &\equiv \sum_{m=0}^{N-1} H_{1,m}(x) Q_{1,\langle l-m\rangle_N}(x)\\ &\equiv Y_l(x) \bmod M_N, \frac{x^N-1}{x-1},\end{aligned}$$

$$\begin{aligned}Y_{2,l} &\equiv \sum_{m=0}^{N-1} H_{2,m} Q_{2,\langle l-m\rangle_N}\\ &\equiv Y_l(x) \bmod M_N, (x-1), \quad l = 0, 1, \cdots, N-1\end{aligned} \tag{18}$$

where

$$\begin{aligned}H_{1,m}(x) &\equiv H_m(x) \bmod M_N, \frac{x^N-1}{x-1};\\ Q_{1,r}(x) &\equiv Q_r(x) \bmod M_N, \frac{x^N-1}{x-1};\\ H_{2,m} &\equiv H_m(x) \bmod M_N, (x-1);\\ Q_{2,r} &\equiv Q_r(x) \bmod M_N, (x-1)\end{aligned} \tag{19}$$

for $m, r = 0, 1, \cdots, N-1$. Note that $\{Y_{2,l}\}$ is simply a 1-D integer CC of length N which can be computed using Mersenne Number Transform (MNT) [4]. The computation of $\{Y_{1,l}(x)\}$ can be done by using MNPT $(N, x, M_N, (x^N - 1)/(x-1))$ as follows:

$$\begin{aligned}\bar{Q}_k(x) &\equiv \sum_{r=0}^{N-1} Q_{1,r}(x) x^{rk} \bmod M_N, \frac{x^N-1}{x-1},\\ \bar{H}_k(x) &\equiv \sum_{m=0}^{N-1} H_{1,m}(x) x^{mk} \bmod M_N, \frac{x^N-1}{x-1},\\ \bar{Y}_k(x) &\equiv \bar{Q}_k(x)\bar{H}_k(x) \bmod M_N, \frac{x^N-1}{x-1},\\ &\qquad k = 0, 1, \cdots, N-1,\\ Y_{1,l}(x) &\equiv N^{-1} \sum_{k=0}^{N-1} \bar{Y}_k(x) x^{-lk} \bmod M_N, \frac{x^N-1}{x-1},\\ &\qquad l = 0, 1, \cdots, N-1.\end{aligned} \tag{20}$$

We recover $\{Y_l(x)\}$ based on a Chinese Remainder Theorem (CRT) on $Z_p[x]$ [21] from $\{Y_{1,l}(x)\}$ and $\{Y_{2,l}\}$:

$$\begin{aligned}Y_l(x) \equiv\ & Y_{1,l}(x)\left(N - \frac{x^N-1}{x-1}\right)N^{-1}\\ &+ Y_{2,l}\frac{x^N-1}{x-1}N^{-1} \bmod M_N, x^N - 1\end{aligned} \tag{21}$$

for $l = 0, 1, \cdots, N-1$. The above is summarized in Fig. 1, where the 2-D data $\{q_{r,s}\}$ are arranged into polynomial forms $Q_r(x)$ at the beginning of the algorithm (14).[2] Then $\{Q_{1,r}(x)\}$ and $\{Q_{2,r}\}$ are generated (19). The $\{Q_{2,r}\}$ are convolved with $\{H_{2,m}\}$ to obtain $\{Y_{2,l}\}$ using MNT as indicated above. Polynomials $\{Q_{1,r}(x)\}$ are transformed into $\{\bar{Q}_k(x)\}$ (20) among which $\bar{Q}_0(x)$ is the direct summation of $\{Q_{1,r}(x)\}$, and the others are computed first modulo $x^N - 1$, then modulo $(x^N - 1)/(x-1)$. The inverse polynomial transform procedure is similar to that for direct polynomial transform. The products $\{\bar{Q}_k(x)\bar{H}_k(x)\}$ can be efficiently computed by (i) computing them modulo $x - 2^j$, for $j = 1, \cdots, N-1$ and (ii) reconstructing based on the CRT on $Z_p[x]$. The proof and the details of the above procedures can be found in [21]. The following is an example of computing 2-D integer CC with

[2]Note that the data array $\{h_{m,n}\}$ are treated as the impulse response of the corresponding 2-D filter; thus all the polynomials related to $\{h_{m,n}\}$ and used in the algorithm are computed beforehand.

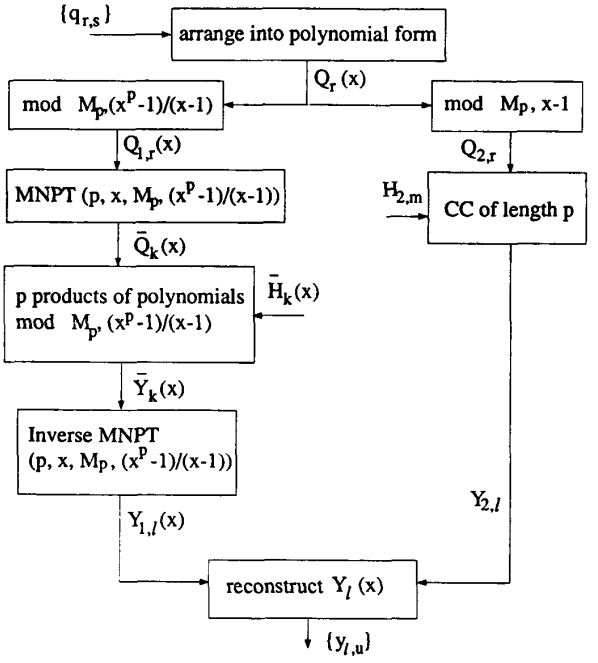


Fig. 1. Block diagram of the algorithm for computing 2-D integer circular convolution using MNPT.

the above fast algorithm based on MNPT. Referring (11), the 2-D data array are given in the following,

$$\{h_{m,n}\}: \begin{matrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{matrix} \quad \{q_{r,s}\}: \begin{matrix} 0 & 1 & -1 \\ 1 & -1 & 0 \\ -1 & 0 & 1 \end{matrix} \tag{22}$$

and we want to compute

$$y_{\ell,u} = \sum_{m=0}^{2}\sum_{n=0}^{2} h_{m,n} q_{\langle \ell-m\rangle_3, \langle u-n\rangle_3}$$

for $\ell, u = 0, 1, 2$. We choose $N = 3$ and $M_3 = 7$ for the modulo arithmetic, which satisfy the requirement (12). Now we use MNPT to compute $y_{\ell,u}$ mod 7 as follows, where the notations are based on the those defined above.

$$N^{-1}H_{2,m}: \begin{matrix} 2 & 5 & 5 \\ 0 & 1 & \end{matrix} \quad Q_{2,r}: \begin{matrix} 0 & 0 & 0 \\ 1 & 2 & \end{matrix}$$

$$H_{1,m}(x): \begin{matrix} -1 & 0 \\ -1 & -1 \end{matrix} \quad Q_{1,r}(x): \begin{matrix} 1 & -1 \\ -2 & -1 \end{matrix}$$

$$\bar{Q}_k(x): \begin{matrix} 0 & 0 \\ 3 & -1 \\ 0 & 0 \end{matrix} \quad \bar{H}_k(x): \begin{matrix} -2 & 0 \\ 0 & 1 \\ 2 & 2 \end{matrix} \tag{23}$$

$$T(x)N^{-1} \equiv (-x-2)5 \equiv 2(x+2) \bmod 7$$

$$\left(N^{-1}\right)^2 T(x)\bar{H}_k(x): \begin{matrix} 2 & 1 \\ 4 & 3 \\ -1 & 5 \end{matrix}$$

where $T(x) \equiv (N - \frac{x^N-1}{x-1})/(x-1)$, and the $(N^{-1})^2$ in front of $T(x)\bar{H}_k(x)$ are the N^{-1}s in the inverse polynomial transform (20) and in the CRT reconstruction (21); they are combined with $\{\bar{H}_k(x)\}$ to reduce the number of operations.

$$(N^{-1})^2 T(x)\bar{Y}_k(x) \equiv \bar{Q}_k(x)(N^{-1})^2 T(x)\bar{H}_k(x): \begin{matrix} 0 & 0 \\ 1 & 1 \\ 0 & 0 \end{matrix} \tag{24}$$

Then, we get

$$N^{-1}T(x)Y_{1,\ell}(x): \begin{matrix} 1 & 1 \\ 0 & -1 \\ -1 & 0 \end{matrix} \tag{25}$$

Obviously, we have $Y_{2,\ell}$: 000. Applying (21)

$$Y_\ell(x) \equiv N^{-1}T(x)Y_{1,\ell}(x)(x-1) + Y_{2,\ell}(x^2+x+1) \bmod 7, x^3-1$$

for $\ell = 0, 1, 2$, from which we obtain

$$\{y_{\ell,u}\}: \begin{matrix} -1 & 0 & 1 \\ 0 & 1 & -1 \\ 1 & -1 & 0 \end{matrix} \tag{26}$$

C. Fast Algorithms for 2-D Circular Convolutions Based on FNPT

For a Fermat number F_t, we consider a 1-D polynomial CC (13) of length $N = 2^{t+1}$ and modulo $F_t, x^N - 1$:

$$Y_l(x) \equiv \sum_{m=0}^{N-1} H_m(x) Q_{\langle l-m\rangle_N}(x) \bmod F_t, x^N - 1. \tag{27}$$

Define

$$\begin{aligned} Y_{1,l}(x) &\equiv \sum_{m=0}^{N-1} H_{1,m}(x) Q_{1,\langle l-m\rangle_N}(x) \\ &\equiv Y_l(x) \bmod F_t, x^{2^t}+1, \\ Y_{2,l}(x) &\equiv \sum_{m=0}^{N-1} H_{2,m}(x) Q_{2,\langle l-m\rangle_N}(x) \\ &\equiv Y_l(x) \bmod F_t, x^{2^t}-1, \\ & \qquad l = 0, 1, \cdots, N-1 \end{aligned} \tag{28}$$

where

$$\begin{aligned} H_{1,m}(x) &\equiv H_m(x) \bmod F_t, x^{2^t}+1, \\ Q_{1,r}(x) &\equiv Q_r(x) \bmod F_t, x^{2^t}+1, \\ H_{2,m}(x) &\equiv H_m(x) \bmod F_t, x^{2^t}-1, \\ Q_{2,r}(x) &\equiv Q_r(x) \bmod F_t, x^{2^t}-1, \\ & \qquad r, m = 0, 1, \cdots, N-1. \end{aligned} \tag{29}$$

$\{Y_l(x)\}$ can be computed from $\{Y_{1,l}(x)\}$ and $\{Y_{2,l}(x)\}$ by the CRT:

$$\begin{aligned} Y_l(x) \equiv\ & Y_{1,l}(x)2^{2^t-1}\left(x^{2^t}-1\right) \\ & + Y_{2,l}(x)2^{2^{t+1}-1}\left(x^{2^t}+1\right) \bmod F_t, x^{2^{t+1}}-1, \\ & \qquad l = 0, 1, \cdots, N-1. \end{aligned} \tag{30}$$

We use FNPT $(2^{t+1}, x, F_t, x^{2^t}+1)$ to compute $\{Y_{1,l}\}$:

$$\bar{Q}_k(x) \equiv \sum_{r=0}^{N-1} Q_{1,r}(x)x^{rk} \bmod F_t, x^{2^t}+1,$$

$$\bar{H}_k(x) \equiv \sum_{m=0}^{N-1} H_{1,m}(x)x^{mk} \bmod F_t, x^{2^t}+1,$$

$$\bar{Y}_k(x) \equiv \bar{Q}_k(x)\bar{H}_k(x) \bmod F_t, x^{2^t}+1, \quad k=0,1,\cdots,N-1,$$

$$Y_{1,l}(x) \equiv N^{-1}\sum_{k=0}^{N-1} \bar{Y}_k(x)x^{-lk} \bmod F_t, x^{2^t}+1, \quad l=0,1,\cdots,N-1 \quad (31)$$

where $N^{-1} \equiv (2^{t+1})^{-1} \equiv 2^{2^{t+1}-(t+1)} \bmod F_t$. For $\{Y_{2,l}\}$, we recognize that $\{Y_{2,l}\}_{l=0}^{N-1}$ corresponds to an $N \times (N/2)$ 2-D integer array with each row expressed in a polynomial form, and is the result of an $N \times (N/2)$ 2-D integer CC (see [21]); this $N \times (N/2)$ 2-D integer CC can be also expressed as an $(N/2) \times N$ 2-D integer CC and, thus can be written as a 1-D polynomial (of degree $N-1$) CC of length $N/2$, denoted as follows:

$$Y_l'(x) \equiv \sum_{m=0}^{2^t-1} H_m'(x)Q'_{\langle l-m\rangle_{2^t}}(x) \bmod F_t, x^{2^{t+1}}-1, \quad l=0,1,\cdots,2^t-1 \quad (32)$$

where the data array corresponding to $\{Y_l'(x)\}$, $\{H_m'(x)\}$, and $\{Q'_i(x)\}$ are just the transposes of those for $\{Y_{2,l}(x)\}$, $\{H_{2,m}(x)\}$, and $\{Q_{2,i}(x)\}$, respectively. Again, define

$$\begin{aligned} Y'_{1,l}(x) &\equiv \sum_{m=0}^{2^t-1} H'_{1,m}(x)Q'_{1,\langle l-m\rangle_{2^t}}(x) \\ &\equiv Y_l'(x) \bmod F_t, x^{2^t}+1, \\ Y'_{2,l}(x) &\equiv \sum_{m=0}^{2^t-1} H'_{2,m}(x)Q'_{2,\langle l-m\rangle_{2^t}}(x) \\ &\equiv Y_l'(x) \bmod F_t, x^{2^t}-1, \quad l=0,1,\cdots,2^t-1. \end{aligned} \quad (33)$$

$\{Y'_{1,l}(x)\}$ can be computed by FNPT $(x^2, 2^t, F_t, x^{2^t}+1)$:

$$\bar{Q}'_k(x) \equiv \sum_{r=0}^{2^t-1} Q'_{1,r}(x)x^{2rk} \bmod F_t, x^{2^t}+1,$$

$$\bar{H}'_k(x) \equiv \sum_{m=0}^{2^t-1} H'_{1,m}(x)x^{2mk} \bmod F_t, x^{2^t}+1,$$

$$\bar{Y}'_k(x) \equiv \bar{Q}'_k(x)\bar{H}'_k(x) \bmod F_t, x^{2^t}+1, \quad k=0,1,\cdots,2^t-1,$$

$$Y'_{1,l}(x) \equiv (2^t)^{-1}\sum_{k=0}^{2^t-1} \bar{Y}'_k(x)x^{-2lk} \bmod F_t, x^{2^t}+1, \quad l=0,1,\cdots,2^t-1. \quad (34)$$

$\{Y'_{2,l}(x)\}$ corresponds a $2^t \times 2^t$ 2-D integer CC and, thus, has a similar computation process as described above. The above computation procedures are summarized in Fig. 2. The procedures of FNPT and its inverse are based on a decomposition process using the fact that $x^{2^{t-i+1}} \equiv 1$, $x^{2^{t-i}} \equiv -1$, mod F_t, $x^{2^{t-i}}+1$. The products of two polynomials modulo $x^{2^t}+1$ are computed using FNT [4]. However, FNT can not be applied directly here; the polynomials in the product are modified by substituting x by $2^{2^i}y$. Then, the product of the modified polynomials are computed with FNT. The details of these procedures are in [21] and are omitted here to reduce the size of the paper.

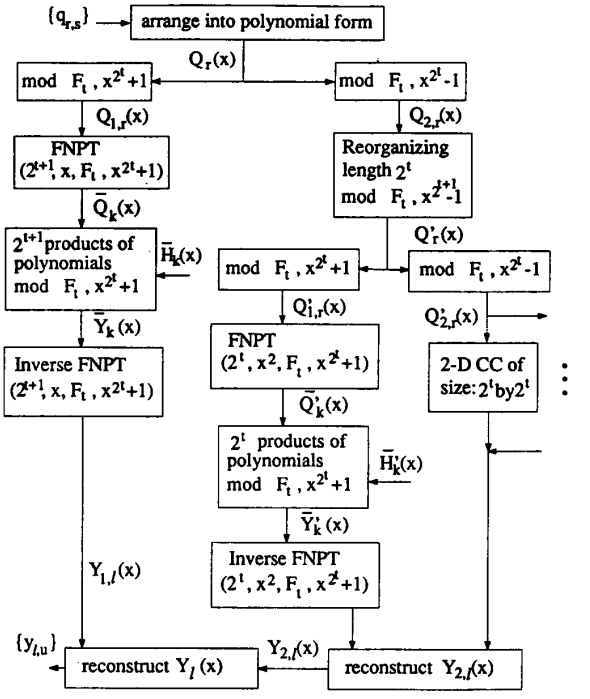


Fig. 2. Block diagram of the algorithm for computing 2-D integer circular convolution using FNPT.

IV. Fast Algorithms for 2-D Number Theoretic Transforms

In this section, we develop fast algorithms for direct computation of 2-D MNT and 2-D FNT's [4], [8] by using MNPT and FNPT. The traditional way for computing 2-D NTT of a 2-D data array is to apply the corresponding 1-D NTT to each row and then to each column (or vice versa) of the 2-D data array. The computational complexity comparisons of the new technique presented in this section and the traditional row–column scheme is given in the next section.

A. Fast Algorithm of 2-D Mersenne Number Transforms Using MNPT

Consider a 2-D MNT of size $N \times N$, where N is a prime number

$$\bar{Q}_{k_1,k_2} \equiv \sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} q_{n_1,n_2}2^{k_1n_1}2^{k_2n_2} \bmod M_N, \quad k_1,k_2=0,1,\cdots,N-1. \quad (35)$$

Define

$$Q_{n_1}(x) \equiv \sum_{n_2=0}^{N-1} q_{n_1,n_2}x^{n_2} \bmod M_N, \quad n_1 = 0,1,\cdots,N-1,$$
$$\bar{Q}_{k_1}(x) \equiv \sum_{n_1=0}^{N-1} Q_{n_1}(x)2^{n_1k_1} \bmod M_N, x^N-1, \quad k_1 = 0,1,\cdots,N-1 \quad (36)$$

then

$$\bar{Q}_{k_1,k_2} \equiv \bar{Q}_{k_1}(x) \bmod M_N, x-2^{k_2}, \quad k_2 = 0,1,\cdots,N-1. \quad (37)$$

When $k_2 = 0$

$$\bar{Q}_{k_1,0} \equiv \sum_{n_1=0}^{N-1}(Q_{n_1}(x) \bmod M_N, x-1)2^{n_1k_1} \bmod M_N, \quad k_1 = 0,1,\cdots,N-1 \quad (38)$$

which is a 1-D MNT.

When $k_2 \neq 0$, define

$$Q'_{n_1}(x) \equiv Q_{n_1}(x) \bmod M_N, \frac{x^N-1}{x-1}, \quad n_1 = 0,1,\cdots,N-1,$$
$$\bar{Q}'_{k_1}(x) \equiv \sum_{n_1=0}^{N-1} Q_{n_1}(x)2^{n_1k_1} \bmod M_N, \frac{x^N-1}{x-1}, \quad k_1 = 0,1,\cdots,N-1 \quad (39)$$

then

$$\bar{Q}_{k_1,k_2} \equiv \bar{Q}'_{k_1}(x) \bmod M_N, x-2^{k_2}, \quad k_2 = 1,2,\cdots,N-1. \quad (40)$$

Since $\{\langle k_2k_1\rangle_N\}$ is a permutation of $\{k_1\}$ when $k_2 \neq 0$

$$\bar{Q}'_{\langle k_1k_2\rangle_N}(x) \equiv \sum_{n_1=0}^{N-1} Q'_{n_1}(x)2^{n_1k_1k_2} \equiv \sum_{n_1=0}^{N-1} Q'_{n_1}(x)x^{n_1k_1} \bmod M_N, \frac{x^N-1}{x-1}, \quad k_1 = 0,1,\cdots,N-1 \quad (41)$$

which is an MNPT [21].

B. Computation of 2-D Fermat Number Transforms Using FNPT

Consider a 2-D FNT of size $2^{t+1} \times 2^{t+1}$

$$\bar{Q}_{k_1,k_2} \equiv \sum_{n_1=0}^{2^{t+1}-1}\sum_{n_2=0}^{2^{t+1}-1} q_{n_1,n_2}2^{k_1n_1}2^{k_2n_2} \bmod F_t, \quad k_1,k_2 = 0,1,\cdots,2^{t+1}-1. \quad (42)$$

Define

$$Q_{n_1}(x) = \sum_{n_2=0}^{2^{t+1}-1} q_{n_1,n_2}x^{n_2}, \quad n_1 = 0,1,\cdots,2^{t+1}-1,$$
$$\bar{Q}_{k_1}(x) \equiv \sum_{n_2=0}^{2^{t+1}-1} Q_{n_1}(x)2^{k_1n_1} \bmod F_t, x^{2^{t+1}}-1, \quad k_1 = 0,1,\cdots,2^{t+1}-1 \quad (43)$$

then

$$\bar{Q}_{k_1,k_2} \equiv \bar{Q}_{k_1}(x) \bmod F_t, x-2^{k_2}, \quad k_1,k_2 = 0,1,\cdots,2^{t+1}. \quad (44)$$

When $k_2 = 2u+1, u = 0,1,\cdots,2^t-1$

$$Q'_{n_1}(x) \equiv Q_{n_1}(x) \bmod F_t, x^{2^t}+1, \quad n_1 = 0,1,\cdots,2^{t+1}-1,$$
$$\bar{Q}'_{\langle k_1k_2\rangle_{2^{t+1}}}(x) \equiv \sum_{n_1=0}^{2^{t+1}-1} Q'_{n_1}(x)2^{k_1k_2n_1} \equiv \sum_{n_1=0}^{2^{t+1}-1} Q'_{n_1}(x)x^{k_1n_1} \bmod F_t, x^{2^t}+1, \quad k_1 = 0,1,\cdots,2^{t+1}-1. \quad (45)$$

When $k_2 = 2u, u = 0,1,\cdots,2^t-1$

$$\bar{Q}'_{k,2u}(x) \equiv \sum_{n_1=0}^{2^{t+1}-1}\sum_{n_2=0}^{2^t-1}(q_{n_1,n_2}+q_{n_1,n_2+2^t})2^{k_1n_1}2^{2un_2} \times \bmod F_t, \quad u,k_1 = 0,1,\cdots,2^{t+1}-1. \quad (46)$$

Define

$$Q_{n_2}(x) = \sum_{n_1=0}^{2^{t+1}-1}(q_{n_1,n_2}+q_{n_1,n_2+2^t})x^{n_1}, \quad n_2 = 0,1,\cdots,2^t-1,$$
$$\bar{Q}_{2u}(x) \equiv \sum_{n_2=0}^{2^t-1} Q_{n_2}(x)2^{2un_2} \bmod F_t, x^{2^{t+1}}-1, \quad u = 0,1,\cdots,2^t-1. \quad (47)$$

Then

$$\bar{Q}_{k_1,2u} \equiv \bar{Q}_{2u}(x) \bmod F_t, x-2^{k_1}, \quad k_1 = 0,1,\cdots,2^{t+1}-1. \quad (48)$$

In this case, for $k_1 = 2v+1, v = 0,1,\cdots,2^t-1$, we define

$$Q'_{n_2}(x) \equiv Q_{n_2}(x) \bmod F_t, x^{2^t}+1, \quad n_2 = 0,1,\cdots,2^t-1,$$
$$\bar{Q}'_{\langle 2uk_1\rangle_{2^{t+1}}} \equiv \sum_{n_2=0}^{2^t-1} Q'_{n_2}(x)x^{2un_2} \bmod F_t, x^{2^t}+1, \quad u = 0,1,\cdots,2^t-1 \quad (49)$$

where in the last equation is FNPT $(2^t, x^2, F_t, x^{2^t}+1)$. Then we compute [21]

$$\bar{Q}_{k_1,\langle 2uk_1\rangle_{2^{t+1}}} \equiv \bar{Q}'_{\langle 2uk_1\rangle_{2^{t+1}}}(x) \bmod F_t, x-2^{k_1}, \quad (50)$$

for $k_1 = 2v + 1$, $v = 0, 1, \cdots, 2^t - 1$. For $k_1 = 2v$, $v = 0, 1, \cdots, 2^t - 1$,

$$\bar{Q}_{2v,2u}(x) \equiv \sum_{n_1=0}^{2^t-1} \sum_{n_2=0}^{2^t-1} [(q_{n_1,n_2} + q_{n_1,n_2+2^t}) + (q_{n_1+2^t,n_2} + q_{n_1+2^t,n_2+2^t})] 2^{2vn_1} 2^{2un_2} \mod F_t, \quad v, u = 0, 1, \cdots, 2^t - 1 \quad (51)$$

which is a 2-D FNT of size $2^t \times 2^t$, and can be computed similarly.

V. Computational Complexity and Comparisons

The computational complexities for the algorithms in the last two sections will be described in terms of numbers of *multiplications* (M), *additions* (A) and *shifts* (S), where shifts are the operations of multiplying a number by 2^i for some $i \geq 1$. Note that this definition of shifts is slightly different from the regular one which corresponds to multiplying a number by 2. One shift ($\times 2^i$) here is actually i consecutive regular shifts. Therefore, we will treat shifts and additions as if they are in the same category in the following for simplicity.

Now we summarize the results of the analysis of computational complexities as follows, whose details can be found in [21]. To compute an $N \times N$ 2-D integer CC using MNPT, we need to perform the following numbers of operations:

$$\begin{aligned} M &= N^2, \\ S &= 2N^3 - 4N^2 + 2, \\ A &= 4N^3 - N^2 - 10N + 8. \end{aligned} \quad (52)$$

To compute a $2^{t+1} \times 2^{t+1}$, $t \geq 1$, 2-D integer CC using FNPT, we have the computational complexity:

$$\begin{aligned} M &= 1 + \sum_{q=0}^{t} 3 \times 2^{2q}, \\ S &= \sum_{q=0}^{t} 3 \times q \times 2^{2q}, \\ A &= \sum_{q=0}^{t} (3 \times q \times 2^{2q+2} + 2^{2q+4}). \end{aligned} \quad (53)$$

We compare the above computational complexities with those for computing the same 2-D integer CC's using the corresponding NTT with row–column scheme in Tables I and II. In Table I, the first two columns under "Parameters" are the sizes N of 2-D integer CC's and the Mersenne number $M_N = 2^N - 1$ used. The next two groups of three columns are the computational complexities for MNPT and 2-D MNT with row–column scheme, respectively. Notice that both algorithms need the same number of multiplications; using MNPT saves some numbers of shifts, listed in column "S" of the last three columns, and needs more numbers of additions which are listed as negative numbers in the last column "A". However, the saving on shifts for MNPT are much larger and growing faster with N than the corresponding spending on additions in terms of numbers of operations. To get an approximate overall comparison between these two algorithm, we subtract the extra-spendings on additions from the saving on shifts for MNPT and enter the resulting numbers in the last column under "$S + A$" along with their percentages with respect to the corresponding numbers of shifts for the row–column scheme. With the above simplification, we conclude that using MNPT saves more than 42% of shifts as compared with the other algorithm.

TABLE I
Comparison of Computational Complexity for Computing 2-D Integer Circular Convolutions Using MNPT and 2-D MNT with Row–Column Scheme

Parameters		MNPT			2-D MNT (row-column scheme)			Savings by using MNPT		
N	M_N	M	S	A	M	S	A	S	A	$S+A$
3	7	9	20	77	9	48	72	28	-5	23 (48%)
5	31	25	152	433	25	320	400	168	-33	135 (42%)
7	127	49	492	1261	49	1008	1176	516	-85	431 (43%)
13	8191	169	3720	8497	169	7488	8112	3768	-385	3383 (45%)
17	131071	289	8672	19201	286	17408	18496	8736	-705	8031 (46%)
19	524287	361	12276	26893	361	24624	25992	12348	-901	11447 (46%)

TABLE II
Comparison of Computational Complexity for Computing 2-D Integer Circular Convolutions Using FNPT and 2-D FNT with Row–Column Scheme

Parameters			FNPT			2-D FNT (row-column scheme)			Savings by using FNPT
t	F_t	2^{t+1}	M	S	A	M	S	A	S
1	5	4	16	12	128	16	16	128	4 (25%)
2	17	8	64	108	768	64	160	768	52 (33%)
3	257	16	256	684	4096	256	1088	4096	404 (37%)
4	65537	32	1024	3756	20480	1024	6272	20480	2516 (40%)

In Table II, which is similar to Table I in style, we compare the computational complexities of FNPT and 2-D FNT with row–column scheme. The numbers of multiplications and additions are the same for both algorithms, whereas less numbers of shifts are needed for FNPT; the savings are more than 25% and are growing when t is increasing (see Table II).

The computational complexity of an $N \times N$ 2-D MNT using MNPT is

$$\begin{aligned} S &= N^3 - 2N^2 + 1, \\ A &= 2N^3 - N^2 - 4N + 4 \end{aligned} \quad (54)$$

where S is half of the S in (52), and A is less than half of the A in (52) by an amount of $0.5N^2 - N$. The computational complexities[3], S and A, for computing the same $N \times N$ 2-D MNT using row–column scheme are halves of those for $N \times N$ 2-D integer CC's, and thus can be obtained directly from Table I. The conclusion for the computational complexity comparison is similar to the one for $N \times N$ 2-D integer CC's.

The computational complexity of a $2^{t+1} \times 2^{t+1}$, $t \geq 1$, 2-D FNT using FNPT are exactly halves of those in (53), and the savings of using FNPT with respect to the row–column

[3]There are no multiplications for computation of NTT.

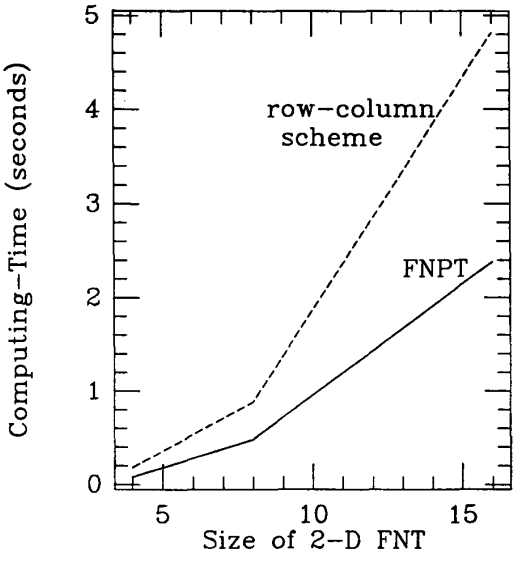


Fig. 3. Computing-time comparison for calculation of 2-D FNT using FNPT and the row–column scheme.

scheme are the same as those percentages in Table II. We have conducted a simulation experiment in which 2-D FNT are programmed in FORTRAN with the algorithm of FNPT and with the row–column scheme. We use general integer multiplications in FORTRAN to realize the shift operations in the algorithms. Since the general integer multiplication are more time-consuming than addition, the difference of the computing times for the above two programs to complete the same 2-D FNT can be an indicator for the difference of shifts used in the two algorithms. The resulting computing-times, on a personal computer, as a function of the sizes of 2-D FNT are shown in Fig. 3 which indicates that using FNPT reduces computing-times by about 50%. This matches with the above computational complexity assessment for these cases.

VI. Summary and Conclusion

In this paper, we developed new fast algorithms for 2-D integer circular convolutions and 2-D NTT. These new algorithms are constructed based on polynomial transforms over Z_p introduced here. Several necessary and sufficient conditions for the existence of polynomial transforms over Z_p are stated and proved. By applying these existence conditions, we have obtained two important polynomial transforms over Z_p: MNPT and FNPT, based on which we then developed fast algorithms for 2-D integer CC's, 2-D MNT and 2-D FNT. Comparing to the conventional row–column computation of 2-D NTT for 2-D integer CC's and 2-D NTT, the new algorithms save more than 25 or 42% of numbers of operations for multiplying 2^i, $i \geq 1$; these percentages of savings also grow with the size of the 2-D integer CC's or the 2-D NTT. These complexity savings of the new algorithms are also indicated by the computing-time results of a simulation experiment on computer.

Appendix A
Proof of Theorem II.2

For p being prime, the following is the "if" part:
When (5) holds (assuming $N > 1$)

$$[g(x)]^N - 1 \equiv (g(x) - 1) \sum_{k=0}^{N-1} [g(x)]^k \equiv 0 \bmod p, M(x). \tag{A.1}$$

Thus, $[g(x)]^{-1} \equiv [g(x)]^{N-1} \bmod p, M(x)$. Now, substitute (3) into (4), and define

$$R_l(x) \equiv \sum_{m=0}^{N-1} H_m(x) N^{-1} \sum_{k=0}^{N-1} [g(x)]^{k(m-l)} \bmod p, M(x) \tag{A.2}$$

for $l = 0, 1, \cdots, N-1$. From the conditions in the theorem, we have

$$R_l(x) \equiv H_l(x), \quad l = 0, 1, \cdots, N-1. \tag{A.3}$$

The following is the "Only if" part:

Suppose $(N, g(x), p, M(x))$ exists, i.e., (A.3) holds. Then the second case of (5) is true. Otherwise, there is some $t, 1 \leq t \leq N-1$, such that $S(t) \not\equiv 0 \bmod p, M(x)$. Then (A.3) can not always hold, e.g., let $H_t(x) \equiv 1$ and $H_i(x) \equiv 0$ for $i \neq t$, then (A.2) becomes

$$R_l(x) \equiv N^{-1} \sum_{k=0}^{N-1} [g(x)]^{k(t-l)} \bmod p, M(x) \tag{A.4}$$

for $l = 0, 1, \cdots, N-1$. Thus, $R_0(x) \equiv S(t) \not\equiv 0 \bmod p, M(x)$, and this is a contradiction. From the second case of (5) and (A.1), we have $[g(x)]^N \equiv 1 \bmod p, M(x)$, i.e., the first case of (5) holds. Finally, since the inverse transform exists, N^{-1} exists, thus $(p, N) = 1$, because, otherwise, $(p, N) = a > 1$, i.e., $p = b_1 a$ and $N = b_2 a$ for some b_1 and b_2; from $NN^{-1} \equiv 1 \bmod p$, we have $NN^{-1} - 1 = cp$ for some c, i.e., $b_2 a N^{-1} - 1 = cb_1 a$ or $(b_2 N^{-1} - cb_1)a = 1$ which is a contradiction.

When p is not a prime number, the whole proof is valid if we can carry out arithmetic modulo $M(x)$ on Z_p, and this is guaranteed if the leading coefficient of $M(x)$ is the unit of Z_p. □

Appendix B
Proof of Theorem II.3

The decomposition expression of $M(x)$ is supported [21]. From i), we get $S(0) \equiv N \bmod p, M(x)$. We denote the degree of $b_i(x)$ by n_i, for $1 \leq i \leq s$. Obviously, $n_i \geq 1$. From (ii), we have $N \mid p^{n_i} - 1$, i.e., $p^{n_i} - 1 = Nk_i$ [21]. Thus, p does not divide N, for otherwise, $N = pk_0$ for some k_0, and then, $p^{n_i} - 1 = pk_i k_0$, which is not true. When $q \not\equiv 0 \bmod N$, $([g(x)]^q - 1)S(q) = [g(x)]^{qN} - 1 \equiv 0 \bmod p, M(x)$. From (ii), we also have $([g(x)]^q - 1, M(x)) \equiv 1, \bmod p$. Thus, $S(q) \equiv 0 \bmod p, M(x)$, if $q \not\equiv 0 \bmod N$. Therefore, $(N, g(x), p, M(x))$ exists, from Theorem II.2.

On the other hand, when $(N, g(x), p, M(x))$ exists, from (5) and (A.1), we have $[g(x)]^N \equiv 1 \bmod p, M(x)$, $[g(x)]^N \equiv 1 \bmod p, b_i(x)$, and $S(q) \equiv 0 \bmod p, b_i(x)$, for $q \not\equiv 0 \bmod$

N. Thus, $[g(x)]^q - 1 \not\equiv 0$, mod $p, b_i(x)$, for $q \not\equiv 0$ mod N for otherwise, $p \mid N$, and this contradicts the requirement $(p, N) = 1$. □

APPENDIX C
PROOF OF THEOREM II.4

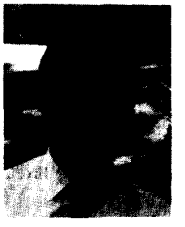
Suppose $(N, g(x), p, M(x))$ exists for some $g(x)$. Then, from Theorems II.3 we know $N \mid (p^{n_i} - 1), 1 \le i \le s$ [21]. Conversely, from N divides the greatest common divisor of $p^{n_1} - 1, p^{n_2} - 1, \cdots, p^{n_s} - 1$, we have $N \mid (p^{n_i} - 1), 1 \le i \le s$. For each i, there is $g_i(x)$ with order N mod $p, b_i(x)$ [21]. Thus, we have $g(x)$, such that, $g(x) \equiv g_i(x)$, mod $p, b_i(x)$, from the CRT. Since $[g_i(x)]^N \equiv 1$, mod $p, b_i(x)$, $[g(x)]^N \equiv 1$, mod $p, b_i(x)$. Thus $[g(x)]^N \equiv 1$, mod $p, M(x)$. Then based on Theorem II.3, we proved this Theorem. □

ACKNOWLEDGMENT

The first author would like to thank Prof. S. C. Zhou of Chongqing University, Chongqing, P.R.C., for his introduction of fast transforms and for his teaching and advice.

REFERENCES

[1] W. K. Pratt, *Digital Image Processing*. New York: Wiley, 1978.
[2] A. N. Netravali and B. G. Haskell, *Digital Pictures*. New York: Plenum, 1988.
[3] J. W. Woods, *Subband Image Coding*. Boston: Kluwer, 1991.
[4] R. E. Blahut, *Fast Algorithms for Digital Signal Processing*. Reading, MA: Addison-Wesley, 1985.
[5] J. G. Proakis, C. M. Rader, F. Y. Ling, and C. L. Nikias, *Advanced Digital Signal Processing*. New York: Macmillan, 1992.
[6] G. Birkhoff and S. MacLane, *A Survey of Modern Algebra*. New York: Macmillan, 1977.
[7] J.-B. Martens, "Number theoretic transforms for the calculation of convolutions," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-31, pp. 969–978, Aug. 1983.
[8] J. H. McClellan and C. M. Rader, *Number Theory in Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1979.
[9] E. Dubois and A. N. Venetsanopoulos, "The discrete Fourier transform over finite rings with application to fast convolution," *IEEE Trans. Comput.*, vol. C-27, pp. 586–593, July 1978.
[10] D. P. Maher, "On Fourier transforms over extension of finite rings," *IEEE Trans. Comput.*, vol. C-29, pp. 331–333, Apr. 1980.
[11] H. J. Nussbaumer and P. Quandalle, "Computation of convolutions and discrete Fourier transforms by polynomial transforms," *IBM J. Res. Develop.*, vol. 22, pp. 134–144, Mar. 1978.
[12] H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*. New York: Springer-Verlag, 1982.
[13] L.-K. Hua, *Introduction to Number Theory*. New York: Springer-Verlag, 1982.
[14] E. Dubois and A. N. Venetsanopoulos, "Convolution using a conjugate symmetry property for the generalized discrete Fourier transform," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-26, pp. 165–170, Apr. 1978.
[15] B. Arambepola and P. J. W. Rayner, "Discrete transforms over polynomial rings with applications in computing multidimensional convolutions," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-28, pp. 407–414, Aug. 1980.
[16] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1975.
[17] J.-L. Wu and Y. M. Huang, "Two-variable modularized fast polynomial transform algorithm for 2-d discrete Fourier transforms," *IEEE Trans. Circ. Syst. Video Technol.*, vol. 2, pp. 84–87, Mar. 1992.
[18] H. J. Nussbaumer, "Fast polynomial transform algorithms for digital convolution," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-28, pp. 205–215, Apr. 1980.
[19] ———, "New polynomial transforms for multidimensional DFT's and convolutions," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-29, pp. 74–84, Feb. 1981.
[20] X. Li and G. Qian, "Block size considerations for multidimensional convolution and correlation," *IEEE Trans. Signal Processing*, vol. 40, pp. 1271–1273, May 1992.
[21] X. Ran and K. J. R. Liu, "Fast algorithms for 2-D circular convolutions and number theoretic transforms based on polynomial transforms over finite rings," SRC Tech. Rep. TR 92-88, Univ. of Maryland, College Park, June 1992.



Xiaonong Ran received the B.S.E.E. and M.S.E.E. degrees from Chongqing University, China, in 1982 and 1984, respectively, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 1992.

During 1984–1986, he was a scientific researcher and lecturer in the Electrical Engineering Department of Chongqing University. During 1987–1992, he was a Research Assistant in System Research Center of UMCP; he received an SRC Fellowship, a Graduate Research Assistantship, and then a Graduate School Dissertation Fellowship. Since August 1992, he has been with the National Semiconductor Corp., Santa Clara, CA, where he is engaged in research and development on video coding and transmission with applications to multimedia and personal communication systems. He has two U.S. patents pending. His research interests include information theory, digital signal processing with applications to video coding and transmission, and fast algorithms.

K. J. Ray Liu (S'86–M'90–SM'93) received the B.S. degree from the National Taiwan University in 1983, the M.S.E. degree from the University of Michigan, Ann Arbor, in 1987, and the Ph.D. degree from the University of California, Los Angeles, in 1990, all in electrical engineering.

Since 1990, he has been an Assistant Professor in the Electrical Engineering Department and Institute for Systems Research, University of Maryland, College Park. His research interests span all the aspects of high performance computational signal processing, including parallel and distributed processing, fast algorithm, VLSI, and concurrent architecture, with application to image/video, radar/sonar, communications, and medical and biomedical technology.

Dr. Liu was the recipient of the IEEE Signal Processing Society's 1993 Senior Award and the 1994 National Science Foundation Young Investigator Award. He was awarded the George Corcoran Award for outstanding contributions to electrical engineering education at the University of Maryland. He has also received many other awards, including the Research Initiation Award from the National Science Foundation, the University Fellowship and the Hortense Fishbaugh Memorial Scholarship from UCLA, the President Research Partnership from the University of Michigan, and the Achievement Award in Science and Engineering from the Taiwanese–American Foundation. He is an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING and is a Member of the VLSI Signal Processing Technical Committee of the IEEE Signal Processing Society.