HW: FFT

Copyright Oliver Serang, 2018
University of Montana Department of Computer Science

1. What is the runtime to evaluate a polynomial with \( n \) coefficients at \( n \) distinct points using a naïve approach?

\[ \Theta(n^2) \]

2. What is the runtime to evaluate a polynomial with \( n \) coefficients at \( n \) distinct points using FFT?

\[ \Theta(n \log(n)) \]

3. What is an invariant? On what invariant does FFT rely?

- A property that doesn't change when a transformation is applied. In FFT, one such invariant is that the first half of the values are negatives of the second half.

4. Where do the complex numbers come from?

- Complex roots of unity - used to ensure at each recursion, the first half of the values are the negatives of the second half.

5. At what stage does FFT get some “free” computation?

- At each recursion, we only need to evaluate the first half of the points.

6. What are the steps for multiplying two polynomials of length \( n \) using FFT?

- Zero-pad both polynomials to size \( 2n \)
- Compute \( 2n \) points for which each poly passes through
- Multiply the two element-wise to get points through which the result passes
- Use inverse FFT to find coefficients of the result
7. What is the cost to multiply two polynomials of length $n$ with FFT?

$$\Theta(n \log(n))$$

8. Are there any FFT algorithms that can run an FFT of length $n$ on a standard desktop computer $\in o(n \log(n))$?

Not currently

9. Is it possible to construct an FFT algorithm that runs $\in o(n \log(n))$ on a standard desktop computer?

Unknown - it's an open problem

10. What is the cost to perform an FFT of length $n$ using a custom circuit?

$$\Theta(\log(n))$$