Brian Bowles

April 5, 2016

CS 5050

Polynomial Multiplication 2

After timing each algorithm again the data was plotted and the equation for each fitted line calculated. From those calculated equations the points at which the “Fast Fourier Transform” crosses the other two algorithms was computed by setting the equation for the “Fast Fourier Transform” equal to the other equations and solving for n, the length of the polynomial when they cross. The computed value for the crossing point of the “Fast Fourier Transform” and “High School” algorithms and the “Fast Fourier Transform” and “Divide & Conquer” algorithms was 1,019.094481 and 196.9295 respectively; those two values can be confirmed by either looking at the graphs below or the raw data for the timing studies.

Unlike the other two algorithms, “Fast Fourier Transform” has a graph that is almost linear because of its complexity. “Divide & Conquer” and “High School” are both O(n^b), which means that there graphs grows by the power of b, 1.585 for “Divide & Conquer” and 2 for “High School”. “Fast Fourier Transform” on the other hand has a complexity of O(n\*log(n)) which means the graph grows nearly linearly because n alone is a linear function and the log(n) term has little effect on the growth rate.

For this last technical analysis the limit on the runtime was set at 1200 seconds and the timing study was ran. The ceiling for the length of the polynomials were 262,144 for the “High School” algorithm, 524,288 for the “Divide & Conquer” algorithm, and a computed guess of 16,777, 216 for the “Fast Fourier Transform” algorithm; the reason the “Fast Fourier Transform” algorithm has a computed guess is because the program ran out of memory, 2GB of RAM, after the length of 4,194,304 was computed. It makes sense that the “Divide & Conquer” and “High School” algorithms would be the same because their complexities are nearly the same, while the “Fast Fourier Transform” algorithm has a much higher cap because its complexity is much better.