Divide and Conquer 4: FFT

Polynomial Multiplication

1. Multiplying polynomials using Fast Fourier Transform
   - Polynomials A(x) and B(x)
     - $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}$
     - $B(x) = b_0 + b_1x + b_2x^2 + \ldots + b_{n-1}x^{n-1}$
   - Want C(x) = A(x)B(x)
     - $C(x) = c_0 + c_1x + c_2x^2 + \ldots + c_{2n-2}x^{2n-2}$
     - $c_k = a_0b_k + a_1b_{k-1} + \ldots + a_kb_0$
   - Given $a = (a_0, a_1, \ldots, a_{n-1})$ and $b = (b_0, b_1, \ldots, B_{n-1})$
     - Compute $c = a * b = (c_0, c_1, \ldots, c_{n-1})$
     - Convolution of a and b

Quiz: PM Example

1. $A(x) = 1 + 2x + 3x^2$
   - a = (1, 2, 3)
2. $B(x) = 2 - x + 4x^2$
   - b = (2, -1, 4)
3. c = a * b = (2,3,8,5,12)

PM: General Problem

1. Given $a = (a_0, a_1, \ldots, a_{n-1})$ and $b = (b_0, b_1, \ldots, B_{n-1})$
   - Compute $c = a * b = (c_0, c_1, \ldots, c_{n-1})$
   - Convolution of a and b
2. Naive solution: O(k) time for c_k => $O(n^2)$ total time
   - FFT: O(nlogn)

Convolution Applications

1. Filtering: data $y = (y_1, \ldots, y_n)$
   - Applications: Reducing noise, adding effects
   - Mean filtering: Replace value with moving mean of surrounding values
   - Gaussian filtering: Replace value with Gaussian weighting of surrounding values
   - Gaussian blur: 2-dim Gaussian filter

Polynomials Basics

1. Polynomial $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}$
   - Two natural representations for A(x):
     - Coefficients $a = (a_0, a_1, \ldots, a_{n-1})$
     - Values: $A(x_1), A(x_2), \ldots, A(x_n)$
   - Lemma: Polynomial of degree n-1 is uniquely determined by its values at any n distinct points
   - FFT converts from coefficients to values and values to coefficients
     - For particular $x_1, \ldots, x_n$

PM: Values

1. Key idea: Multiplying polynomials is easy in values representation
   - Given $A(x_1), \ldots, A(x_{2n})$ and $B(x_1), \ldots, B(x_{2n})$
   - for i = 1 -> 2n $C(x_i) = A(x_i)B(x_i)$
     - O(1) time for each

- $O(n)$ total time

FFT: Opposites

1. Given $a = (a_0, a_1, \ldots, a_{n-1})$ for polynomial $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}$
   - Want to compute $A(x_1)$, $A(x_2)$, $A(x_{2n})$ for 2n points $x_1,\ldots,x_{2n}$ that we choose
2. Key: Suppose $x_1, \ldots, x_n$ are opposite of $x_{n+1}, \ldots, x_{2n}$
   - So: $x_{n+1} = -x_1$, $x_{n+2} = -x_2$
     - Plus/minus property

FFT: Splitting A(x)

1. Plus/minus property: $x_1, \ldots, x_n$ are opposite of $x_{n+1}, \ldots, x_{2n}$
   - Look at $A(x_i)$ and $A(x_{n+i}) = A(-x_i)$
     - Even terms $a_{2k}x^{2k}$ - same
     - Odd terms $a_{2k+1}x^{2k+1}$ - opposite
   - Let $a_{even} = (a_0, a_2, a_4, \ldots, a_{n-2})$
   - Let $a_{odd} = (a_1, a_3, a_5, \ldots, a_{n-1})$

FFT: Even and Odd

1. Given $a = (a_0, a_1, \ldots, a_{n-1})$ for polynomial $A(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}$
   - Let $a_{even} = (a_0, a_2, a_4, \ldots, a_{n-2})$
   - Let $a_{odd} = (a_1, a_3, a_5, \ldots, a_{n-1})$
2. $A_{even}(y) = a_0 + a_2y + a_4y^2 + \ldots + a_{n-2}y^{(n-2)/2}$
3. $A_{odd}(y) = a_1 + a_3y + a_5y^2 + \ldots + a_{n-1}y^{(n-2)/2}$
4. A_{even} and A_{odd} are degree n/2 - 1
5. Note: $A(x) = A_{even}(x^2) + xA_{odd}(x^2)$

FFT: Recursion

1. $A(x) = A_{even}(x^2) + xA_{odd}(x^2)$
   - Plus/minus property: $x_1, \ldots, x_n$ are opposite of $x_{n+1}, \ldots, x_{2n}$
   - $A(x_i) = A_{even}({x_i}^2) + x_iA_{odd}({x_i}^2)$
   - $A(x_{n+i}) = A(-x_i) = A_{even}({x_i}^2) - x_iA_{odd}({x_i}^2)$
2. Given $A_{even}(y_1) \ldots, A_{even}(y_n)$ and $A_{odd}(y_1), \ldots, A_{odd}(y_n)$
   - for $y_1 = {x_1}^2, \ldots, y_n={x_n}^2$
   - Takes $O(n)$ time to get $A(x_1), \ldots, A(x_{2n})$

FFT: Summary

1. To get A(x) of degree <= n-1 at 2n points $x_1, \ldots, x_{2n}$
   - Define $A_{even}(y)$ and $A_{odd}(y)$ of degree <= n/2 - 1
   - Recursively evaluate at n points
     - $y_1 = {x_1}^2 = {x_{n+1}}^2$
     - $y_2 = {x_2}^2 = {x_{n+2}}^2$
     - $y_n = {x_n}^2 = {x_{2n}}^2$
   - In $O(n)$ time, get A(x) at $x_1, \ldots, x_{2n}$
2. $T(n) = 2T(n/2) + O(n)$
   - Evaluates to $O(n\log n)$

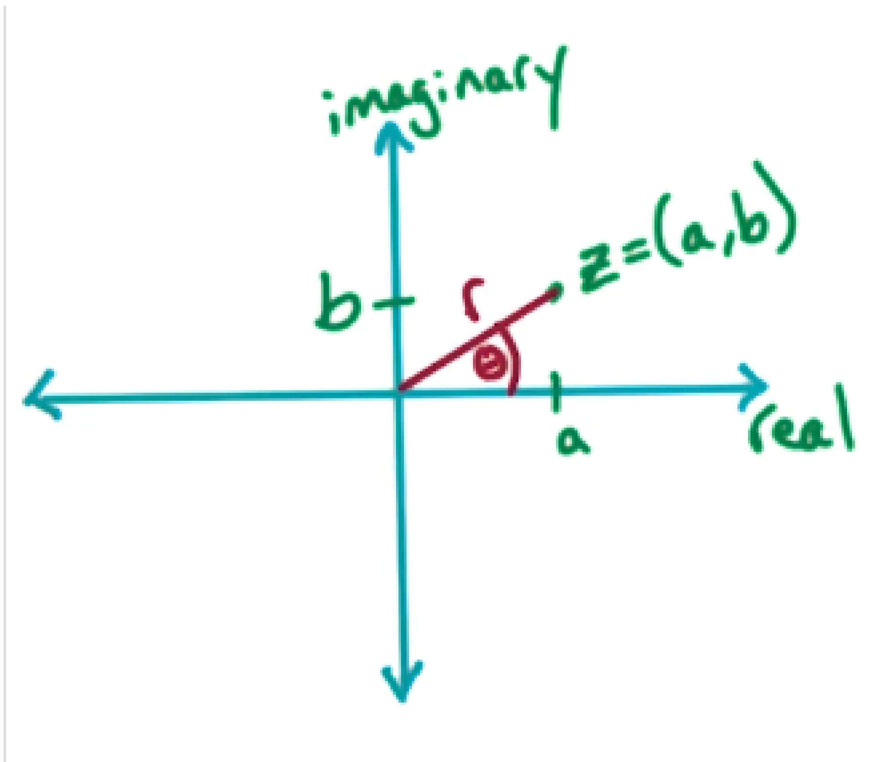
FFT: Recursive Problem

1. Choose: $x_{n+1} = -x_1$, $x_{n+2} = -x_2$, $x_{2n} = -x_n$
   - Next level: $y_1 = {x_1}^2, \ldots, y_n = {x_n}^2$

- y_1 = -$y_{n/2+1}$ <=> x_1^2 = -$x_{n/2+1}^2$
- For real numbers, the square of a number is always positive, so x_1^2 = -$x_{n/2+1}^2$ will never be true
  - Need to use complex numbers

Review: Complex Numbers

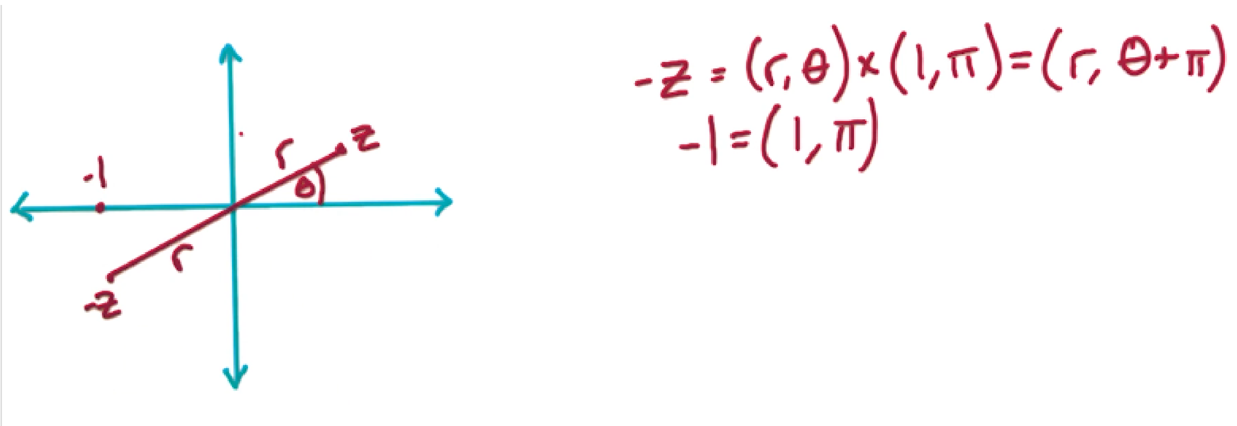
1. z = a + b_i
   - z = (a,b)
   - Alternatively, z = (r,theta) in polar coordinates
   - (a,b) = (rcos(theta),rsin(theta))
   - Euler's formula: r(cos(theta) + isin(theta)) = re^{itheta}



Complex Numbers

Multiplying in Polar

1. Polar is convenient for multiplication
   - z_1z_2 = (r_1r_2,$theta_1$ + $theta_2$)



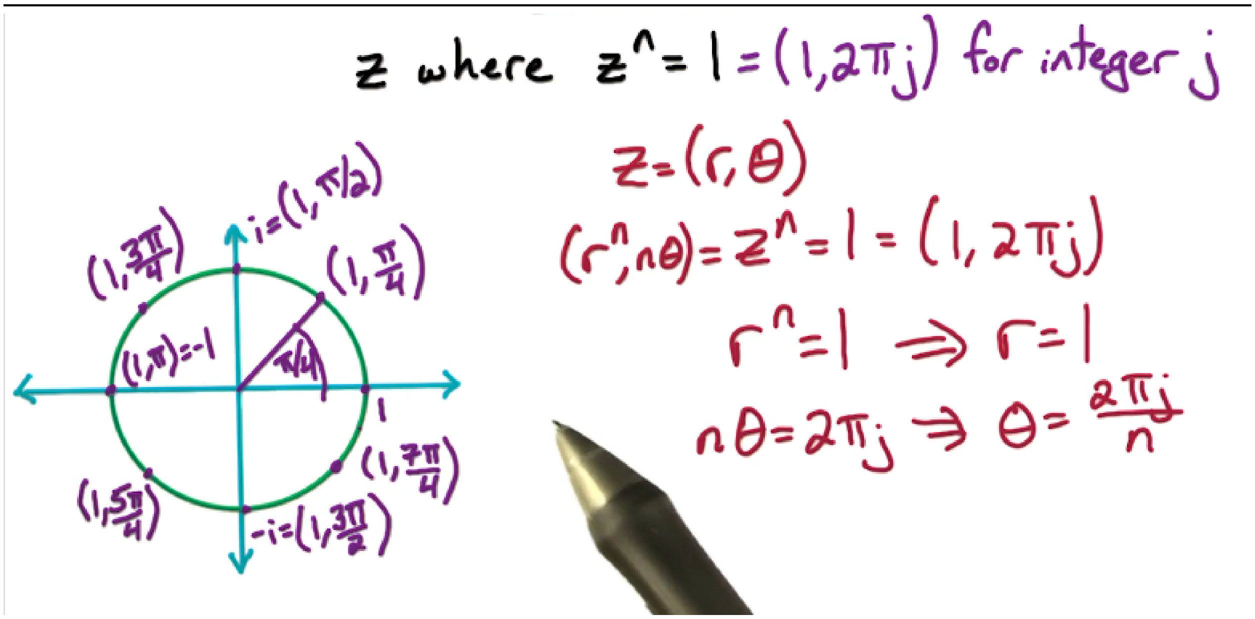
Complex Number Multiplication

Complex Roots

1. n^{th} complex roots of unity
    - What number raised to the nth power equals 1?
        - n = 2: 1, -1
        - n = 4: 1, -1, i, -i
    - z where $z^n = 1$

Roots: Graphical View

1. z where $z^n = 1 = (1,2pij)$ for integer j
    - z = (r,theta)



nth Roots

Roots: Notation

1. (1,2pin/j) for j = 0, 1, . . . , n-1

- theta = 2pi/n
  - Let w_n = (1, 2pi/n) = $e^{2ipi/n}$
2. n^{th} roots of unity:
   - $w_n{}^0$, $w_n{}^1$, . . . , $w_n{}^{n-1}$
   - $w_n{}^j$ = $e^{2ipij/n}$

Roots: Examples

1. n = 2: 1, -1
2. n = 4: 1, i, -1, -i
3. (n^{th} roots)^2 = n/2 roots
   - $w_{16}{}^2$ = w_8
   - Plus/minus property
     - $w_n{}^j$ = -$w_n{}^{j+n/2}$

Complex Roots Practice

1. Consider the n^{th} roots of unity for n = 16
   - What is w_{16} in polar coordinates?
   - (1, pi/8)
2. Consider w_{16}. For what power k is $(w_{16})^k$ = -1?
   - k = 8
   - -1 = (1,pi) and if z = (r,theta) then zk = (rk,kthetha)
3. Consider w_{16}. For what power k is $(w_{16})^k$ = -w_{16}?
   - k = 9
4. Consider w_{16}. For what power k is $(w_{16})^{-1}$ = -$w_{16}{}^k$?
   - k = 15
5. Consider w_{16}. For what power k is $(w_{16})^k$ = -$w_8{}^2$?
   - k = 4

Key Property: Opposites

1. Properties of n^{th} roots of unity:
   - For even n: Satisfy plus/minus property
     - First n/2 are opposite of last n/2
       - $w_n{}^0$ = -$w_n{}^{n/2}$
       - $w_n{}^1$ = -$w_n{}^{n/2+1}$
       - $w_n{}^{n/2-1}$ = -$w_n{}^{n-1}$

Key Property: Squares

1. Properties of n^{th} roots of unity:
   - For n = 2^k:
     - (n^{th} roots)^2 = n/2 roots

Properties of $n^{\underline{th}}$ roots of unity:

2. For $n=2^k$:

$$(n^{\underline{th}}\text{ roots})^2 = \frac{n}{2}\ \underline{\text{nd}}\text{ roots}$$

$$\left(\omega_n^j\right)^2 = \left(1, \frac{2\pi}{n} j\right)^2 = \left(1, \frac{2\pi}{n/2} j\right) = \omega_{n/2}^j$$

$$\left(\omega_n^{\frac{n}{2}+j}\right)^2 = \left(-\omega_n^j\right)^2 = \omega_{n/2}^j$$

nth Roots Squared