Integer multiplication in time $O(n \log n)$
DD2467 Individual Project in TCS

Marcus Östling

June 1, 2022

1 Introduction

The purpose of this project is to a implement the algorithm presented in the paper "Integer multiplication in time $O(n \log n)$" by David Harvey and Joris van der Hoeven, and then compare it to a simple integer multiplication using a FFT in one dimension.

1.1 Previous work

1.2 Purpose

According to Van Der Hoeven the threashold for when their new algorithm is suppose to out perform Schönhage-Strassen is when multiplying integers of $2^{1729^12} \approx 10^{21485709110445525194063504505941734195 2}$ digits. The number 1729 comes from the dimension (fact check) and I will use an lower dimension to see how this algorithm perform on integers that fit inside of a typical RAM (8 - 16 GB).

2 Method

2.1 Parameters

Large integers:
$8GB = 2^{33}bytes = 8589934592bytes$
input: $2 * 2^{29} = 2^{30}$
result: $2 * 2^{29} = 2^{30}$
s product = 2^{30} (d prime values)
t product = 2^{30} (d powers of 2)
d = 4 (dimension)
b = 31 (chunk size)

2.2 Simple FFT multiplication

Implement a simple FFT integer multiplication, similiar to Schönhage-Strassen,. Given two integers, split them up into coefficients for a polynomial, use DFT, apply pointwise multiplication, inverse DFT back into one integer.

Memory analysis: Input string of length n (n bytes)

Each coefficient uses 4 chars ($\approx$ 16 bits, 4 bytes) of the in string and will be converted to long double (8 bytes) The polynomial will have complex long double coefficients (16 bytes)

2.3 Harvey and Van Der Hoeven algorithm

This algorithm is inspired by Schönhage-Strassen: Given two integers, split them up into coefficients for a polynomial, use the algorithm given by the paper, convert back into one integer.

3 Results

3.1 Run algorithms and compare (week 3)

Run both algorithms on the same inputs, up to large integer (size will be decided during week 51), and measure their run-times.

3.2 Conslusion