FAST ALGEBRAIC CONVOLUTION AND CORRELATION – PART I

Reiner Creutzburg¹, Harri Sorokin², Jarmo Takala³, and Torsten Minkwitz⁴

¹Fachhochschule Brandenburg – University of Applied Sciences
Department of Computing and Media, P. O. Box 2132
D–14737 Brandenburg, GERMANY, creutzburg@fh-brandenburg.de
²,³ Institute of Digital and Computer Systems, Tampere University of Technology,
P. O. Box 553, FIN–33101 Tampere, FINLAND, Harri.Sorokin@tut.fi, Jarmo.Takala@tut.fi
⁴Corporate Headquarter, Deutsche Telekom AG, Friedrich-Ebert-Allee 140
D–53113 Bonn, GERMANY

ABSTRACT

In this paper a new fast convolution algorithm is introduced. The concept of this new algorithm is fully algebraic. Therefore, no roundoff errors occur. The signal length \(N\) is a prime power \(N = p^n\), \(p \geq 2\). Hence the paper generalizes the results of previous papers [13,14,19] for power of 2 lengths. The cyclic convolution of signals of length \(N\) is interpreted as a multiplication of polynomials of degree \(N - 1\) modulo the polynomial \(X^N - 1\). The calculation in our new method is done in a finite field extension of the field \(\mathbb{Q}[\omega]\) chosen, with \(\omega\) as a symbolic root of unity and

\[ \text{deg}\left[\mathbb{Q}[\omega] : \mathbb{Q}\right] = \left\lfloor \frac{N}{\sqrt{N}} \right\rfloor \cdot \omega. \]

Here, \(\lceil \cdot \rceil^{*}\) and \(\lfloor \cdot \rfloor^{*}\) denote the well-known CEILING and FLOOR functions, respectively, but rather a mapping to the next larger power of \(p\). The method achieves an arithmetic cost of \(O(N \log N \cdot \log \log N)\) operations over \(\mathbb{Q}\), which is the same as that of the well-known algorithm by Schönhage-Strassen. However, our algorithm avoids the much more complicated FERMAT arithmetic over integers and works for primes \(p \neq 2\) as well, although it performs best with small \(p\), preferably \(p = 2\) or 3.

1. INTRODUCTION

The Chinese remainder theorem and existing FFT algorithms provide for mechanisms to develop divide and conquer strategies for algorithms in algebraic computing. Solving a problem of size \(N = p \cdot q\) is reduced to solving the same problem \(p\) times for size \(q\). If this reduction and its inverse can be done rapidly, then fast algorithms can be achieved. Convolutions are particularly well suited for such an approach, because the discrete Fourier transforms have the well-known cyclic convolution property.

A convolution of size \(\lceil N/2 \rceil\) can always be computed by a cyclic convolution of size \(N\). The higher order coefficients are set to zero and no actual overflow occurs.

The FFT is a the fast method of computing the DFT. It is based on the fact that the roots of unity form a cyclic group \(C_N\). Instead of taking the single group elements (roots of unity) of this large group, first the normal subgroup \(C_q(N = p \cdot q)\) is taken and the DFT is computed for the smaller group \(C_N/C_q\). The cosets are now used instead of the single roots of unity. Thereafter, the same algorithm is applied recursively for the cosets. Substituting the DFTs with FFTs in (Theorem 1.) yields an efficient method to do convolutions. In real applications the coefficients will often be elements of the ring \(\mathbb{Z}\) of integers. However, the maximal DFT size that can be defined in \(\mathbb{Z}\) is 2, since higher roots of unity are missing. Therefore, the integers are embedded in the field \(\mathbb{C}\) of complex numbers, which includes all roots of unity, or in a suitable number ring, which contains just the necessary roots of unity. The next section discusses some drawbacks of these approaches next to describing some alternative methods that avoid just those difficulties, but don’t achieve a comparable efficiency for large problem sizes.

If the vectors are interpreted as polynomials, their convolution is a polynomial multiplication. This interpretation will be used in the later sections to introduce a new algorithm, which does not suffer from any of the disadvantages that the others have. It uses the concept of finite field extensions to use a FFT type divide and conquer strategy.

2. PREVIOUS ALGORITHMS

A number of different methods have been designed to do convolutions [1-15,17-22]. If the coefficients are integers, two have been suggested in the introduction, which will be covered in this section. Besides the simple method which is well-known for polynomial multiplications, there are also more efficient algorithms that don’t require the concept of roots of unity. As an example the algorithm of Karatsuba is described here. Another algorithm that uses the convolution property of FFTs is the method described by Schönhage, which does convolutions over the ring of integers by mapping the vectors into large numbers and then doing fast multiplications of those large numbers.
2.1. Convolution in the field of complex numbers C

As was mentioned in the introduction, the convolution property of the FFT yields a fast algorithm since the necessary roots of unity exist in C. The calculations are done with floating point arithmetic. The problem comes from the fact that there is no exact mapping of all roots of unity into floating point numbers on a machine [15]. Also the use of floating point arithmetic results in roundoff errors. The convolution can therefore be only of limited precision. Another disadvantage of this method is that complex arithmetic with floating point numbers is usually slower than simple integer arithmetic. Therefore, the speedup achieved by this algorithm is somewhat diminished.

2.2. Convolutions using NTTs

Instead of using the ring of integers, which does not contain the needed roots of unity, a residue number system can be chosen, which does include those roots of unity. A residue number system is the ring of integers modulo \( m \). Let \( m = 2^n + 1 \), for some integer \( n \geq 0 \). Then the ring of residues modulo \( m \) is called a FERMAT ring [3-13,20-22]. If the maximum coefficient size is \( l \) bit and a convolution of size \( N \) is done, then the resulting coefficients have sizes smaller or equal than \( 2l + \log N \) bit. Now choose \( n \) to be so that \( 2^n > 2l + \log N \). It can be shown that the DFT exists for sizes \( N = 2^n \), \( 0 \leq i \leq (n + 1) \). In the ring of integers modulo \( m = 2^n + 1 \) for an integer \( n \geq 0 \), the elements \( 2^i \), \( 0 \leq i \leq n \), are primitive \((2^{n+1})\)th roots of unity modulo \( m \), respectively.

Now a FFT type algorithm can be used. The corresponding transforms are called number theoretic transforms or NTTs [3-13,20]. Other rings than FERMAT rings must be used if the convolution is required to have a size which is not a power of two. There are two drawbacks of this method. Although the multiplication with roots of unity are mere shift operations on a binary machine, the arithmetic that must be used is rather cumbersome on a general purpose machine. Also the size of the ring and therefore the complexity of every operation increases with the size of the convolution, since the necessary roots of unity must be included.

2.3. Karatsuba’s Algorithm

This method uses a simple recursion scheme [17]. The vectors are interpreted as polynomials. The formula for a polynomial multiplication of length \( N \) is:

\[
A \ast B = \sum_{i=0}^{2N-2} (\sum_{j+k=i} a_j \cdot b_k) X^i.
\]

The vectors \( A \) and \( B \) can be split into upper halves \( A_1 \) and \( B_1 \), (coefficients \( a_i, b_i, N/2 \leq i < N \)) and lower halves \( A_0 \) and \( B_0 \), (coefficients \( a_i, b_i, 0 \leq i < \lfloor N/2 \rfloor \)). Now the multiplication formula is

\[
A \ast B = (A_1 \ast B_1)X^{2\lfloor N/2 \rfloor} + (A_1 \ast B_0 + A_0 \ast B_1)X^{\lfloor N/2 \rfloor} + (A_0 \ast B_0)
\]

which is equivalent to

\[
A \ast B = (A_1 \ast B_1)(X^{2\lfloor N/2 \rfloor} + X^{\lfloor N/2 \rfloor})
\]

\[
+((A_1 - A_0) \ast (B_0 - B_1))X^{\lfloor N/2 \rfloor}
\]

\[
+ (A_0 \ast B_0)(X^{\lfloor N/2 \rfloor} + 1).
\]

The last formula shows how one multiplication of length \( N \) is reduced to only 3 multiplications of length \( N/2 \) instead of 4 in the previous formula. The scheme can be applied recursively. Therefore, instead of \( O(N^2) \), this method computes polynomial multiplications of length \( N \) with \( O(N \log_3 2) \approx O(N^{1.585}) \) elementary multiplications. For small \( N \), this method is faster than the others described here.

2.4. Sch"onhage’s Algorithm

Instead of explicitly doing a convolution, here the coefficients are mapped into a large integer and the convolution is done by multiplying those large integers. If the maximum coefficient is \( l \) bits long and the convolution is of length \( N \), then the numbers to be multiplied are:

\[
A' = \sum_{i=0}^{N-1} a_i \cdot 2^{i(2l + \log(N) + 1)}
\]

\[
B' = \sum_{i=0}^{N-1} b_i \cdot 2^{i(2l + \log(N) + 1)}
\]

These numbers \( A' \) and \( B' \) are multiplied by the efficient SCH"ONHAGE-STRASSEN algorithm and the result of the convolution can be read from the result of the multiplication of numbers. The \((i)\)th coefficient is in the \( 2l + \log(N) + 1 \) bits beginning at the \((i(2l + \log(N) + 1))\)th bit. For a fixed maximum size of resulting coefficients, this method has a cost of \( O(N \log N \cdot \log \log N) \) operations. The disadvantage it has in comparison with the algorithm introduced in the following section is the cumbersome arithmetic used in the SCH"ONHAGE-STRASSEN algorithm [21-22].

3. CONVOLUTION WITH FINITE FIELD EXTENSIONS

Instead of looking at direct convolution, the rest of this paper will deal with cyclic convolution of lengths \( N = p^n \), \( n \) an integer and \( p \) a prime, which are polynomial multiplications modulo the polynomial \( X^n - 1 \), [1,3-6,8,19-20]. These can be used to do direct convolution of length \( \lfloor N/2 \rfloor \). The factorization of \( X^n - 1 \) yields (see fig. 1)

\[
X^n - 1 = (X^{p^{n-1}} - 1)P_n(X)
\]

with

\[
P_n(X) = \sum_{i=0}^{p-1}(X^{ip^{n-1}}).
\]

If a fast method is known to perform convolution modulo \( P_n(X) \), the Chinese remainder theorem suggests a fast algorithm to perform cyclic convolution.
For signal lengths $N = 2^n$ the well-known factorization of $X^N - 1$ is much simpler and shown in fig. 2.

3.1. Extending the field of rational numbers $\mathbb{Q}$

The field $\mathbb{Q}$ of rational numbers does not include the necessary roots of unity to do cyclic convolutions of lengths greater than 2, [19]. Therefore, $P_n(X)$ is irreducible over $\mathbb{Q}$.

**Definition 1** Let $k \in \mathbb{Z}$ and $p$ be a prime. Then $\lfloor k \rfloor^*$ denotes the mapping to the next smaller power of $p$ with regard to $k$ and $\lceil k \rceil^*$ denotes the mapping to the next greater power of $p$ with regard to $k$.

In order to improve the situation, calculations will be done in $\mathbb{Q}(\omega(\sqrt[N]{N})^*)$, where $\omega$ is an $(N)$th root of unity. $P_n(X)$ is reducible over this field. It now splits into

$$P_n(X) = \prod_{i=1}^{\mu=\lceil \sqrt{N} \rceil^*} (X^{\sqrt{N}} - (\omega^{\sqrt{N}})^{\mu^{-1}}),$$

where $\alpha$ is a $(p - 1)$-th primitive root of unity in $\mathbb{Z}_p$. The Chinese remainder theorem shows that doing multiplications modulo $P_n(X)$ can be performed by calculating the remainders modulo the factors of $P_n(X)$, doing separate multiplications modulo these and, by interpolation, calculating the product. Since the original coefficients were rationals, the remainders modulo the factors in (*) are all conjugate. Therefore, only one remainder needs to be considered (see fig. 3). This remainder has $\lfloor \sqrt{N} \rfloor^*$ coefficients and can be acquired without any real calculations. To multiply in this modulo, a double length cyclic convolution is performed to do a full multiplication. The remainder is calculated thereafter. The cyclic convolution is a polynomial multiplication modulo $X^{2\lfloor \sqrt{N} \rfloor^*} - 1$. For $p = 2$ the algorithm is straightforward. The $(2\lfloor \sqrt{N} \rfloor^*)$-th root of unity exists in the field extension. So a symbolic FFT-type convolution algorithm can be used to do the cyclic convolution. To multiply the transformed coefficients, the method is used recursively. If $p > 2$, that simple scheme is not applicable. However, $(2\lfloor \sqrt{N} \rfloor^*) = 1$ and so the AGARWAL-COOLEY algorithm can be used to split the problem into a cyclic convolution of length 2 and a cyclic convolution of length $\lfloor \sqrt{N} \rfloor^*$. Since the necessary symbolic roots of unity exist, both of these can be performed by a symbolic FFT-type convolution algorithm. Again the method is used recursively to multiply the transformed coefficients.
For signal lengths \( N = 2^n \) the factorization of \( P_n(X) = X^M + 1 \) in (1) is much simpler
\[
X^M + 1 = \prod_{i=1}^{\lceil \sqrt{M} \rceil} (X^{\sqrt{M}} - \omega^{\sqrt{M}})^{-1}
\]
and is illustrated in fig. 4.

What is left, is the complexity of multiplying \( 2 \sqrt{N} \) coefficients of length \( \frac{N}{p} \sqrt{N} \). Since the same algorithm applies, the expense is again
\[
O(2 \sqrt{N} \sqrt{N} \log \sqrt{N}) = O(N \log N).
\]
So every recursion brings about this cost. It stops after about \( \log \log N \) recursion steps. Therefore, the total cost is:
\[
O(N \log N \cdot \log \log N).
\]

4. EXAMPLE

To illustrate the algorithm the following example is given.

**Example** This is a complete example for a convolution of length \( N = 8 \). It is calculated through a cyclic convolution of length 16. The randomly chosen signals to be convolved are:

\[
\begin{align*}
A &= (2, 5, 7, 2, 7, 9, 4, 1) \\
B &= (6, 3, 2, 1, 1, 8, 7, 3).
\end{align*}
\]

The cyclic convolution is a multiplication of polynomials modulo \( X^{16} - 1 \). The factors are the polynomials:
\[
\begin{align*}
A &= 2X^7 + 5X^6 + 7X^5 + 2X^4 + 7X^3 + 9X^2 + 4X + 1, \\
B &= 6X^7 + 3X^6 + 2X^5 + X^3 + 8X^2 + 7X + 3.
\end{align*}
\]

The higher order coefficients are all set to zero. According to the Chinese remainder theorem, the factorization of the polynomial \( X^{16} - 1 = (X^8 - 1)(X^8 + 1) \) asks for calculating the remainders modulo \( (X^8 - 1) \) and \( (X^8 + 1) \). Since the higher order coefficients are all equal to zero, the remainders are again \( A \) and \( B \). The result of \( A \times B \mod (X^8 - 1) \) is:
\[
C \mod (X^8 - 1) = A \times B \mod (X^8 - 1) = 142X^7 + 119X^6 + 142X^5 + 193X^4 + 162X^3 + 132X^2 + 126X + 131.
\]

This was calculated by the same algorithm. A word on notation: a column vector will denote an element of the field extension from now on. The meaning of
\[
\begin{pmatrix}
k_0 \\
k_1 \\
k_2 \\
k_3
\end{pmatrix}
\]
is \( k_3 \psi^3 + k_2 \psi^2 + k_1 \psi + k_0 \), where \( \psi = \sqrt{i} \) is a primitive 8th root of unity.

Now the multiplication of \( A \) and \( B \mod (X^8 + 1) \) has to be calculated using the new algorithm. Since \( N/2 = M = 8 \), \( \sqrt{8} \) has order 2 and \( \sqrt{8}^* = 4 \), a 4-dimensional extension is chosen. Therefore, the factors are reduced modulo \( (X^2 - \sqrt{i}) \):
\[
A \mod (X^2 - \sqrt{i}) = \begin{pmatrix} 4 \\ 7 \\ 2 \end{pmatrix} X + \begin{pmatrix} 1 \\ 9 \\ 2 \end{pmatrix},
\]

3.2. Investigation of Time Expense

The cyclic convolution is a multiplication of polynomials modulo \( X^N - 1 \). That problem is reduced to multiplying once modulo \( X^{N/p} - 1 \) and once modulo \( P_n(X) \) by the Chinese remainder theorem. Just scanning the input takes \( O(N) \) operations to do, so the algorithm will at least have that cost. Therefore, reducing the problemsize to \( N/p \) results in a runtime which is at most half of that of the original length. Since the above reduction and its inverse can be performed in \( O(N) \) steps, the investigations can be limited to the expense of multiplying modulo \( P_n(X) \). Twice the operations for that problem plus the \( O(N/p) \) from the reduction are an upper bound for the total cost.

The polynomials modulo \( P_n(X) \) have the length \( \frac{N}{p} \). The reduction modulo \( X^{\sqrt{N}} - \omega^{\sqrt{N}} \) takes \( O(N/p) \) steps, since it is just a copying of the input (v. example 1). The FFT of prime power length is known to have a cost of \( O(n \log n) \) for input of length \( n \). Therefore, the double length FFT’s needs \( O(N \log N) \) operations in the field extension. Since the coefficients are of length \( \frac{N}{p} \sqrt{N} \), every one of those operations requires \( O(\sqrt{N}) \) steps (only additions, subtractions and shifts occur). So the total expense so far is
\[
O(\sqrt{N} \sqrt{N} \log N) = O(N \log N).
\]
\[ B \text{ mod } (X^2 - \sqrt{1}) = \begin{pmatrix} 7 \\ 1 \\ 2 \\ 6 \end{pmatrix} X + \begin{pmatrix} 3 \\ 8 \\ 1 \\ 3 \end{pmatrix}. \]

Note that these remainders have been achieved by mere copying. Now a cyclic convolution of length 4 is performed (twice the length of the polynomials, so that no overflow occurs). That is a multiplication modulo \((X^4 - 1)\). The top coefficients are set to zero and the symbolic FFT calculates the 4 spectral coefficients:

\[
\begin{align*}
a_3 &= \begin{pmatrix} 8 \\ 11 \\ -2 \\ -2 \end{pmatrix}, & a_2 &= \begin{pmatrix} -3 \\ 2 \\ -5 \\ 3 \end{pmatrix}, \\
ab_1 &= \begin{pmatrix} -6 \\ 7 \\ 6 \\ 12 \end{pmatrix}, & b_0 &= \begin{pmatrix} 5 \\ 16 \\ 9 \\ 7 \end{pmatrix}, \\
b_3 &= \begin{pmatrix} 5 \\ 14 \\ -6 \\ 2 \end{pmatrix}, & b_2 &= \begin{pmatrix} -4 \\ 7 \\ -1 \\ -3 \end{pmatrix}, \\
b_1 &= \begin{pmatrix} 1 \\ 2 \\ 8 \\ 4 \end{pmatrix}, & b_0 &= \begin{pmatrix} 10 \\ 9 \\ 3 \\ 9 \end{pmatrix}.
\end{align*}
\]

During the symbolic FFT, multiplications by roots of unity are negative shifts. Those spectral coefficients are multiplied separately:

\[
\begin{align*}
c_3 &= \begin{pmatrix} 34 \\ 159 \\ 100 \\ -88 \end{pmatrix}, & c_2 &= \begin{pmatrix} -8 \\ -41 \\ 46 \\ -40 \end{pmatrix}, \\
c_1 &= \begin{pmatrix} -106 \\ -125 \\ -76 \\ 56 \end{pmatrix}, & c_0 &= \begin{pmatrix} -184 \\ 103 \\ 186 \\ 244 \end{pmatrix}.
\end{align*}
\]

The inverse FFT yields the result of the cyclic convolution:

\[
\begin{align*}
C_3 &= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, & C_2 &= \begin{pmatrix} -30 \\ 7 \\ 52 \end{pmatrix}, \\
C_1 &= \begin{pmatrix} -88 \\ 72 \\ 70 \\ 142 \end{pmatrix}, & C_0 &= \begin{pmatrix} -66 \\ 24 \\ 64 \end{pmatrix}.
\end{align*}
\]

That result is used to calculate the remainder:

\[
C \text{ mod } (X^2 - \sqrt{1}) = \begin{pmatrix} -88 \\ 72 \\ 142 \end{pmatrix} X + \begin{pmatrix} -125 \\ -6 \\ 71 \end{pmatrix}.
\]

Another reindexing of the data produces the result:

\[
C \text{ mod } (X^8 + 1) = 142X^7 + 95X^6 + 70X^5 + 71X^4 + 72X^3 - 6X^2 - 88X - 125.
\]

This result is combined with the previous result of

\[
C \text{ mod } (X^8 - 1)
\]

to get

\[
C \text{ mod } (X^{16} - 1)
\]

by interpolation. Since the factors \(A\) and \(B\) had length 8 and therefore this cyclic convolution of length 16 resulted in no overflow, the final result is:

\[
C = (12, 36, 61, 45, 69, 107, 128, 142, 107, 106, 132, 117, 63, 19, 3).
\]

5. RESULTS

The new convolution algorithms were implemented on SUN workstations. All programs were written in machine code. All input coefficients were coded as 32 bit integers. The resulting coefficients were 64 bit integers. A comparison shows that our new algorithm clearly outperforms all known direct convolution and FFT convolution methods for convolution lengths \(N > 256\).

6. REFERENCES


