ARMv8에서의 FFT 구현 동향

엄시우*, 권혁동*, 심민주*, 서화정**

*한성대학교 (대학원생)
**한성대학교 (교수)

Trends in Implementation of Fast Fourier Transform on ARMv8

Si-Woo Eum*, Hyeok-Dong Kwon*, Min-Joo Sim*, Hwa-Jeong Seo**

*Hansung University(Graduate student)
**Hansung University(Professor)

요 약

다항식 곱셈은 인공지능 학습과 격자 기반 암호에서 많이 사용되고 있다. 다항식 곱셈에 DFT 알고리즘 중 하나인 FFT를 적용하여 계산복잡도를 줄임으로써 성능 향상이 가능하다. 본 논문에서는 고성능 프로세서 중 하나인 ARMv8상에서의 최근 FFT 연구 동향에 대해서 알아본다.

I. 서론

DFT(Discree Fourier Transform)는 디지털 신호 처리 분야에서 이산 변환을 위해 가장 많이 사용되는 알고리즘 중 하나이다. DFT 알고리즘 중 하나인 FFT는 다항식 곱셈에 활용하여 효율적인 곱셈이 가능하다. 다항식 곱셈은 인공지능 분야에서 모델 학습을 위해서 많이 사용하며, 다가오는 양자컴퓨터를 대비해 개발되고 있는 격자 기반 암호에서도 사용된다. 본 논문에서는 고성능 프로세서 중 하나인 ARMv8상에서의 FFT 최적화 구현 연구 동향에 대해서 알아본다.

II. 관련 연구

2.1 Fast Fourier Transform(FFT)

Fourier Transform은 시간에 따라서 변하는 신호(통신, 영상처리 등), 즉 연속된 데이터를 주파수 성분으로 분해하는 변환이다. 하지만 연속된 데이터가 아닌 이산형 데이터를 분석하기 위해 이산 Fourier Transform을 사용한다. 이산 Fourier Transform의 시간복잡도는 $O(n^2)$다. 이를 더 빠르게 연산될 수 있도록 개선한 알고리즘을 Fast Fourier Transform(FFT)라고 한다.

FFT의 대표적인 알고리즘은 Cooley-Tukey 알고리즘이 있다[1]. 수식 (1),(2)는 F(n)을 푸리에 변환을 적용한 값이라고 할 때, 원소들을 홀수와 짝수 항으로 나누어 계산하는 수식을 나타낸다.

$$F(n) = F_{even}(n) + w_N^n \cdot F_{odd}(n) \quad (1)$$

$$F(n+N/2) = F_{even}(n) + w_N^n \cdot F_{odd}(n) \quad (2)$$

분할 정복 알고리즘을 적용할 수 있기 때문에 시간복잡도가 $O(n \cdot \log n)$으로 줄일 수 있다. 수식을 이용하기 위해 재귀함수를 사용하며, 이를 버터플라이 알고리즘을 활용하여 구현한다. (그림 1)은 8개의 데이터를 처리하는 FFT 순서도이

다[2].

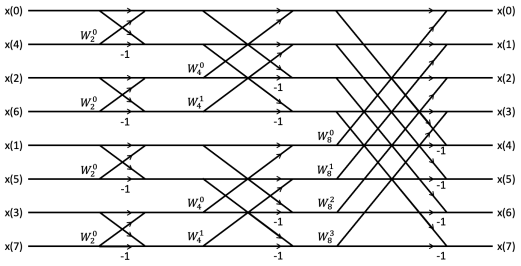

Fig. 1. Butterfly FFT Algorithm Flowchart[3]

2.2 ARMv8 Architecture

ARMv8-A는 고성능 임베디드 64-bit 아키텍처로, 32-bit(AArch32), 64-bit(AArch64)아키텍처 모두를 지원한다. 64-bit로 사용 가능한 x0-x30의 범용 레지스터 31개를 제공하고, 이 범용 레지스터는 w0-w30으로 32-bit로도 사용 가능하다. 벡터 레지스터는 128-bit 크기를 가지며, v0-v31로 32개를 제공하고 있다[4].

III. ARMv8에서의 FFT 구현

3.1 Multi-dimensional Real FFT on ARMv8 Platform[5]

2018년 Wang et al.은 ARMv8 플랫폼에서 1D, 2D Real DFT를 구현하고 최적화 하였다. Real DFT는 실제 응용 프로그램에서 Fourier Transform은 실수 입력 처리에 집중되는데, 해당 논문에서는 이러한 변환을 Real DFT(Real discrete fourier transform)라고 한다.

해당 연구에서는 첫 번째로 Real DFT의 대칭성을 활용하여 기존의 Real DFT의 계산복잡성을 감소시켰다. 두 번째로 버터플라이 네트워크를 재구성하고, 버터플라이 계산을 단순화시키고, SIMD 어셈블리 명령어를 사용하여 ARMv8 플랫폼에서 고성능 1D Cooley-Tukey FFT 알고리즘을 최적화 구현하였다. 마지막으로 2D Real DFT의 구현에서는 캐시 성능 향상을 위해서 ARMv8 플랫폼용 캐시 인식 알고리즘을 제안하여 성능 향상을 하였다.

결론적으로 FFTw 3.3.7과 비교하여 1D-float DFT는 모든 Real DFT에서 평균적으로 1.52배 속도가 향상되었다. 또한 1D-Double DFT에서는 평균 1,34배 속도 향상을 얻었다. 2D-Float DFT에서는 평균 1.41배 속도 향상을 얻었고, 2D-Double에서는 평균 1.10배 속도가 향상되었다.

3.2 Accelerating Falcon on ARMv8[6]

2022년 Kim et al.d은 디지털 서명 알고리즘은 Falcon의 FFT 기반 다항식 곱셈 최적화를 진행하였다. Falcon은 NIST에서 진행하고 있는 PQC(Post-Quantum Cryptography) 표준화에서 최종 선택된 디지털 서명 알고리즘이다. Falcon의 주요 계산은 복소수 영역과 정수 영역의 다항식 곱셈이다. 일반적으로 FFT 기반의 곱셈방식은 복소수 영역의 효율적인 다항식 곱셈을 위해 사용되어 왔다.

복소수 연산은 Falcon의 서명 생성 및 키 쌍 생성에서 사용된다. Falcon에서는 복소수 연산을 효율적으로 처리하기 위해서 낮은 수준의 연산인 FPC-MUL, FPC-DIV, FPC,ADD, FPC-SUB 연산을 제공하고 있다. 이러한 연산에서 많은 FFT 곱셈을 형성하고 있다. 여러 함수에서 FFT 기반 곱셈 성능을 향상시키기 위해서 FP 명령어와 NEON 엔진을 활용하여 병렬 구현 기법을 제안하고 있다. 또한 NEON 엔진에서 사용 가능한 레지스터를 최대한 활용하여 다항식 곱셈 중 중복 메모리 액세스 수를 최소화하였다.

결과적으로 FFT와 NTT 기반 다항식 곱셈 병렬 구현을 통해 키 생성에서 15.1%, 16.5%, 65.4%의 성능 향상이 되었다.

3.3 Research on the realization and optimization of FFTs[7]

2020년 Qi Du와 Hui Huang은 ARMv8 플랫폼에서 FFT 소프트웨어 패키지를 구현하고 최적화를 진행했다. 오픈 소스 FFT 라이브러리에서 FFTW와 FFTs 패키지는 주로 ARMv7 플랫폼에 적합하고, ARMv8에서는 FFTW만 적합하였다.

FFTW는 계산을 위해 길이가 N인 시퀀스를 길이가 N1 및 N2인 짧은 시퀀스로 분해한다. 이러한 짧은 시퀀스는 변환을 해결하는 방법인

Solver라고 한다. 동일한 길이 N에 대해 분해하는 방법이 여러 가지가 있다. FFTW는 다양한 방법에 대해 평가하여 실행 속도가 빠른 계획은 선택한다.

FFTs는 가장 빠른 푸리에 변환으로도 알려졌으며, 2012년 뉴질랜드 와이카토 대학의 Blake가 Github의 오픈 소스 프로젝트로 개발되었다. 하지만 FFTs는 x86, x64, ARM 및 기타 플랫폼을 지원하지만 Aarch64는 지원하고 있지 않다. 해당 논문에서는 FFTs에서 ARMv7으로 구현된 파일에 대해 ARMv8에서 동작할 수 있도록 ARMv7 명령어 및 레지스터를 ARMv8 명령어 및 레지스터로 교체하여 구현하였다.

결과적으로 ARMv8에서 FFTs의 최적화 구현을 달성하였고, FFTW 대비 18% 성능 향상을 보여주고 있다.

IV. 결론

본 논문에서는 ARMv8상에서의 Fast Fourier Transform 구현 연구에 대해서 알아보았다. FFT는 인공지능 학습과 격자 기반 암호에서 사용되는 다항식 곱셈 계산 속도를 향상시키기 위한 중요한 알고리즘이다. FFT의 최적화 연구는 점점 더 많은 계산을 요구하고 있는 인공지능 학습과 양자컴퓨터를 대비해 개발된 격자 기반 암호의 다항식 곱셈 증가에 맞춰 앞으로도 꾸준한 연구를 통한 효율적인 구현 연구가 필요하다.

V.Acknowledgement

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00264, Research on Blockchain Security Technology for IoT Services, 100%).

[참고문헌]

[1] Cooley J W, Tukey J W. An algorithm for the machine calculation of complex Fourier series[J]. Mathematics of Computation, 1965, 19(90): 297-301.

[2] 전다윤, 윤상혁, 박능수.(2020).GPGPU 메모리 할당을 고려한 2D FFT 병렬화.한국정보과학회 학술발표논문집,(),1127-1129.

[3] Reshma P.G.a, Varun P. Gopia,V. Suresh Babub, Khan A. Wahidc "Analog CMOS implementation of FFT using cascode current mirror",pp 31-32 ,2017

[4] Armv8-A Instruction Set Architecture. [online] available: https://documentation-service.arm.com/static/613a2c38674a052ae36ca307?token=.

[5] Wang, Xiao, et al. "Implementation and optimization of multi-dimensional real FFT on ARMv8 platform." International Conference on Algorithms and Architectures for Parallel Processing. Springer, Cham, pp. 338-353. 2018.

[6] Kim, Youngbeom, Jingyo Song, and Seog Chung Seo. "Accelerating Falcon on ARMv8." IEEE Access 10 : 44446-44460. 2022.

[7] Du, Qi, and Hui Huang. "Research on the realization and optimization of FFTs in ARMv8 platform." IOP Conference Series: Materials Science and Engineering. Vol. 768. No. 7, p. 072114. IOP Publishing, 2020.