Polynomial multiplication
(Fast Fourier Transform)

Jordi Cortadella and Jordi Petit
Department of Computer Science
Why polynomial multiplication?
Polynomials: coefficient representation

- A polynomial is represented as a vector of coefficients \((a_0, a_1, \ldots, a_{n-1})\):

  \[
  A(x) = 2x^4 + x^2 - 4x + 3
  \]

  \[
  A = (3, -4, 1, 0, 2)
  \]

- Addition: \(O(n)\)

  \[
  A(x) + B(x) = (a_0 + b_0) + (a_1 + b_1)x + \cdots + (a_{n-1} + b_{n-1})x^{n-1}
  \]

- Evaluation: \(O(n)\) using Horner’s method

  \[
  A(x) = a_0 + (x(a_1 + x(a_2 + \cdots + x(a_{n-2} + x(a_{n-1})) \cdots )))
  \]

- Multiplication: \(O(n^2)\) using brute force

  \[
  A(x) \cdot B(x) = \sum_{i=0}^{2n-2} c_i x^i, \quad \text{where} \quad c_i = \sum_{j=0}^{i} a_j b_{i-j}
  \]
Polynomials: point-value representation

- Fundamental Theorem (Gauss): A degree $n$ polynomial with complex coefficients has exactly $n$ complex roots.

- Corollary: A degree $n - 1$ polynomial $A(x)$ is uniquely identified by its evaluation at $n$ distinct values of $x$. 
Polynomials: point-value representation

- A polynomial is represented as a set of pairs \((x_i, y_i)\):
  \[ A(x) = \{ (x_0, y_0), \ldots, (x_{n-1}, y_{n-1}) \} \]
  \[ B(x) = \{ (x_0, z_0), \ldots, (x_{n-1}, z_{n-1}) \} \]

- Addition: \(O(n)\)
  \[ A(x) + B(x) = \{ (x_0, y_0 + z_0), \ldots, (x_{n-1}, y_{n-1} + z_{n-1}) \} \]

- Multiplication: \(O(n)\), but with \(2n - 1\) points
  \[ A(x) \cdot B(x) = \{ (x_0, y_0 \cdot z_0), \ldots, (x_{n-1}, y_{n-1} \cdot z_{n-1}) \} \]

- Evaluation: \(O(n^2)\) using Lagrange’s formula
  \[ A(x) = \sum_{k=0}^{n-1} y_k \frac{\prod_{j \neq k} (x - x_j)}{\prod_{j \neq k} (x_k - x_j)} \]
Conversion between both representations

<table>
<thead>
<tr>
<th>representation</th>
<th>addition</th>
<th>multiplication</th>
<th>evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>coefficient</td>
<td>$O(n)$</td>
<td>$O(n^2)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>point-value</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
<td>$O(n^2)$</td>
</tr>
</tbody>
</table>

$\{a_0, a_1, \cdots, a_{n-1}\}$

Evaluation

$\{(x_0, y_0), \cdots, (x_{n-1}, y_{n-1})\}$

Interpolation

Coefficient representation

Point-value representation
Given a polynomial $a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$, evaluate it at $n$ different points $x_0, \ldots, x_{n-1}$:

$$
\begin{bmatrix}
  y_0 \\
  y_1 \\
  y_2 \\
  \vdots \\
  y_{n-1}
\end{bmatrix}
= 
\begin{bmatrix}
  1 & x_0 & x_0^2 & \cdots & x_0^{n-1} \\
  1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\
  1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\
  \vdots & \vdots & \vdots & \ddots & \vdots \\
  1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1}
\end{bmatrix}
\begin{bmatrix}
  a_0 \\
  a_1 \\
  a_2 \\
  \vdots \\
  a_{n-1}
\end{bmatrix}
$$

**Runtime:** $O(n^2)$ matrix-vector multiplication (apply Horner $n$ times).
Evaluation by divide-and-conquer


• We want to evaluate $A(x)$ at $n$ different points. Let us choose them to be positive-negative pairs: $\pm x_0, \pm x_1, \ldots, \pm x_{n/2-1}$

• The computations for $A(x_i)$ and $A(-x_i)$ overlap a lot.

• Split the polynomial into odd and even powers

\[3 + 4x + 6x^2 + 2x^3 + x^4 + 10x^5 = (3 + 6x^2 + x^4) + x(4 + 2x^2 + 10x^4)\]

• The terms in parenthesis are polynomials in $x^2$:

\[A(x) = A_e(x^2) + xA_o(x^2)\]
Evaluation by divide-and-conquer

• The calculations needed for $A(x_i)$ can be reused for computing $A(-x_i)$.

\[
A(x_i) = A_e(x_i^2) + x_i A_o(x_i^2) \\
A(-x_i) = A_e(x_i^2) - x_i A_o(x_i^2)
\]

• Evaluating $A(x)$ at $n$ paired points

\[\pm x_0, \pm x_1, \ldots, \pm x_{n/2-1}\]

reduces to evaluating $A_e(x)$ and $A_o(x)$ at just $n/2$ points: $x_0^2, \ldots, x_{n/2-1}^2$
Evaluate by divide-and-conquer

Evaluate: $A(x)$
degree $\leq n - 1$

Evaluate:
$A_e(x)$ and $A_o(x)$
degree $\leq n/2 - 1$

If we could recurse, we would get a running time:

$$T(n) = 2T(n/2) + O(n) = O(n \log n)$$

But can we recurse?
Evaluation by divide-and-conquer

Evaluate: \( A(x) \)
degree \( \leq n - 1 \)

Evaluate:
\( A_e(x) \) and \( A_o(x) \)
degree \( \leq n/2 - 1 \)

The problem: ? We need \( x_0^2 \) and \( x_1^2 \) to be a plus-minus pair. But a square cannot be negative!
Evaluation by divide-and-conquer

Note:
\[
\sqrt{i} = \pm \frac{1}{\sqrt{2}} (1 + i)
\]
\[
\sqrt{-i} = \pm \frac{1}{\sqrt{2}} (1 - i)
\]
Complex numbers: review

\[ z = a + bi \]

\[ z = r(\cos \theta + i \sin \theta) = re^{i\theta} \]

Polar coordinates: \((r, \theta)\)

Length: \(r = \sqrt{a^2 + b^2}\)

Angle \(\theta \in [0,2\pi): \cos \theta = \frac{a}{r}, \sin \theta = \frac{b}{r}\)

\(\theta\) can always be reduced modulo \(2\pi\)

Some examples:

<table>
<thead>
<tr>
<th>Number</th>
<th>(-1)</th>
<th>(i)</th>
<th>(5 + 5i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polar copts</td>
<td>((1, \pi))</td>
<td>((1, \pi/2))</td>
<td>((5\sqrt{2}, \pi/4))</td>
</tr>
</tbody>
</table>
Complex numbers: multiplication

For any \( z = (r, \theta) \):

\[-z = (r, \theta + \pi), \text{ since } -1 = (1, \pi)\]

If \( z \) is on the unit circle, then \( z^n = (1, n\theta) \)

\((r_1, \theta_1) \times (r_2, \theta_2) = (r_1 r_2, \theta_1 + \theta_2)\)
Solutions to the equation $z^n = 1$

$(n = 16)$

Solutions are $z = (1, \theta)$, for $\theta$ a multiple of $2\pi/n$

All roots are plus-minus paired:

$-(1, \theta) = (1, \theta + \pi)$
Divide-and-conquer step

Evaluate $A(x)$ at $n$th roots of unity

Evaluate $A_e(x)$ and $A_o(x)$ at $(n/2)$nd roots of unity
Divide-and-conquer steps
Roots of unity for $n = 8$

$$+i$$

$$-\sqrt{-i}$$

$$-1$$

$$-\sqrt{i}$$

$$-i$$

$$+\sqrt{-i}$$

$$+1$$
Polynomial evaluation (Fast Fourier Transform)

**Function** $\text{FFT}(a,\omega)$

*Inputs:* $a = (a_0, a_1, ..., a_{n-1})$, for $n$ a power of 2

*\omega:* A primitive $n$th root of unity

*Output:* $(a(1), a(\omega), a(\omega^2), ..., a(\omega^{n-1}))$

**if** $w=1$: **return** $a$

$(s_0, s_1, ..., s_{n/2-1}) = \text{FFT}((a_0, a_2, ..., a_{n-2}), \omega^2)$

$(s'_0, s'_1, ..., s'_{n/2-1}) = \text{FFT}((a_1, a_3, ..., a_{n-1}), \omega^2)$

**for** $j = 0$ **to** $n/2 - 1$:

$r_j = s_j + \omega^j s'_j$

$r_{j+n/2} = s_j - \omega^j s'_j$

**return** $(r_0, r_1, ..., r_{n-1})$
Unfolding the FFT

\[ a_0, a_2, \ldots, a_{n-2}, a_1, a_3, \ldots, a_{n-1} \]

\[ FFT_{n/2} \]

\[ r_j, r_{j+n/2} \]

\[ \omega_j, \omega^{j+n/2} \]
Unfolding the FFT

Divide & Conquer

© Dept. CS, UPC

\[ a_0, a_4, a_2, a_6, a_1, a_5, a_3, a_7 \]

\[ A(\omega^0), A(\omega^1), A(\omega^2), A(\omega^3), A(\omega^4), A(\omega^5), A(\omega^6), A(\omega^7) \]
Conversion between both representations

<table>
<thead>
<tr>
<th>representation</th>
<th>addition</th>
<th>multiplication</th>
<th>evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>coefficient</td>
<td>$O(n)$</td>
<td>$O(n^2)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>point-value</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
<td>$O(n^2)$</td>
</tr>
</tbody>
</table>

\[
\langle \text{values} \rangle = \text{FFT}(\langle \text{coefficients} \rangle, \omega)
\]

\[
\langle \text{coefficients} \rangle = \frac{1}{n} \text{FFT}(\langle \text{values} \rangle, \omega^{-1})
\]

Coefficient representation \( a_0, a_1, \ldots, a_{n-1} \) \hspace{1cm} \text{interpolation} \hspace{1cm} \text{evaluation} \hspace{1cm} \text{Point-value representation} \( (x_0, y_0), \ldots, (x_{n-1}, y_{n-1}) \)
FFT application in Signal Processing

\[ \sum_{n=1}^{5} n \times \cos(n \times \omega \times t), \quad \omega = 10 \times 2\pi \]

Converting a signal: time domain \(\leftrightarrow\) frequency domain