Polynomials and the FFT
Rings

A ring $R$ is a set together with addition ($+$) and multiplication ($\cdot$) operators:

- addition is associative and commutative
- $R$ contains a (unique) additive identity, denoted by $0$
- every element $a \in R$ has a (unique) additive inverse, denoted by $-a$
- multiplication is associative and commutative
- multiplication distributes over addition
- $R$ contains a (unique) multiplicative identity, denoted by $1$
Units:

- An element \( a \in R \) is a unit if it has a (unique) multiplicative inverse (denoted by \( a^{-1} \)).
- The set of units is denoted by \( R^* \), and is closed under multiplication (in fact, it’s a group).

Examples:

- \( \mathbb{Z} \) is a ring, \( \mathbb{Z}^* = \{ \pm 1 \} \) are the only units.
- \( \mathbb{Q} \) (the rationals) is a ring, all nonzero elements are units (such a ring is called a field).
- \( \mathbb{R} \) (the reals) and \( \mathbb{C} \) (the complex numbers) are also fields.
More examples:

- For positive integer $m$, let $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$, with arithmetic done modulo $m$
  - $\mathbb{Z}_m$ is a ring
  - $\mathbb{Z}_m^* = \{a \in \mathbb{Z}_m : \gcd(a, m) = 1\}$
  - $\mathbb{Z}_m$ is a field iff $m$ is prime
Polynomial rings

Let $R$ be a ring

The ring of polynomials, denoted by $R[X]$, consists of all *formal expressions* of the form

$$a_0 + a_1X + \cdots + a_dX^d,$$

where each $a_i \in R$

Let $f = \sum_i a_iX^i$, $g = \sum_i b_iX^i$

Usual rules for addition and multiplication:

- $i$th coefficient of $f + g$: $a_i + b_i$
- $i$th coefficient of $f \cdot g$: $\sum_{j+k=i} a_j b_k$
Polynomials define functions, but it’s best to just think of them as formal expressions

For computations, they are represented by their coefficient vectors: \( f = \sum_{i=0}^{d} a_i x^i \) is represented by 
\[
(a_0, a_1, \ldots, a_d)
\]

Typically, we assume that \( a_d \neq 0 \), in which case 
\[
\deg(f) := d, \quad \text{len}(f) := d + 1
\]

Also, 
\[
\deg(0) := -\infty, \quad \text{len}(0) := 1
\]
Polynomial arithmetic

For algorithms on polynomials, running time = number of operations in $R$

Naive algorithms:

- $f + g$: $O(\text{len}(f) + \text{len}(g))$ operations in $R$
- $f \cdot g$: $O(\text{len}(f)\text{len}(g))$ operations in $R$

Karatsuba for polynomials:

- uses $O(n^{\log_2 3})$ operations in $R$, where $n$ bounds the input lengths

We can do better: almost linear time!
Addition Algorithm:

**input:** coeff. vector \((a_0, \ldots, a_{n-1})\) for \(f = \sum_i a_i X^i\),
coeff. vector \((b_0, \ldots, b_{n-1})\) for \(g = \sum_i b_i X^i\)

**output:** coeff. vector \(c = (c_0, \ldots, c_{n-1})\) for \(h = f + g\)

for \(i\) in \([0..n]\) do
    \(c_i \leftarrow a_i + b_i\)

Naive Multiplication:

**input:** coeff. vector \((a_0, \ldots, a_{n-1})\) for \(f = \sum_i a_i X^i\),
coeff. vector \((b_0, \ldots, b_{n-1})\) for \(g = \sum_i b_i X^i\)

**output:** coeff. vector \(c = (c_0, \ldots, c_{2n-2})\) for \(h = f \cdot g\)

initialize \(c_i \leftarrow 0\) for \(i\) in \([0..2n-2]\)
for \(i\) in \([0..n]\) do
    for \(j \in [0..n]\)
        \(c_{i+j} \leftarrow c_{i+j} + b_i \cdot a_j\)
Karatsuba multiplication for polynomials

Divide input polynomials \( f, g \) into two pieces:

\[
\begin{align*}
f &= f_{\text{hi}}X^k + f_{\text{lo}} \\
g &= g_{\text{hi}}X^k + g_{\text{lo}},
\end{align*}
\]

where \( k := \lfloor n/2 \rfloor \)

Compute \( F \leftarrow f_{\text{hi}} + f_{\text{lo}}, \; G \leftarrow g_{\text{hi}} + g_{\text{lo}} \)

Recursively compute three products:

\[
U \leftarrow f_{\text{hi}}g_{\text{hi}}, \quad V \leftarrow f_{\text{lo}}g_{\text{lo}}, \quad W \leftarrow FG
\]

Return \( UX^{2k} + (W - U - V)X^k + V \)


**O(n log n) Polynomial Multiplication**

**Domain transformation:** polynomial evaluation and interpolation

Let $p_0, \ldots, p_{n-1}$ be fixed, distinct points in $R$

For any polynomial $f \in R[X]$ of degree $< n$, we define its *evaluation vector*

$$(f(p_0), \ldots, f(p_{n-1})) \in R^n$$

Under the right technical conditions (e.g., $R$ is a field), $f$ is uniquely determined by its evaluation vector

- this is just the generalization of “two points determine a line”

We can recover $f$ from its coefficient vector using a “polynomial interpolation” algorithm
The high-level strategy:

- Let $f, g \in R[X]$, with $\text{deg}(f) + \text{deg}(g) < n$
- Evaluate $f$ and $g$ at $n$ points, obtaining their evaluation vectors
- Multiply the evaluation vectors element-wise, obtaining
  \[(f(p_0)g(p_0), \ldots, f(p_{n-1})g(p_{n-1}))\]
  This is the evaluation vector of $h := f \cdot g$, and can be computed using $n$ multiplications in $R$
- Interpolate $h$'s evaluation vector to obtain $h$
Polynomial evaluation

A matrix point of view

Let \((a_0, \ldots, a_{n-1})\) be \(f\)'s coefficient vector

We can write:

\[
\begin{pmatrix}
  f(p_0) \\
  f(p_1) \\
  \vdots \\
  f(p_{n-1})
\end{pmatrix} =
\begin{pmatrix}
  1 & p_0 & \cdots & p_0^{n-1} \\
  1 & p_1 & \cdots & p_1^{n-1} \\
  \vdots & \vdots & \vdots & \vdots \\
  1 & p_{n-1} & \cdots & p_{n-1}^{n-1}
\end{pmatrix}
\begin{pmatrix}
  a_0 \\
  a_1 \\
  \vdots \\
  a_{n-1}
\end{pmatrix}
\]

Vandermonde matrix \(V(p_0, \ldots, p_{n-1})\)
Computation: a general matrix-vector product takes $O(n^2)$ operations in $R$, so this doesn’t help

But:

- the matrix has a special structure
- we are free to choose the points $p_0, \ldots, p_{n-1}$ to make our lives easier

We will see how to do the evaluation and interpolation steps using $O(n \log n)$ operations in $R$

This immediately gives us a polynomial multiplication algorithm that takes $O(n \log n)$ operations in $R$
The Fast Fourier Transform (FFT)

Assume $1 < n = 2^k$ and $2 \in R^*$

$\omega \in R$ is called a *primitive nth root of unity* if $\omega^{n/2} = -1$

Simple algebraic facts:

- $\omega^n = 1$
- $\omega^i = 1$ iff $i \equiv 0 \pmod{n}$
- $\omega^i - 1 \in R^*$ for all $i \not\equiv 0 \pmod{n}$

Our evaluation points will be

$$p_i := \omega^i \text{ for } i \in [0 \ldots n)$$
Example: complex roots of unity

\[ \omega = e^{2\pi i/n} \in \mathbb{C} \]

For \( n = 8 \), \( \omega = \omega_8 \):
Example: roots of unity in finite fields

Let $m$ be a prime with $m \equiv 1 \pmod{n}$

Algebraic facts:

- $\mathbb{Z}_m$ is a field with $m$ elements
- $\mathbb{Z}_m^*$ contains a primitive $n$th root of unity

Example: $m = 17$

- 4 is a primitive 4th root of unity ($4^2 = 16 \equiv -1 \pmod{17}$)
- 2 is a primitive 8th root of unity ($2^2 = 4$)
- 6 is a primitive 16th root of unity ($6^2 \equiv 2 \pmod{17}$)
FFT: basic idea

Let \( f = \sum_{i=0}^{n-1} a_i X^i \in R[X] \)

Split \( f \) into “even” and “odd” parts:

\[
\begin{align*}
  f_0 &= \sum_{i=0}^{n/2} a_{2i} X^i, \\
  f_1 &= \sum_{i=0}^{n/2} a_{2i+1} X^i
\end{align*}
\]

so that

\[
f(X) = f_0(X^2) + X \cdot f_1(X^2)
\]

Plug in \( \omega^j \):

\[
f(\omega^j) = f_0(\omega^{2j}) + \omega^j f_1(\omega^{2j})
\]

Key observations: \( \omega^2 \) is a primitive \((n/2)\)th root of unity, and \( \omega^{2j} = (\omega^2)^k \), where \( k = j \mod n/2 \)
Algorithm FFT:

**input:** a primitive $n$th root of unity $\omega$, and a coefficient vector $(a_0, \ldots, a_{n-1})$ for $f = \sum_i a_i x^i$

**output:** the evaluation vector $(f(\omega^0), \ldots, f(\omega^{n-1}))$

if $n = 1$ then
   return $(a_0)$
else
   $(\alpha_0, \alpha_1, \ldots, \alpha_{n/2-1}) \leftarrow FFT(\omega^2, (a_0, a_2, a_4, \ldots, a_{n-2}))$
   $(\beta_0, \beta_1, \ldots, \beta_{n/2-1}) \leftarrow FFT(\omega^2, (a_1, a_3, a_5, \ldots, a_{n-1}))$
   for $j$ in $[0..n)$ do
      $k \leftarrow j \mod n/2$
      $\gamma_j \leftarrow \alpha_k + \omega^j \beta_k$
   return $(\gamma_0, \gamma_1, \ldots, \gamma_{n-1})$
Divide and conquer analysis:

- split into 2 subproblems each of size $n/2$
- local computation cost $O(n)$
- Therefore, total time $O(n \log n)$
The inverse FFT

Recall matrix point of view:

\[
\begin{pmatrix}
  f(\omega_0) \\
  f(\omega_1) \\
  \vdots \\
  f(\omega_{n-1})
\end{pmatrix}
= V
\begin{pmatrix}
  a_0 \\
  a_1 \\
  \vdots \\
  a_{n-1}
\end{pmatrix}
\]

where \( V = V(\omega^0, \ldots, \omega^{n-1}) \) is the Vandermonde matrix for the points \( \omega^0, \ldots, \omega^{n-1} \).

If \( V \) is invertible, then

\[
\begin{pmatrix}
  a_0 \\
  a_1 \\
  \vdots \\
  a_{n-1}
\end{pmatrix}
= V^{-1}
\begin{pmatrix}
  f(\omega_0) \\
  f(\omega_1) \\
  \vdots \\
  f(\omega_{n-1})
\end{pmatrix}
\]
So ... interpolation corresponds to multiplication by an inverse Vandermonde matrix.

**Theorem.** We have

\[ V^{-1}(\omega^0, \ldots, \omega^{n-1}) = n^{-1} \cdot V(\zeta^0, \ldots, \zeta^{n-1}), \]

where \( \zeta = \omega^{-1} \).

Notes:

- \( \zeta = \omega^{n-1} \) is also a primitive \( n \)th root of unity
- since 2 is a unit, so is \( n = 2^k \)

**Implication:** *interpolation is just another FFT!!* ... plus \( n \) multiplications in \( R \) (by \( n^{-1} \))
Proof of Theorem

We can write $V = (\omega^{ij})$, with indices $i, j \in [0..n)$

Let $W = (\omega^{-ij})$

We want to show that $VW = nI$, where $I$ is the identity matrix.

Let $VW = (Z_{ij})$

Using the usual rules of matrix multiplication,

$$z_{ij} = \sum_s \omega^{is} \omega^{-sj} = \sum_s \omega^{s(i-j)}$$

For diagonal entries ($i = j$), we have $z_{ii} = n$

We want to show that $z_{ij} = 0$ for off-diagonal entries ($i \neq j$)
Let $\Delta := i - j \not\equiv 0 \pmod{n}$.
Want to show:

$$\sum_{s=0}^{n-1} \omega^{s\Delta} = 0$$

General fact:

$$(X^n - 1) = (X - 1) \sum_{s=0}^{n-1} X^s$$

Plug in $\omega^\Delta$:

$$(\omega^{n\Delta} - 1) = (\omega^\Delta - 1) \sum_{s=0}^{n-1} \omega^{s\Delta}$$

Finally,

$$\omega^{n\Delta} - 1 = 0 \text{ and } \omega^\Delta - 1 \in R^* \implies \sum_{s=0}^{n-1} \omega^{s\Delta} = 0$$
Summary:

Using the FFT, we can multiply polynomials over $R$ of length less than $n$ using $O(n \log n)$ operations in $R$

This assumes $2 \in R^*$ and $R$ contains a primitive $n$th root of unity

for complex roots of unity, one can use floating point approximations

rounding error has been extensively analyzed in the literature

for roots of unity in finite fields, there are no issues with rounding

It is possible to generalize to arbitrary rings: operation count becomes $O(n \log n \log \log \log n)$