
332:573 – Data Structures & Algorithms
Project Report
Improvements to Polynomial Multiplication Through Fast Fourier
Transforms

By David Lambropoulos, Demetrios Lambropoulos

Professor Shantenu Jha
May 6th, 2018



Rutgers University
School of Engineering

Contents

1	Introduction/Motivation	1
2	Algorithms/Theory	2
3	Experimental Setup	2
4	Results and Analysis	2
5	Discussion	3

1 Introduction/Motivation

Polynomials are expressions built up of a combination of constants c and symbols called variables. Polynomials with a single indeterminate x can always be written in the form

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0$$

where a_0, \dots, a_n are constants and x is the indeterminate variable. Polynomials can also be expressed in the following canonical form

$$\sum_{k=0}^n a_k x^k$$

Each term in a polynomial is product of some constant called the coefficient of the term and indeterminate raised to a nonnegative integer power n (i.e. $n \in \mathbb{Z}^+$). Polynomials are used in many fields to represent problems or model behavior such as Chemistry, Physics, Economics, Social Scientists, Calculus, Numerical Analysis, etc.

The way polynomials are multiplied is through a method called First, Outer, Inner, Last (FOIL) as can be seen below in Figure 1. The issue with the FOIL method is that it requires too many operations to perform multiplications, especially for large polynomials. Usually real world problems will be modeled with polynomials of 10000+ terms.

FIND THE PRODUCT:

$$\begin{array}{l}
 8x + 4 \quad 2x + 5 \\
 \text{FIRST} \quad (8x + 4)(2x + 5) \\
 \text{OUTER} \quad = 16x^2 + 40x + 8x + 20 \\
 \text{INNER} \quad = 16x^2 + 48x + 20 \\
 \text{LAST} \quad
 \end{array}$$

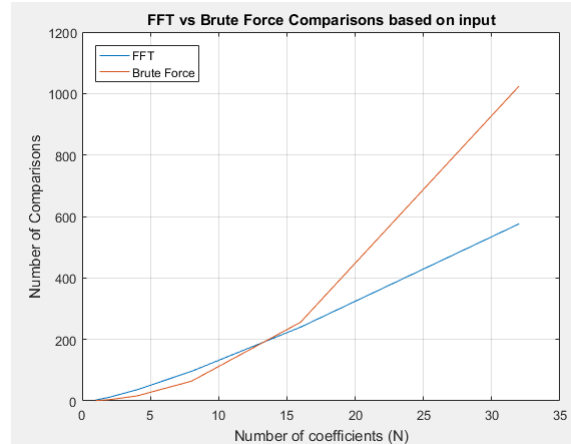
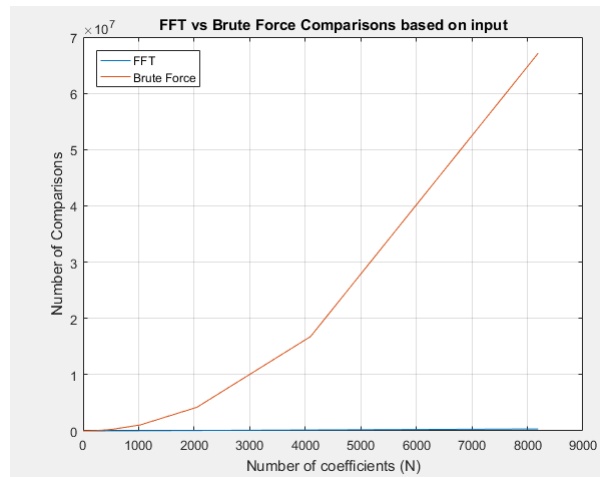
Figure 1: Demonstration of FOIL method

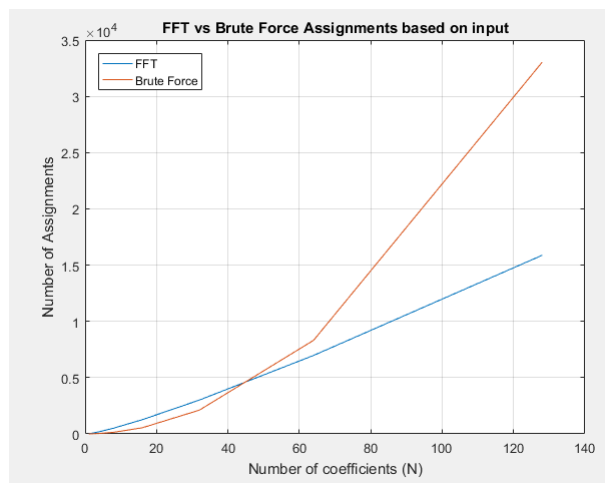
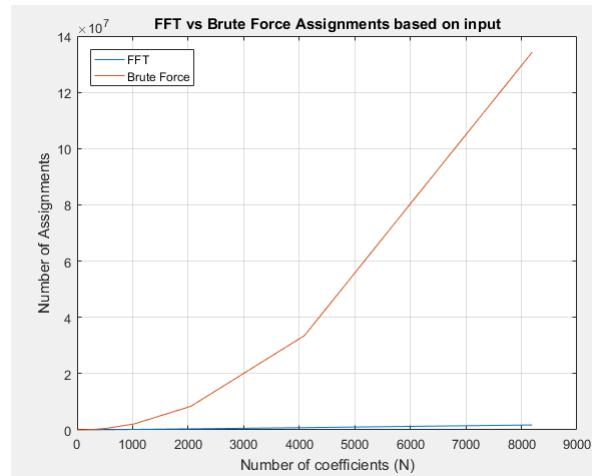
The main scientific motivation behind improving the speed in which we multiple polynomials is that solving large polynomials by hand is tedious and error prone. As discussed previous solving large polynomial multiplications through quadratic algorithms (i.e. FOIL) would cost more time than any person might have. A way that can fix this costly calculation is to implement recursion which uses the divide and conquer method.

2 Algorithms/Theory

3 Experimental Setup

4 Results and Analysis





5 Discussion

[1]

References

[1] “test,” 2017. test.