동형 암호 시스템을 위한 정수 푸리에 변환의 메모리 절약 구조

문상국
목원대학교

Memory saving architecture of number theoretic transform for lattice cryptography

Sangook Moon
E-mail : smoon@mokwon.ac.kr

요 약

동형 암호화 시스템을 구현하는 데 있어, encrypt, decrypt, recrypt 연산은 큰 골격을 이루는 연산이다. 각각에 있어 공통된 가장 중요한 연산은 백만 비트가 넘는 큰 정수에 대한 법 곱셈이며, 이것은 푸리에 변환을 반복적으로 수행하여 얻을 수 있는 매우 큰 정수에 대한 곱셈 연산과 곱셈 결과에 대한 법 간소화를 요구한다. 본 논문에서는 Schonhage-Strassen이 제안한 큰 정수에 대한 법 곱셈을 수행하는 알고리즘을 응용하여, 이를 다시 메모리를 절약할 수 있는 효율적인 알고리즘을 제안하고 구현한다. 제안한 정수 푸리에 변환 구조는 FPGA에 구현하여 성능을 비교하였다.

ABSTRACT

In realizing a homomorphic encryption system, the operations of encrypt, decypt, and recrypt constitute major portions. The most important common operation for each back-bone operations include a polynomial modulo multiplication for over million-bit integers, which can be obtained by performing integer Fourier transform, also known as number theoretic transform. In this paper, we adopt and modify an algorithm for calculating big integer multiplications introduced by Schonhage-Strassen to propose an efficient algorithm which can save memory. The proposed architecture of number theoretic transform has been implemented on an FPGA and evaluated.

키워드

homomorphic encryption, cryptography, number theoretic transform

Ⅰ. 서 론

매년 MIT는 향후 큰 영향력을 끼칠 것으로 예상하는 10대 최신 기술을 선정하여 Technology Review라는 웹사이트에 발표하며, 2011년에는 cancer genomics, cloud streaming, crash-proof code, gestural interfaces, homomorphic encryption, social indexing, smart transformers, solid-state batteries, separating chromosomes, synthetic cells의 10가지 기술이 선정되었다 [1]. 이 중 본 논문에서 제안하고자 하는 기술은 동형 암호화 (homomorphic encryption; HE)에 관한 것으로, 암호화된 데이터에 대해 복호화하지 않고 연산을 고속으로 처리할 수 있는 기술이다. HE를 구현하는데 필요한 핵심적인 이슈는 매우 큰 암호 키에 대한 (백만 비트 이상) modular multiplication과 이에 수반하는 정수 FFT (fast fourier transform) 연산이며, 또한 이 큰 연산이 격자 기반의 초대형 벡터를 사용한다는 점이다. 본 논문에서는 Strassen 방식의 알고리즘을 개선하여 메모리를 효율적으로 절약할 수 있는 알고리즘을 제안하고 FPGA에 구현하여 성능을 평가한다.

II. 본 론

1. Strassen NTT 알고리즘

입력 : $(n-1)$ 차수의 polynomial $a(x) \in Z_q[x]$와 primitive n-th root of unity $\omega \in Z_p$, 소수 q
출력 : $(n-1)$ 차수의 polynomial $A(x) \in Z_q[x]$ = $\mathrm{NTT}_\omega(a)$

1. A <- bit_reverse($a(x)$)
2. m <- 2
3. **while** $m \le N$ **do**
4. s <- 0
5. **while** $s < N$ **do**
6. **for** i to $m/2-1$ **do**
7. N <- $i \cdot n/m$
8. a <- $s + i$
9. b <- $s + i + m/2$
10. c <- $A[a]$
11. d <- $A[b]$
12. $A[a]$ <- $c + \omega^{N \bmod n} d \bmod q$
13. $A[b]$ <- $c - \omega^{N \bmod n} d \bmod q$
14. s <- $s + m$
15. m <- $m \cdot 2$
16. **return** A

NTT는 순수 조합회로로 구성이 가능하다. 하지만 Strassen 방식으로 구현하게 되면 파라미터들을 담을 메모리가 많이 필요하게 되어, 본 논문에서 의도하는 FPGA에 구현하기가 어렵다.

2. NTT 알고리즘의 개선

입력 : Polynomial $a(x) \in Z_q[x]$ of degree $n-1$ and $n-th$ primitive root $w_n \in Z_q$ of unity
출력 : Polynomial $A(x) \in Z_q[x] = \mathrm{NTT}(a)$

begin
$A \leftarrow BitReverse(a)$;
for $m = 2$ to $n/2$ by $m = 2m$ **do**
$\omega_m \leftarrow m-th$ primitiveroot(1);
$\omega \leftarrow squareroot(\omega_m)$ or 1
for $j = 0$ to $m/2-1$ **do**
for $k = 0$ to $n/2-1$ by m **do**
$(t_1, u_1) \leftarrow (A[k+j+m/2], A[k+j])$
$(t_2, u_2) \leftarrow (A[k+m+j+m/2], A[k+m+j])$
$t_1 \leftarrow \omega \cdot t_1$;
$t_2 \leftarrow \omega \cdot t_2$;
$(A[k+j+m/2], A[k+j]) \leftarrow (u_1 - t_1, u_1 + t_1)$;
$(A[k+m+j+m/2], A[k+m+j]) \leftarrow (u_2 - t_2, u_2 + t_2)$;
$MEMORY[k+j] \leftarrow (A[k+j+m], A[k+j])$;
$MEMORY[k+j+m/2] \leftarrow (A[k+j+3m/2], A[k+j+m/2])$;
end
$\omega \leftarrow \omega \cdot \omega_n$;
end
end
$m \leftarrow n$;
$k \leftarrow 0$;
$\omega \leftarrow squareroot(\omega_m)$ or 1 /* Depending on forward or backward NTT */
for $j = 0$ to $m/2-1$ **do**
$(t_1, u_1) \leftarrow (A[j+m/2], A[j])$
$t_1 \leftarrow \omega \cdot t_1$;
$(A[j+m/2], A[j]) \leftarrow (u_1 - t_1, u_1 + t_1)$;
$MEMORY[j] \leftarrow (A[j+m/2], A[j])$;
$\omega \leftarrow \omega \cdot \omega_m$;
end
end

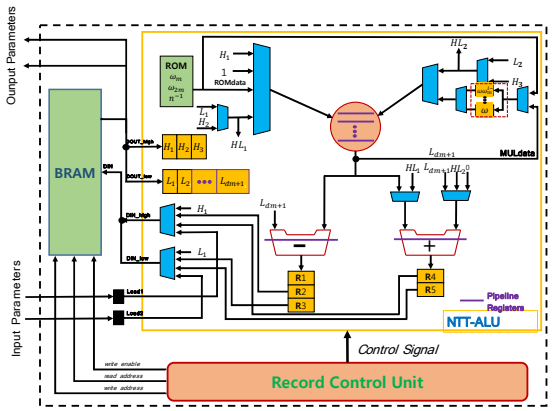

그림 1. 메모리 소비를 줄인 NTT의 반복 구조

그림 1은 Strassen 알고리즘을 개선하여 본 논문에서 제안한 알고리즘을 FPGA에 구현한 것이다. 기존 알고리즘에 비해 BRAM 소모량을 약 30% 줄일 수 있다.

III. 결 론

동형 암호화 시스템을 구현하는 데 있어 가장 중요한 연산은 백만 비트가 넘는 큰 정수에 대한 법 곱셈이며, 이것은 푸리에 변환을 반복적으로 수행하여 얻을 수 있는 매우 큰 정수에 대한 곱셈 연산과 곱셈 결과에 대한 법 간소화인 정수 푸리에 변환이다. 본 논문에서는 Strassen 방식의 정수 푸리에 변환을 개선하여 메모리를 최소 용량으로 사용할 수 있도록 하였다. Ring-LWE를 FPGA 상에 구현하는데 적용이 가능하다.

감사의 글

이 논문은 2014년도 정부 (교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2014R1A1A2A16053925)

참고문헌

[1] S. S. Roy, et al., “Compact Ring-LWE Cryptoprocessor,” Lecture Notes in Computer Science, Vol. 8731, pp. 371-391, 2014.

[2] Tsz-Wo Sze, “Schonhage-Strassen Algorithm with MapReduce for Multiplying Terabit Integers,” Proceedings of the 2011 International Workshop on Symbolic-Numeric Computation, San Jose, California, June 7-9, 2011.