Title: AutoFFT: A Template-Based FFT Codes Auto-Generation Framework for ARM and X86 CPUs

JX-Ma

2024/11/2

ABSTRACT

This article proposes a template based code generation framework called AutoFFT, which can automatically generate high-performance Fast Fourier Transform (FFT) code. AutoFFT adopts the Cooley Tukey FFT algorithm and utilizes the symmetry and periodicity of DFT matrix as an external parallelization framework.

INTRODUCE

The Cooley Tukey algorithm is the most widely used Fast Fourier Transform (FFT) algorithm in many practical applications. It uses divide and conquer method to recursively decompose large DFT into smaller DFT, achieving a complexity of O (n log n).

The AutoFFT framework is mainly divided into two stages: the installation stage generates an efficient FFT kernel, and the runtime stage constructs an efficient butterfly network for a given FFT size through empirical search to find the best solution.
During the installation phase, three components were mainly used for optimization: the calculation template designer, the C FFT kernel generator, and the assembly optimizer.
The running process consists of three components, namely a factorization engine, a butterfly network generator, and a performance evaluator.

The main FFT optimizations are divided into prime cardinality and The Quad Pattern for Power of two Radices, The optimization of prime cardinality utilizes the horizontal and vertical characteristics of the conversion formula for optimization.The Quad Pattern for Power of Two Radices utilizes horizontal symmetry, vertical symmetry, and the Periodic Property.

The required number of real number multiplications for DFT transformation is

$4r^2$, The number of real number additions is $4r^2 - 2r$, The required number of real number multiplications for pair pattern is $r^2 - 2r + 1$, the additions is $2r^2 - 3r + 3$. The required number of real number multiplications for quad pattern is $\sum_{k=1}^{\frac{r}{4}-1} \frac{(r+4)}{p}$, the additions is $r^2 - 5r + 16$.

The following formula was adopted in the famous benchmark test benchFFT:

$$Gflops = \frac{5N * log2N * 10^{-9}}{t}$$

Summary

This paper proposes a template-based framework named AutoFFT that makes full use of the experience of domain and optimization experts to automatically generate extremely high-performance as- sembly FFT codes for ARMv8 and Intel Haswell processors. AutoFFT thus substantially reduces the laborious work of developing assem- bly codes manually. The experiments show that AutoFFT performs generally better than FFTW, ARMPL, and Intel MKL.