CSE 548 / AMS 542: Analysis of Algorithms

Lecture 4
( Divide-and-Conquer Algorithms: Polynomial Multiplication )

Rezaul A. Chowdhury
Department of Computer Science
SUNY Stony Brook
Fall 2023
Coefficient Representation of Polynomials

\[ A(x) = \sum_{j=0}^{n-1} a_j x^j \]

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

\( A(x) \) is a polynomial of degree bound \( n \) represented as a vector \( a = (a_0, a_1, \ldots, a_{n-1}) \) of coefficients.

The **degree** of \( A(x) \) is \( k \) provided it is the largest integer such that \( a_k \) is nonzero. Clearly, \( 0 \leq k \leq n - 1 \).

**Evaluating \( A(x) \) at a given point:**

Takes \( \Theta(n) \) time using Horner’s rule:

\[ A(x_0) = a_0 + a_1 x_0 + a_2 (x_0)^2 + \cdots + a_{n-1} (x_0)^{n-1} \]
\[ = a_0 + x_0 \left( a_1 + x_0 (a_2 + \cdots + x_0 (a_{n-2} + x_0 (a_{n-1}) \cdots) \right) \]
Coefficient Representation of Polynomials

Adding Two Polynomials:

Adding two polynomials of degree bound \( n \) takes \( \Theta(n) \) time.

\[
C(x) = A(x) + B(x)
\]

where, \( A(x) = \sum_{j=0}^{n-1} a_j x^j \) and \( B(x) = \sum_{j=0}^{n-1} b_j x^j \).

Then \( C(x) = \sum_{j=0}^{n-1} c_j x^j \), where, \( c_j = a_j + b_j \) for \( 0 \leq j \leq n - 1 \).
**Coefficient Representation of Polynomials**

**Multiplying Two Polynomials:**

The product of two polynomials of degree bound $n$ is another polynomial of degree bound $2n - 1$.

\[
C(x) = A(x)B(x)
\]

where, 

\[
A(x) = \sum_{j=0}^{n-1} a_j x^j \quad \text{and} \quad B(x) = \sum_{j=0}^{n-1} b_j x^j.
\]

Then 

\[
C(x) = \sum_{j=0}^{2n-2} c_j x^j \quad \text{where}, \quad c_j = \sum_{k=0}^{j} a_k b_{j-k} \quad \text{for} \quad 0 \leq j \leq 2n - 2.
\]

The coefficient vector $c = (c_0, c_1, \cdots, c_{2n-2})$, denoted by $c = a \otimes b$, is also called the *convolution* of vectors $a = (a_0, a_1, \cdots, a_{n-1})$ and $b = (b_0, b_1, \cdots, b_{n-1})$.

Clearly, straightforward evaluation of $c$ takes $\Theta(n^2)$ time.
Polynomial Multiplication and Convolution

\[
\begin{align*}
(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3) &= a_0 b_0 \\
+ \cdots
\end{align*}
\]
Polynomial Multiplication and Convolution

\[
(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3) = a_0 b_0 + (a_0 b_1 + a_1 b_0) x + \cdots
\]
Polynomial Multiplication and Convolution

\((a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3)\)

\[= a_0 b_0 + (a_0 b_1 + a_1 b_0) x + (a_0 b_2 + a_1 b_1 + a_2 b_0) x^2 + \ldots\]
Polynomial Multiplication and Convolution

\[
\begin{align*}
(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3) &= a_0 b_0 \\
&\quad + (a_0 b_1 + a_1 b_0)x \\
&\quad + (a_0 b_2 + a_1 b_1 + a_2 b_0)x^2 \\
&\quad + (a_0 b_3 + a_1 b_2 + a_2 b_1 + a_3 b_0)x^3 \\
&\quad + \ldots
\end{align*}
\]
Polynomial Multiplication and Convolution

\[
(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3)
= a_0 b_0 \\
\quad + (a_0 b_1 + a_1 b_0) x \\
\quad + (a_0 b_2 + a_1 b_1 + a_2 b_0) x^2 \\
\quad + (a_0 b_3 + a_1 b_2 + a_2 b_1 + a_3 b_0) x^3 \\
\quad + (a_1 b_3 + a_2 b_2 + a_3 b_1) x^4 \\
\quad + \ldots
\]
Polynomial Multiplication and Convolution

\[(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3) = a_0 b_0 + (a_0 b_1 + a_1 b_0) x + (a_0 b_2 + a_1 b_1 + a_2 b_0) x^2 + (a_0 b_3 + a_1 b_2 + a_2 b_1 + a_3 b_0) x^3 + (a_1 b_3 + a_2 b_2 + a_3 b_1) x^4 + (a_2 b_3 + a_3 b_2) x^5 + \ldots\]
Polynomial Multiplication and Convolution

\[
(a_0 + a_1 x + a_2 x^2 + a_3 x^3) \times (b_0 + b_1 x + b_2 x^2 + b_3 x^3)
= a_0 b_0 \\
+ (a_0 b_1 + a_1 b_0) x \\
+ (a_0 b_2 + a_1 b_1 + a_2 b_0) x^2 \\
+ (a_0 b_3 + a_1 b_2 + a_2 b_1 + a_3 b_0) x^3 \\
+ (a_1 b_3 + a_2 b_2 + a_3 b_1 + a_4 b_0) x^4 \\
+ (a_2 b_3 + a_3 b_2 + a_4 b_1) x^5 \\
+ (a_3 b_3) x^6
\]
Coefficient Representation of Polynomials

Multiplying Two Polynomials:

We can use Karatsuba’s algorithm (assume $n$ to be a power of 2):

$$A(x) = \sum_{j=0}^{n-1} a_j x^j = \sum_{j=0}^{\frac{n}{2}-1} a_j x^j + x^{\frac{n}{2}} \sum_{j=0}^{\frac{n}{2}-1} a_{n+j} x^j = A_1(x) + x^{\frac{n}{2}} A_2(x)$$

$$B(x) = \sum_{j=0}^{n-1} b_j x^j = \sum_{j=0}^{\frac{n}{2}-1} b_j x^j + x^{\frac{n}{2}} \sum_{j=0}^{\frac{n}{2}-1} b_{n+j} x^j = B_1(x) + x^{\frac{n}{2}} B_2(x)$$

Then $C(x) = A(x)B(x)$

$$= A_1(x)B_1(x) + x^{\frac{n}{2}}[A_1(x)B_2(x) + A_2(x)B_1(x)] + x^n A_2(x)B_2(x)$$

But $A_1(x)B_2(x) + A_2(x)B_1(x)$

$$= [A_1(x) + A_2(x)][B_1(x) + B_2(x)] - A_1(x)B_1(x) - A_2(x)B_2(x)$$

3 recursive multiplications of polynomials of degree bound $\frac{n}{2}$. Similar recurrence as in Karatsuba’s integer multiplication algorithm leading to a complexity of $O(n^{\log_2 3}) = O(n^{1.59})$. 12
Point-Value Representation of Polynomials

A point-value representation of a polynomial $A(x)$ is a set of $n$ point-value pairs $\{(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})\}$ such that all $x_k$ are distinct and $y_k = A(x_k)$ for $0 \leq k \leq n - 1$.

A polynomial has many point-value representations.

Adding Two Polynomials:

Suppose we have point-value representations of two polynomials of degree bound $n$ using the same set of $n$ points.

Let $A: \{(x_0, y_0^a), (x_1, y_1^a), \ldots, (x_{n-1}, y_{n-1}^a)\}$ and $B: \{(x_0, y_0^b), (x_1, y_1^b), \ldots, (x_{n-1}, y_{n-1}^b)\}$.

If $C(x) = A(x) + B(x)$ then

$$C: \{(x_0, y_0^a + y_0^b), (x_1, y_1^a + y_1^b), \ldots, (x_{n-1}, y_{n-1}^a + y_{n-1}^b)\}$$

Thus polynomial addition takes $\Theta(n)$ time.
Point-Value Representation of Polynomials

Multiplying Two Polynomials:

Suppose we have extended (why?) point-value representations of two polynomials of degree bound $n$ using the same set of $2n$ points.

$A: \{(x_0, y_0^a), (x_1, y_1^a), \ldots, (x_{2n-1}, y_{2n-1}^a)\}$

$B: \{(x_0, y_0^b), (x_1, y_1^b), \ldots, (x_{2n-1}, y_{2n-1}^b)\}$

If $C(x) = A(x)B(x)$ then

$C: \{(x_0, y_0^a y_0^b), (x_1, y_1^a y_1^b), \ldots, (x_{2n-1}, y_{2n-1}^a y_{2n-1}^b)\}$

Thus polynomial multiplication also takes only $\Theta(n)$ time!

( compare this with the $\Theta(n^2)$ time needed in the coefficient form )
Faster Polynomial Multiplication? (in Coefficient Form)

\[ A(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \]
\[ B(x) = b_0 + b_1 x + \cdots + b_{n-1} x^{n-1} \]
\[ C(x) = c_0 + c_1 x + \cdots + c_{2n-1} x^{2n-1} \]
Faster Polynomial Multiplication? (in Coefficient Form)

Coefficient Representation $\Rightarrow$ Point-Value Representation:

We select any set of $n$ distinct points \( \{x_0, x_1, ..., x_{n-1} \} \), and evaluate \( A(x_k) \) for \( 0 \leq k \leq n - 1 \).

Using Horner’s rule this approach takes $\Theta(n^2)$ time.

Point-Value Representation $\Rightarrow$ Coefficient Representation:

We can interpolate using Lagrange’s formula:

\[
A(x) = \sum_{k=0}^{n-1} \frac{\Pi_{j \neq k}(x - x_j)}{\Pi_{j \neq k}(x_k - x_j)} y_k
\]

This again takes $\Theta(n^2)$ time.

In both cases we need to do much better!
A polynomial of degree bound $n$: $A(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$

A set of $n$ distinct points: $\{x_0, x_1, \ldots, x_{n-1}\}$

Compute point-value form: $\{(x_0, A(x_0)), (x_1, A(x_1)), \ldots, (x_{n-1}, A(x_{n-1}))\}$

Using matrix notation:

$$
\begin{bmatrix}
A(x_0) \\
A(x_1) \\
\vdots \\
A(x_{n-1})
\end{bmatrix}
= 
\begin{bmatrix}
1 & x_0 & (x_0)^2 & \cdots & (x_0)^{n-1} \\
1 & x_1 & (x_1)^2 & \cdots & (x_1)^{n-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_{n-1} & (x_{n-1})^2 & \cdots & (x_{n-1})^{n-1}
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
\vdots \\
a_{n-1}
\end{bmatrix}
$$

We want to choose the set of points in a way that simplifies the multiplication.

In the rest of the lecture on this topic we will assume:

$n$ is a power of 2.
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_1 x_0 + a_2 (x_0)^2 + a_3 (x_0)^3 + a_4 (x_0)^4 + a_5 (x_0)^5 + a_6 (x_0)^6 + a_7 (x_0)^7 \\
A(x_1) &= a_0 + a_1 x_1 + a_2 (x_1)^2 + a_3 (x_1)^3 + a_4 (x_1)^4 + a_5 (x_1)^5 + a_6 (x_1)^6 + a_7 (x_1)^7 \\
A(x_2) &= a_0 + a_1 x_2 + a_2 (x_2)^2 + a_3 (x_2)^3 + a_4 (x_2)^4 + a_5 (x_2)^5 + a_6 (x_2)^6 + a_7 (x_2)^7 \\
A(x_3) &= a_0 + a_1 x_3 + a_2 (x_3)^2 + a_3 (x_3)^3 + a_4 (x_3)^4 + a_5 (x_3)^5 + a_6 (x_3)^6 + a_7 (x_3)^7
\end{align*}
\]

\[
\begin{array}{c|c}
\text{Strategy:} & \\
\hline
x_4 = -x_0 & A(x_4) = a_0 + a_1 x_4 + a_2 (x_4)^2 + a_3 (x_4)^3 + a_4 (x_4)^4 + a_5 (x_4)^5 + a_6 (x_4)^6 + a_7 (x_4)^7 \\
x_5 = -x_1 & A(x_5) = a_0 + a_1 x_5 + a_2 (x_5)^2 + a_3 (x_5)^3 + a_4 (x_5)^4 + a_5 (x_5)^5 + a_6 (x_5)^6 + a_7 (x_5)^7 \\
x_6 = -x_2 & A(x_6) = a_0 + a_1 x_6 + a_2 (x_6)^2 + a_3 (x_6)^3 + a_4 (x_6)^4 + a_5 (x_6)^5 + a_6 (x_6)^6 + a_7 (x_6)^7 \\
x_7 = -x_3 & A(x_7) = a_0 + a_1 x_7 + a_2 (x_7)^2 + a_3 (x_7)^3 + a_4 (x_7)^4 + a_5 (x_7)^5 + a_6 (x_7)^6 + a_7 (x_7)^7 \\
\end{array}
\]

STRATEGY: Set \( x_{4+j} = -x_j \) for \( 0 \leq j < 4 \)
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_1 x_0 + a_2 (x_0)^2 + a_3 (x_0)^3 + a_4 (x_0)^4 + a_5 (x_0)^5 + a_6 (x_0)^6 + a_7 (x_0)^7 \\
A(x_1) &= a_0 + a_1 x_1 + a_2 (x_1)^2 + a_3 (x_1)^3 + a_4 (x_1)^4 + a_5 (x_1)^5 + a_6 (x_1)^6 + a_7 (x_1)^7 \\
A(x_2) &= a_0 + a_1 x_2 + a_2 (x_2)^2 + a_3 (x_2)^3 + a_4 (x_2)^4 + a_5 (x_2)^5 + a_6 (x_2)^6 + a_7 (x_2)^7 \\
A(x_3) &= a_0 + a_1 x_3 + a_2 (x_3)^2 + a_3 (x_3)^3 + a_4 (x_3)^4 + a_5 (x_3)^5 + a_6 (x_3)^6 + a_7 (x_3)^7 \\
\end{align*}
\]

| \(x_4 = -x_0\) | \(A(-x_0) = a_0 - a_1 x_0 + a_2 (x_0)^2 - a_3 (x_0)^3 + a_4 (x_0)^4 - a_5 (x_0)^5 + a_6 (x_0)^6 - a_7 (x_0)^7\) |
| \(x_5 = -x_1\) | \(A(-x_1) = a_0 - a_1 x_1 + a_2 (x_1)^2 - a_3 (x_1)^3 + a_4 (x_1)^4 - a_5 (x_1)^5 + a_6 (x_1)^6 - a_7 (x_1)^7\) |
| \(x_6 = -x_2\) | \(A(-x_2) = a_0 - a_1 x_2 + a_2 (x_2)^2 - a_3 (x_2)^3 + a_4 (x_2)^4 - a_5 (x_2)^5 + a_6 (x_2)^6 - a_7 (x_2)^7\) |
| \(x_7 = -x_3\) | \(A(-x_3) = a_0 - a_1 x_3 + a_2 (x_3)^2 - a_3 (x_3)^3 + a_4 (x_3)^4 - a_5 (x_3)^5 + a_6 (x_3)^6 - a_7 (x_3)^7\) |

**STRATEGY:** Set \(x_{4+j} = -x_j\) for \(0 \leq j < 4\)
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_2(x_0)^2 + a_4(x_0)^4 + a_6(x_0)^6 + a_1 x_0 + a_3(x_0)^3 + a_5(x_0)^5 + a_7(x_0)^7 \\
A(x_1) &= a_0 + a_2(x_1)^2 + a_4(x_1)^4 + a_6(x_1)^6 + a_1 x_1 + a_3(x_1)^3 + a_5(x_1)^5 + a_7(x_1)^7 \\
A(x_2) &= a_0 + a_2(x_2)^2 + a_4(x_2)^4 + a_6(x_2)^6 + a_1 x_2 + a_3(x_2)^3 + a_5(x_2)^5 + a_7(x_2)^7 \\
A(x_3) &= a_0 + a_2(x_3)^2 + a_4(x_3)^4 + a_6(x_3)^6 + a_1 x_3 + a_3(x_3)^3 + a_5(x_3)^5 + a_7(x_3)^7
\end{align*}
\]

| \(x_4 = -x_0\) | \(A(-x_0)\) & = & \(a_0 + a_2(x_0)^2 + a_4(x_0)^4 + a_6(x_0)^6 - a_1 x_0 - a_3(x_0)^3 - a_5(x_0)^5 - a_7(x_0)^7\) \\
| \(x_5 = -x_1\) | \(A(-x_1)\) & = & \(a_0 + a_2(x_1)^2 + a_4(x_1)^4 + a_6(x_1)^6 - a_1 x_1 - a_3(x_1)^3 - a_5(x_1)^5 - a_7(x_1)^7\) \\
| \(x_6 = -x_2\) | \(A(-x_2)\) & = & \(a_0 + a_2(x_2)^2 + a_4(x_2)^4 + a_6(x_2)^6 - a_1 x_2 - a_3(x_2)^3 - a_5(x_2)^5 - a_7(x_2)^7\) \\
| \(x_7 = -x_3\) | \(A(-x_3)\) & = & \(a_0 + a_2(x_3)^2 + a_4(x_3)^4 + a_6(x_3)^6 - a_1 x_3 - a_3(x_3)^3 - a_5(x_3)^5 - a_7(x_3)^7\) \\

**STRATEGY:** Set \(x_{4+j} = -x_j\) for \(0 \leq j < 4\)
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 \]

\[ A(x_0) = (a_0 + a_2(x_0)^2 + a_4(x_0)^4 + a_6(x_0)^6) + x_0 (a_1 + a_3(x_0)^2 + a_5(x_0)^4 + a_7(x_0)^6) \]
\[ A(x_1) = (a_0 + a_2(x_1)^2 + a_4(x_1)^4 + a_6(x_1)^6) + x_1 (a_1 + a_3(x_1)^2 + a_5(x_1)^4 + a_7(x_1)^6) \]
\[ A(x_2) = (a_0 + a_2(x_2)^2 + a_4(x_2)^4 + a_6(x_2)^6) + x_2 (a_1 + a_3(x_2)^2 + a_5(x_2)^4 + a_7(x_2)^6) \]
\[ A(x_3) = (a_0 + a_2(x_3)^2 + a_4(x_3)^4 + a_6(x_3)^6) + x_3 (a_1 + a_3(x_3)^2 + a_5(x_3)^4 + a_7(x_3)^6) \]

| \( x_4 = -x_0 \) | \( A(-x_0) = (a_0 + a_2(x_0)^2 + a_4(x_0)^4 + a_6(x_0)^6) - x_0 (a_1 + a_3(x_0)^2 + a_5(x_0)^4 + a_7(x_0)^6) \) |
| \( x_5 = -x_1 \) | \( A(-x_1) = (a_0 + a_2(x_1)^2 + a_4(x_1)^4 + a_6(x_1)^6) - x_1 (a_1 + a_3(x_1)^2 + a_5(x_1)^4 + a_7(x_1)^6) \) |
| \( x_6 = -x_2 \) | \( A(-x_2) = (a_0 + a_2(x_2)^2 + a_4(x_2)^4 + a_6(x_2)^6) - x_2 (a_1 + a_3(x_2)^2 + a_5(x_2)^4 + a_7(x_2)^6) \) |
| \( x_7 = -x_3 \) | \( A(-x_3) = (a_0 + a_2(x_3)^2 + a_4(x_3)^4 + a_6(x_3)^6) - x_3 (a_1 + a_3(x_3)^2 + a_5(x_3)^4 + a_7(x_3)^6) \) |

**STRATEGY:** Set \( x_{4+j} = -x_j \) for \( 0 \leq j < 4 \)
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3 \]

\[ A_{\text{odd}}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3 \]

\[
\begin{align*}
A(x_0) &= (a_0 + a_2 (x_0^2) + a_4 (x_0^2)^2 + a_6 (x_0^2)^3) + x_0 (a_1 + a_3 (x_0^2) + a_5 (x_0^2)^2 + a_7 (x_0^2)^3) \\
A(x_1) &= (a_0 + a_2 (x_1^2) + a_4 (x_1^2)^2 + a_6 (x_1^2)^3) + x_1 (a_1 + a_3 (x_1^2) + a_5 (x_1^2)^2 + a_7 (x_1^2)^3) \\
A(x_2) &= (a_0 + a_2 (x_2^2) + a_4 (x_2^2)^2 + a_6 (x_2^2)^3) + x_2 (a_1 + a_3 (x_2^2) + a_5 (x_2^2)^2 + a_7 (x_2^2)^3) \\
A(x_3) &= (a_0 + a_2 (x_3^2) + a_4 (x_3^2)^2 + a_6 (x_3^2)^3) + x_3 (a_1 + a_3 (x_3^2) + a_5 (x_3^2)^2 + a_7 (x_3^2)^3) \\
A(-x_0) &= (a_0 + a_2 (x_0^2) + a_4 (x_0^2)^2 + a_6 (x_0^2)^3) - x_0 (a_1 + a_3 (x_0^2) + a_5 (x_0^2)^2 + a_7 (x_0^2)^3) \\
A(-x_1) &= (a_0 + a_2 (x_1^2) + a_4 (x_1^2)^2 + a_6 (x_1^2)^3) - x_1 (a_1 + a_3 (x_1^2) + a_5 (x_1^2)^2 + a_7 (x_1^2)^3) \\
A(-x_2) &= (a_0 + a_2 (x_2^2) + a_4 (x_2^2)^2 + a_6 (x_2^2)^3) - x_2 (a_1 + a_3 (x_2^2) + a_5 (x_2^2)^2 + a_7 (x_2^2)^3) \\
A(-x_3) &= (a_0 + a_2 (x_3^2) + a_4 (x_3^2)^2 + a_6 (x_3^2)^3) - x_3 (a_1 + a_3 (x_3^2) + a_5 (x_3^2)^2 + a_7 (x_3^2)^3)
\end{align*}
\]

**STRATEGY:** Set \( x_{4+j} = -x_j \) for \( 0 \leq j < 4 \)
**Given a Polynomial of Degree Bound 8**

**Find 8 Distinct Points to Efficiently Evaluate it at**

\[
A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7
\]

\[
A_{\text{even}}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3
\]

\[
A_{\text{odd}}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3
\]

<table>
<thead>
<tr>
<th>( x_0 )</th>
<th>( A(x_0) = )</th>
<th>( A_{\text{even}}(x_0^2) )</th>
<th>+</th>
<th>( x_0 A_{\text{odd}}(x_0^2) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( A(x_1) = )</td>
<td>( A_{\text{even}}(x_1^2) )</td>
<td>+</td>
<td>( x_1 A_{\text{odd}}(x_1^2) )</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( A(x_2) = )</td>
<td>( A_{\text{even}}(x_2^2) )</td>
<td>+</td>
<td>( x_2 A_{\text{odd}}(x_2^2) )</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>( A(x_3) = )</td>
<td>( A_{\text{even}}(x_3^2) )</td>
<td>+</td>
<td>( x_3 A_{\text{odd}}(x_3^2) )</td>
</tr>
<tr>
<td>( x_4 = -x_0 )</td>
<td>( A(-x_0) = )</td>
<td>( A_{\text{even}}(x_0^2) )</td>
<td>−</td>
<td>( x_0 A_{\text{odd}}(x_0^2) )</td>
</tr>
<tr>
<td>( x_5 = -x_1 )</td>
<td>( A(-x_1) = )</td>
<td>( A_{\text{even}}(x_1^2) )</td>
<td>−</td>
<td>( x_1 A_{\text{odd}}(x_1^2) )</td>
</tr>
<tr>
<td>( x_6 = -x_2 )</td>
<td>( A(-x_2) = )</td>
<td>( A_{\text{even}}(x_2^2) )</td>
<td>−</td>
<td>( x_2 A_{\text{odd}}(x_2^2) )</td>
</tr>
<tr>
<td>( x_7 = -x_3 )</td>
<td>( A(-x_3) = )</td>
<td>( A_{\text{even}}(x_3^2) )</td>
<td>−</td>
<td>( x_3 A_{\text{odd}}(x_3^2) )</td>
</tr>
</tbody>
</table>

**STRATEGY:** Set \( x_{4+j} = -x_j \) for \( 0 \leq j < 4 \)
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3 \]
\[ A_{\text{odd}}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3 \]

\[
\begin{align*}
A(x_0) &= A_{\text{even}}(x_0^2) + x_0 A_{\text{odd}}(x_0^2) \\
A(x_1) &= A_{\text{even}}(x_1^2) + x_1 A_{\text{odd}}(x_1^2) \\
A(x_2) &= A_{\text{even}}(x_2^2) + x_2 A_{\text{odd}}(x_2^2) \\
A(x_3) &= A_{\text{even}}(x_3^2) + x_3 A_{\text{odd}}(x_3^2)
\end{align*}
\]

\[
\begin{align*}
x_4 &= -x_0 \\
x_5 &= -x_1 \\
x_6 &= -x_2 \\
x_7 &= -x_3
\end{align*}
\]

\[
\begin{align*}
A(-x_0) &= A_{\text{even}}(x_0^2), \\
A(-x_1) &= A_{\text{even}}(x_1^2), \\
A(-x_2) &= A_{\text{even}}(x_2^2), \\
A(-x_3) &= A_{\text{even}}(x_3^2)
\end{align*}
\]

**STRATEGY:** Set \( x_{4+j} = -x_j \) for \( 0 \leq j < 4 \)

We save roughly half the work.
Given a Polynomial of Degree Bound 2
Find 2 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x \]

\[ A(x_0) = a_0 + a_1 x_0 \]

\[ x_1 = -x_0 \]

\[ A(x_1) = a_0 + a_1 x_1 \]

STRATEGY: Set \( x_{1+j} = -x_j \) for \( 0 \leq j < 1 \)
Given a Polynomial of Degree Bound 2
Find 2 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x \]

\[
\begin{array}{c|c}
A(x_0) & a_0 + a_1 x_0 \\
A(-x_0) & a_0 - a_1 x_0 \\
\end{array}
\]

STRATEGY: Set \( x_{1+j} = -x_j \) for \( 0 \leq j < 1 \)
\[ A(x) = a_0 + a_1 x \]

| \( x_0 = 1 \) | \( A(x_0) = a_0 + a_1 \) |
| \( x_1 = -1 \) | \( A(x_1) = a_0 - a_1 \) |

**STRATEGY:** We will evaluate any polynomial of degree bound 2 at

\( x_0 = 1 \)
\( x_1 = -1 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1x + a_2x^2 + a_3x^3 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_1x_0 + a_2(x_0)^2 + a_3(x_0)^3 \\
A(x_1) &= a_0 + a_1x_1 + a_2(x_1)^2 + a_3(x_1)^3 \\
x_2 = -x_0 & \quad A(x_2) = a_0 + a_1x_2 + a_2(x_2)^2 + a_3(x_2)^3 \\
x_3 = -x_1 & \quad A(x_3) = a_0 + a_1x_3 + a_2(x_3)^2 + a_3(x_3)^3
\end{align*}
\]

**STRATEGY:** Set \( x_{2+j} = -x_j \) for \( 0 \leq j < 2 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_1 x_0 + a_2 (x_0)^2 + a_3 (x_0)^3 \\
A(x_1) &= a_0 + a_1 x_1 + a_2 (x_1)^2 + a_3 (x_1)^3 \\
x_2 = -x_0 &\quad A(-x_0) = a_0 - a_1 x_0 + a_2 (x_0)^2 - a_3 (x_0)^3 \\
x_3 = -x_1 &\quad A(-x_1) = a_0 - a_1 x_1 + a_2 (x_1)^2 - a_3 (x_1)^3
\end{align*}
\]

**STRATEGY:** Set \( x_{2+j} = -x_j \) for \( 0 \leq j < 2 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[
\begin{align*}
A(x_0) &= a_0 + a_2(x_0)^2 + a_1 x_0 + a_3 (x_0)^3 \\
A(x_1) &= a_0 + a_2(x_1)^2 + a_1 x_1 + a_3 (x_1)^3 \\
x_2 = -x_0 & \quad A(-x_0) = a_0 + a_2(x_0)^2 - a_1 x_0 - a_3 (x_0)^3 \\
x_3 = -x_1 & \quad A(-x_1) = a_0 + a_2(x_1)^2 - a_1 x_1 - a_3 (x_1)^3
\end{align*}
\]

**STRATEGY:** Set \( x_{2+j} = -x_j \) for \( 0 \leq j < 2 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[
\begin{align*}
A(x_0) &= (a_0 + a_2 (x_0)^2) + x_0 (a_1 + a_3 (x_0)^2) \\
A(x_1) &= (a_0 + a_2 (x_1)^2) + x_1 (a_1 + a_3 (x_1)^2)
\end{align*}
\]

\[
\begin{align*}
x_2 = -x_0 & \quad A(-x_0) = (a_0 + a_2 (x_0)^2) - x_0 (a_1 + a_3 (x_0)^2) \\
x_3 = -x_1 & \quad A(-x_1) = (a_0 + a_2 (x_1)^2) - x_1 (a_1 + a_3 (x_1)^2)
\end{align*}
\]

**STRATEGY:** Set \( x_{2+j} = -x_j \) for \( 0 \leq j < 2 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x \]
\[ A_{\text{odd}}(x) = a_1 + a_3 x \]

\[
\begin{align*}
A(x_0) &= (a_0 + a_2 (x_0^2)) + x_0 (a_1 + a_3 (x_0^2)) \\
A(x_1) &= (a_0 + a_2 (x_1^2)) + x_1 (a_1 + a_3 (x_1^2)) \\
A(-x_0) &= (a_0 + a_2 (x_0^2)) - x_0 (a_1 + a_3 (x_0^2)) \\
A(-x_1) &= (a_0 + a_2 (x_1^2)) - x_1 (a_1 + a_3 (x_1^2))
\end{align*}
\]

**STRATEGY:** Set \(x_{2+j} = -x_j\) for \(0 \leq j < 2\)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[ A_{even}(x) = a_0 + a_2 x \]
\[ A_{odd}(x) = a_1 + a_3 x \]

\[
\begin{array}{l}
A(x_0) = A_{even}(x_0^2) + x_0 A_{odd}(x_0^2) \\
A(x_1) = A_{even}(x_1^2) + x_1 A_{odd}(x_1^2) \\
A(-x_0) = A_{even}(x_0^2) - x_0 A_{odd}(x_0^2) \\
A(-x_1) = A_{even}(x_1^2) - x_1 A_{odd}(x_1^2)
\end{array}
\]

\[ x_2 = -x_0 \]
\[ x_3 = -x_1 \]

**STRATEGY:** Set \( x_2+j = -x_j \) for \( 0 \leq j < 2 \)
**Given a Polynomial of Degree Bound 4**

**Find 4 Distinct Points to Efficiently Evaluate it at**

\[
A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3
\]

\[
A_{\text{even}}(x) = a_0 + a_2 x
\]

\[
A_{\text{odd}}(x) = a_1 + a_3 x
\]

**STRATEGY:** Set \( x_{2+j} = -x_j \) for \( 0 \leq j < 2 \)
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[
A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3
\]

\[
A_{\text{even}}(x) = a_0 + a_2 x
\]

\[
A_{\text{odd}}(x) = a_1 + a_3 x
\]

\[
A(x_0) = A_{\text{even}}(x_0^2) + x_0 A_{\text{odd}}(x_0^2)
\]

\[
A(x_1) = A_{\text{even}}(x_1^2) + x_1 A_{\text{odd}}(x_1^2)
\]

\[
x_2 = -x_0\quad A(-x_0) = A_{\text{even}}(x_0^2) - x_0 A_{\text{odd}}(x_0^2)
\]

\[
x_3 = -x_1\quad A(-x_1) = A_{\text{even}}(x_1^2) - x_1 A_{\text{odd}}(x_1^2)
\]

Observe that we evaluate both \(A_{\text{even}}(x)\) and \(A_{\text{odd}}(x)\) at \(x = x_0^2\) and \(x = x_1^2\).

But we decided to always evaluate polynomials of degree bound 2 at \(x = 1\) and \(x = -1\).

So, \(x_0^2 = 1 \Rightarrow x_0 = 1\) and \(x_1^2 = -1 \Rightarrow x_1 = \sqrt{-1} = i\).
Given a Polynomial of Degree Bound 4
Find 4 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x \]

\[ A_{\text{odd}}(x) = a_1 + a_3 x \]

\[
\begin{align*}
A(x_0) &= &A_{\text{even}}(x_0^2) &+ &x_0 A_{\text{odd}}(x_0^2) \\
A(x_1) &= &A_{\text{even}}(x_1^2) &+ &x_1 A_{\text{odd}}(x_1^2) \\
x_2 = -x_0 &\quad A(-x_0) = &A_{\text{even}}(x_0^2) &- &x_0 A_{\text{odd}}(x_0^2) \\
x_3 = -x_1 &\quad A(-x_1) = &A_{\text{even}}(x_1^2) &- &x_1 A_{\text{odd}}(x_1^2)
\end{align*}
\]

So, we evaluate any polynomial of degree bound 4 at
\[ x_0 = 1, \quad x_1 = i \]
and
\[ x_2 = -x_0 = -1, \quad x_3 = -x_1 = -i \]
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3 \]
\[ A_{\text{odd}}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3 \]

\[
\begin{align*}
A(x_0) &= A_{\text{even}}(x_0^2) + x_0 A_{\text{odd}}(x_0^2) \\
A(x_1) &= A_{\text{even}}(x_1^2) + x_1 A_{\text{odd}}(x_1^2) \\
A(x_2) &= A_{\text{even}}(x_2^2) + x_2 A_{\text{odd}}(x_2^2) \\
A(x_3) &= A_{\text{even}}(x_3^2) + x_3 A_{\text{odd}}(x_3^2) \\
x_4 = -x_0 &\quad A(-x_0) = A_{\text{even}}(x_0^2) - x_0 A_{\text{odd}}(x_0^2) \\
x_5 = -x_1 &\quad A(-x_1) = A_{\text{even}}(x_1^2) - x_1 A_{\text{odd}}(x_1^2) \\
x_6 = -x_2 &\quad A(-x_2) = A_{\text{even}}(x_2^2) - x_2 A_{\text{odd}}(x_2^2) \\
x_7 = -x_3 &\quad A(-x_3) = A_{\text{even}}(x_3^2) - x_3 A_{\text{odd}}(x_3^2)
\end{align*}
\]

Observe that we evaluate both \( A_{\text{even}}(x) \) and \( A_{\text{odd}}(x) \) at \( x = x_0^2, x = x_1^2, x = x_2^2 \) and \( x = x_3^2 \).

But we decided to always evaluate polynomials of degree bound 4 at \( x = 1, x = i, x = -1 \) and \( x = -i \).

So, \( x_0^2 = 1 \Rightarrow x_0 = 1, x_1^2 = i \Rightarrow x_1 = \frac{1+i}{\sqrt{2}}, x_2^2 = -1 \Rightarrow x_2 = i, \) and \( x_3^2 = -i \Rightarrow x_3 = -\frac{1+i}{\sqrt{2}} \).
Given a Polynomial of Degree Bound 8
Find 8 Distinct Points to Efficiently Evaluate it at

\[ A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \]

\[ A_{\text{even}}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3 \]
\[ A_{\text{odd}}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3 \]

\[
\begin{align*}
A(x_0) &= A_{\text{even}}(x_0^2) + x_0 A_{\text{odd}}(x_0^2) \\
A(x_1) &= A_{\text{even}}(x_1^2) + x_1 A_{\text{odd}}(x_1^2) \\
A(x_2) &= A_{\text{even}}(x_2^2) + x_2 A_{\text{odd}}(x_2^2) \\
A(x_3) &= A_{\text{even}}(x_3^2) + x_3 A_{\text{odd}}(x_3^2) \\
A(x_4) &= A_{\text{even}}(x_0^2) - x_0 A_{\text{odd}}(x_0^2) \\
A(x_5) &= A_{\text{even}}(x_1^2) - x_1 A_{\text{odd}}(x_1^2) \\
A(x_6) &= A_{\text{even}}(x_2^2) - x_2 A_{\text{odd}}(x_2^2) \\
A(x_7) &= A_{\text{even}}(x_3^2) - x_3 A_{\text{odd}}(x_3^2)
\end{align*}
\]

So, we evaluate any polynomial of degree bound 8 at

\[
x_0 = 1, \ x_1 = \frac{1+i}{\sqrt{2}}, \ x_2 = i, \ x_3 = \frac{-1+i}{\sqrt{2}}
\]
and

\[
x_4 = -x_0 = -1, \ x_5 = -x_1 = -\frac{1+i}{\sqrt{2}}, \ x_6 = -x_2 = -i, \ x_7 = -x_3 = -\frac{-1+i}{\sqrt{2}}
\]
### Given a Polynomial of Degree Bound \( n = 2^k \)

#### Find \( n = 2^k \) Distinct Points to Efficiently Evaluate it at

<table>
<thead>
<tr>
<th>Degree Bound</th>
<th>How Did We Find the Points to Evaluate the Polynomial at?</th>
<th>The Points</th>
<th>Point Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 2^1 )</td>
<td>... ... ...</td>
<td>( 1, ) ( -1 )</td>
<td>all 2nd roots of unity</td>
</tr>
<tr>
<td>( 2^2 )</td>
<td>take positive and negative square roots of points used for degree bound ( 2^1 ) which are already the 2nd roots of unity</td>
<td>( 1, ) ( i, ) ( -1, ) ( -i )</td>
<td>all 4th roots of unity</td>
</tr>
<tr>
<td>( 2^3 )</td>
<td>take positive and negative square roots of points used for degree bound ( 2^2 ) which are already the 4th roots of unity</td>
<td>( 1, ) ( \frac{1+i}{\sqrt{2}}, ) ( i, ) ( \frac{-1+i}{\sqrt{2}}, ) ( -1, ) ( -\frac{1+i}{\sqrt{2}}, ) ( -i, ) ( -\frac{-1+i}{\sqrt{2}} )</td>
<td>all 8th roots of unity</td>
</tr>
<tr>
<td>( 2^4 )</td>
<td>take positive and negative square roots of points used for degree bound ( 2^3 ) which are already the 8th roots of unity</td>
<td>( 1, ) ( \frac{\sqrt{2}+\sqrt{2}}{2}+i\frac{\sqrt{2}-\sqrt{2}}{2}, ) ( ... ) ( ... ) ( -1, ) ( -\frac{\sqrt{2}+\sqrt{2}}{2}+i\frac{\sqrt{2}-\sqrt{2}}{2}, ) ( ... ) ( ... )</td>
<td>all 16th roots of unity</td>
</tr>
<tr>
<td>( ... )</td>
<td>... ... ...</td>
<td>... ...</td>
<td>... ... ...</td>
</tr>
<tr>
<td>( 2^{k-1} )</td>
<td>take positive and negative square roots of points used for degree bound ( 2^{k-2} ) which are already the ( 2^{k-2} )th roots of unity</td>
<td>... ...</td>
<td>all ( 2^{k-1} )th roots of unity</td>
</tr>
<tr>
<td>( n = 2^k )</td>
<td>take positive and negative square roots of points used for degree bound ( 2^{k-1} ) which are already the ( 2^{k-1} )th roots of unity</td>
<td>... ...</td>
<td>all ( 2^k )th roots of unity (i.e., ( n )th roots of unity)</td>
</tr>
</tbody>
</table>
How to Find all $n^{th}$ Roots of Unity

The $n^{th}$ roots of unity are: $1, \omega_n, (\omega_n)^2, (\omega_n)^3, \ldots, (\omega_n)^{n-1}$, where $\omega_n = \cos \frac{2\pi}{n} + i \sin \frac{2\pi}{n} = e^{\frac{2\pi i}{n}}$ is known as the primitive $n^{th}$ roots of unity.

The result above can be derived using Euler’s Formula.

Euler’s Formula: For any real number $\alpha$, $\cos \alpha + i \sin \alpha = e^{i\alpha}$

Euler’s formula follows very easily from the following three power series each of which holds for $-\infty < \alpha < +\infty$:

\[
\cos \alpha = 1 - \frac{\alpha^2}{2!} + \frac{\alpha^4}{4!} - \frac{\alpha^6}{6!} + \frac{\alpha^8}{8!} - \ldots
\]

\[
\sin \alpha = \alpha - \frac{\alpha^3}{3!} + \frac{\alpha^5}{5!} - \frac{\alpha^7}{7!} + \frac{\alpha^9}{9!} - \ldots
\]

\[
e^{\alpha} = 1 + \alpha + \frac{\alpha^2}{2!} + \frac{\alpha^3}{3!} + \frac{\alpha^4}{4!} + \frac{\alpha^5}{5!} + \frac{\alpha^6}{6!} + \frac{\alpha^7}{7!} + \frac{\alpha^8}{8!} + \ldots
\]
How to Find all $n^{\text{th}}$ Roots of Unity

Observe that for (any) real numbers $\alpha$ and $p$,
\[
(cos \alpha + i \sin \alpha)^p = (e^{i\alpha})^p = e^{i(p\alpha)} = \cos(p\alpha) + i \sin(p\alpha)
\]

Also observe that for any integer $k$, $\cos(k \times 2\Pi) + i \sin(k \times 2\Pi) = 1 + i \times 0 = 1$

Then the $n^{\text{th}}$ root of 1 (unity) is:
\[
1^{\frac{1}{n}} = (\cos(k \times 2\Pi) + i \sin(k \times 2\Pi))^{\frac{1}{n}} = \cos \left( k \times \frac{2\Pi}{n} \right) + i \sin \left( k \times \frac{2\Pi}{n} \right)
\]

Observe that $\cos \left( k \times \frac{2\Pi}{n} \right) + i \sin \left( k \times \frac{2\Pi}{n} \right)$ takes $n$ distinct values for $0 \leq k < n$, and then simply repeats those values for $k < 0$ and $k \geq n$.

When $k = 1$, we have:
\[
\cos \left( k \times \frac{2\Pi}{n} \right) + i \sin \left( k \times \frac{2\Pi}{n} \right) = \cos \left( \frac{2\Pi}{n} \right) + i \sin \left( \frac{2\Pi}{n} \right) = \omega_n = \text{primitive } n^{\text{th}} \text{ root of 1.}
\]

Clearly, for any $k$, $\cos \left( k \times \frac{2\Pi}{n} \right) + i \sin \left( k \times \frac{2\Pi}{n} \right) = \left( \cos \left( \frac{2\Pi}{n} \right) + i \sin \left( \frac{2\Pi}{n} \right) \right)^k = (\omega_n)^k$

Hence, $1^{\frac{1}{n}} = \cos \left( k \times \frac{2\Pi}{n} \right) + i \sin \left( k \times \frac{2\Pi}{n} \right) = (\omega_n)^k$, for $k = 0, 1, 2, \ldots, n - 1$.

In other words, the $n^{\text{th}}$ roots of 1 (unity) are: $1, \omega_n, (\omega_n)^2, (\omega_n)^3, \ldots \ldots \ldots, (\omega_n)^{n-1}$
Coefficient Form ⇒ Point-Value Form

For a polynomial of degree bound $n = 2^k$, we need to apply the trick recursively at most $\log n = k$ times.

We choose $x_0 = 1 = \omega_n^0$ and set $x_j = \omega_n^j$ for $1 \leq j \leq n - 1$.

Then we compute the following product:

$$\begin{bmatrix}
y_0 \\
y_1 \\
y_2 \\
\vdots \\
y_{n-1}
\end{bmatrix} = \begin{bmatrix}
A(1) \\
A(\omega_n) \\
A(\omega_n^2) \\
\vdots \\
A(\omega_n^{n-1})
\end{bmatrix} = \begin{bmatrix}
1 & 1 & 1 & \cdots & 1 \\
1 & \omega_n & (\omega_n)^2 & \cdots & (\omega_n)^{n-1} \\
1 & \omega_n^2 & (\omega_n^2)^2 & \cdots & (\omega_n^2)^{n-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \omega_n^{n-1} & (\omega_n^{n-1})^2 & \cdots & (\omega_n^{n-1})^{n-1}
\end{bmatrix} \begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_{n-1}
\end{bmatrix}$$

The vector $y = (y_0, y_1, \cdots, y_{n-1})$ is called the discrete Fourier transform (DFT) of $(a_0, a_1, \cdots, a_{n-1})$.

This method of computing DFT is called the fast Fourier transform (FFT) method.
Coefficient Form $\Rightarrow$ Point-Value Form

Example: For $n = 2^3 = 8$:

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7$$

We need to evaluate $A(x)$ at $x = \omega^i_8$ for $0 \leq i < 8$.

Now $A(x) = A_{even}(x^2) + x \cdot A_{odd}(x^2)$,

where $A_{even}(y) = a_0 + a_2 y + a_4 y^2 + a_6 y^3$

and $A_{odd}(y) = a_1 + a_3 y + a_5 y^2 + a_7 y^3$
Coefficient Form ⇒ Point-Value Form

Observe that:

\[
\begin{align*}
\omega_8^0 &= \omega_8^8 = \omega_4^0 \\
\omega_8^2 &= \omega_8^{10} = \omega_4^1 \\
\omega_8^4 &= \omega_8^{12} = \omega_4^2 \\
\omega_8^6 &= \omega_8^{14} = \omega_4^3
\end{align*}
\]

Also:

\[
\begin{align*}
\omega_8^4 &= -\omega_8^0 \\
\omega_8^5 &= -\omega_8^1 \\
\omega_8^6 &= -\omega_8^2 \\
\omega_8^7 &= -\omega_8^3
\end{align*}
\]

\[
\begin{align*}
A(\omega_8^0) &= A_{even}(\omega_8^0) + \omega_8^0 \cdot A_{odd}(\omega_8^0) = A_{even}(\omega_4^0) + \omega_8^0 \cdot A_{odd}(\omega_4^0) \\
A(\omega_8^1) &= A_{even}(\omega_8^2) + \omega_8^1 \cdot A_{odd}(\omega_8^2) = A_{even}(\omega_4^1) + \omega_8^1 \cdot A_{odd}(\omega_4^1) \\
A(\omega_8^2) &= A_{even}(\omega_8^4) + \omega_8^2 \cdot A_{odd}(\omega_8^4) = A_{even}(\omega_4^2) + \omega_8^2 \cdot A_{odd}(\omega_4^2) \\
A(\omega_8^3) &= A_{even}(\omega_8^6) + \omega_8^3 \cdot A_{odd}(\omega_8^6) = A_{even}(\omega_4^3) + \omega_8^3 \cdot A_{odd}(\omega_4^3) \\
A(\omega_8^4) &= A_{even}(\omega_8^8) + \omega_8^4 \cdot A_{odd}(\omega_8^8) = A_{even}(\omega_4^0) - \omega_8^0 \cdot A_{odd}(\omega_4^0) \\
A(\omega_8^5) &= A_{even}(\omega_8^{10}) + \omega_8^5 \cdot A_{odd}(\omega_8^{10}) = A_{even}(\omega_4^1) - \omega_8^1 \cdot A_{odd}(\omega_4^1) \\
A(\omega_8^6) &= A_{even}(\omega_8^{12}) + \omega_8^6 \cdot A_{odd}(\omega_8^{12}) = A_{even}(\omega_4^2) - \omega_8^2 \cdot A_{odd}(\omega_4^2) \\
A(\omega_8^7) &= A_{even}(\omega_8^{14}) + \omega_8^7 \cdot A_{odd}(\omega_8^{14}) = A_{even}(\omega_4^3) - \omega_8^3 \cdot A_{odd}(\omega_4^3)
\end{align*}
\]
Coefficient Form $\Rightarrow$ Point-Value Form

\[ \text{Rec-FFT} \left( \left( a_0, a_1, \ldots, a_{n-1} \right) \right) \quad \{ n = 2^k \text{ for integer } k \geq 0 \} \]

1. \text{if } n = 1 \text{ then}

2. \text{return } (a_0)

3. $\omega_n \leftarrow e^{2\pi i/n}$

4. $\omega \leftarrow 1$

5. $y^{\text{even}} \leftarrow \text{Rec-FFT} \left( \left( a_0, a_2, \ldots, a_{n-2} \right) \right)$

6. $y^{\text{odd}} \leftarrow \text{Rec-FFT} \left( \left( a_1, a_3, \ldots, a_{n-1} \right) \right)$

7. \text{for } j \leftarrow 0 \text{ to } n/2 - 1 \text{ do}

8. $y_j \leftarrow y^{\text{even}}_j + \omega y^{\text{odd}}_j$

9. $y_{n/2+j} \leftarrow y^{\text{even}}_j - \omega y^{\text{odd}}_j$

10. $\omega \leftarrow \omega^{\omega}$

11. \text{return } y

Running time:

\[ T(n) = \begin{cases} \Theta(1), & \text{if } n = 1, \\ 2T \left( \frac{n}{2} \right) + \Theta(n), & \text{otherwise}. \end{cases} \]

\[ = \Theta(n \log n) \]
Faster Polynomial Multiplication? (in Coefficient Form)

\[ A(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1} \]
\[ B(x) = b_0 + b_1x + \cdots + b_{n-1}x^{n-1} \]

Ordinary multiplication: Time \( \Theta(n^2) \)

\[ C(x) = c_0 + c_1x + \cdots + c_{2n-1}x^{2n-1} \]

- Ordinary multiplication
- Forward FFT: Time \( \Theta(n \log n) \)
- Pointwise multiplication: Time \( \Theta(n) \)
- Interpolation: Time?
Point-Value Form $\Rightarrow$ Coefficient Form

Given:

\[
\begin{bmatrix}
1 & 1 & 1 & \cdots & 1 \\
1 & \omega_n & (\omega_n)^2 & \cdots & (\omega_n)^{n-1} \\
1 & \omega_n^2 & (\omega_n^2)^2 & \cdots & (\omega_n^2)^{n-1} \\
\vdots & \vdots & \vdots & \cdots & \vdots \\
1 & \omega_n^{n-1} & (\omega_n^{n-1})^2 & \cdots & (\omega_n^{n-1})^{n-1}
\end{bmatrix}
\]

\[V(\omega_n)\]

Vandermonde Matrix

\[\Rightarrow V(\omega_n) \cdot \bar{a} = \bar{y}\]

We want to solve: \[\bar{a} = [V(\omega_n)]^{-1} \cdot \bar{y}\]

It turns out that: \[[V(\omega_n)]^{-1} = \frac{1}{n} V \left( \frac{1}{\omega_n} \right)\]

That means $[V(\omega_n)]^{-1}$ looks almost similar to $V(\omega_n)$!
Point-Value Form ⇒ Coefficient Form

Show that: \[ [V(\omega_n)]^{-1} = \frac{1}{n} V \left( \frac{1}{\omega_n} \right) \]

Let \[ U(\omega_n) = \frac{1}{n} V \left( \frac{1}{\omega_n} \right) \]

We want to show that \[ U(\omega_n)V(\omega_n) = I_n, \]
where \( I_n \) is the \( n \times n \) identity matrix.

Observe that for \( 0 \leq j, k \leq n-1 \), the \( (j, k)^{th} \) entries are:

\[ [V(\omega_n)]_{jk} = \omega_n^{jk} \quad \text{and} \quad [U(\omega_n)]_{jk} = \frac{1}{n} \omega_n^{-jk} \]

Then entry \( (p, q) \) of \( U(\omega_n)V(\omega_n) \),

\[ [U(\omega_n)V(\omega_n)]_{pq} = \sum_{k=0}^{n-1} [U(\omega_n)]_{pk} [V(\omega_n)]_{kq} = \frac{1}{n} \sum_{k=0}^{n-1} \omega_n^{k(q-p)} \]
Point-Value Form $\Rightarrow$ Coefficient Form

$$[U(\omega_n)V(\omega_n)]_{pq} = \frac{1}{n} \sum_{k=0}^{n-1} \omega_n^k(q-p)$$

CASE $p = q$:

$$[U(\omega_n)V(\omega_n)]_{pq} = \frac{1}{n} \sum_{k=0}^{n-1} \omega_n^0 = \frac{1}{n} \sum_{k=0}^{n-1} 1 = \frac{1}{n} \times n = 1$$

CASE $p \neq q$:

$$[U(\omega_n)V(\omega_n)]_{pq} = \frac{1}{n} \sum_{k=0}^{n-1} (\omega_n^{q-p})^k = \frac{1}{n} \times \left(\frac{\omega_n^{q-p}}{\omega_n^{q-p} - 1}\right)^n - 1$$

$$= \frac{1}{n} \times \frac{(\omega_n^{n})^{q-p} - 1}{\omega_n^{q-p} - 1} = \frac{1}{n} \times \frac{(1)^{q-p} - 1}{\omega_n^{q-p} - 1} = 0$$

Hence $U(\omega_n)V(\omega_n) = I_n$
We need to compute the following matrix-vector product:

\[
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
\vdots \\
a_{n-1}
\end{bmatrix}
= \frac{1}{n} \times
\begin{bmatrix}
1 & 1 & 1 & \cdots & 1 \\
1 & \frac{1}{\omega_n} & \left(\frac{1}{\omega_n}\right)^2 & \cdots & \left(\frac{1}{\omega_n}\right)^{n-1} \\
1 & \frac{1}{\omega_n^2} & \left(\frac{1}{\omega_n^2}\right)^2 & \cdots & \left(\frac{1}{\omega_n^2}\right)^{n-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & \frac{1}{\omega_n^{n-1}} & \left(\frac{1}{\omega_n^{n-1}}\right)^2 & \cdots & \left(\frac{1}{\omega_n^{n-1}}\right)^{n-1}
\end{bmatrix}
\begin{bmatrix}
y_0 \\
y_1 \\
y_2 \\
\vdots \\
y_{n-1}
\end{bmatrix}
\begin{bmatrix}
\omega_0^{-1} \\
\omega_1 \\
\omega_2 \\
\vdots \\
\omega_{n-1}
\end{bmatrix}
\]

This inverse problem is almost similar to the forward problem, and can be solved in \(\Theta(n \log n)\) time using the same algorithm as the forward FFT with only minor modifications!
Faster Polynomial Multiplication? (in Coefficient Form)

Two polynomials of degree bound $n$ given in the coefficient form can be multiplied in $\Theta(n \log n)$ time!
Some Applications of Fourier Transform and FFT

- Signal processing
- Image processing
- Noise reduction
- Data compression
- Solving partial differential equation
- Multiplication of large integers
- Polynomial multiplication
- Molecular docking
Some Applications of Fourier Transform and FFT

Any periodic signal can be represented as a sum of a series of sinusoidal (sine & cosine) waves. [1807]
Spatial (Time) Domain ↔ Frequency Domain

Source: The Scientist and Engineer’s Guide to Digital Signal Processing by Steven W. Smith
Spatial (Time) Domain ⇔ Frequency Domain

Function $s(x)$ (in red) is a sum of six sine functions of different amplitudes and harmonically related frequencies. The Fourier transform, $S(f)$ (in blue), which depicts amplitude vs frequency, reveals the 6 frequencies and their amplitudes.

Function $s(x)$ (in red) is a sum of six sine functions of different amplitudes and harmonically related frequencies. The Fourier transform, $S(f)$ (in blue), which depicts amplitude vs frequency, reveals the 6 frequencies and their amplitudes.

Function $s(x)$ (in red) is a sum of six sine functions of different amplitudes and harmonically related frequencies. The Fourier transform, $S(f)$ (in blue), which depicts amplitude vs frequency, reveals the 6 frequencies and their amplitudes.

Spatial (Time) Domain ⇔ Frequency Domain

Function $s(x)$ (in red) is a sum of six sine functions of different amplitudes and harmonically related frequencies. The Fourier transform, $S(f)$ (in blue), which depicts amplitude vs frequency, reveals the 6 frequencies and their amplitudes.

Function $s(x)$ (in red) is a sum of six sine functions of different amplitudes and harmonically related frequencies. The Fourier transform, $S(f)$ (in blue), which depicts amplitude vs frequency, reveals the 6 frequencies and their amplitudes.

Spatial (Time) Domain ↔ Frequency Domain (Fourier Transforms)

Let $s(t)$ be a signal specified in the time domain.

The strength of $s(t)$ at frequency $f$ is given by:

$$S(f) = \int_{-\infty}^{\infty} s(t) \cdot e^{-2\pi ift} \, dt$$

Evaluating this integral for all values of $f$ gives the frequency domain function.

Now $s(t)$ can be retrieved by summing up the signal strengths at all possible frequencies:

$$s(t) = \int_{-\infty}^{\infty} S(f) \cdot e^{2\pi ift} \, df$$
Why do the Transforms Work?

Let’s try to get a little intuition behind why the transforms work. We will look at a very simple example.

Suppose: \( s(t) = \cos(2\pi h \cdot t) \)

\[
\frac{1}{T} \int_{-T}^{T} s(t) \cdot e^{-2\pi if t} \ dt = \begin{cases} 
1 + \frac{\sin(4\pi fT)}{4\pi fT}, & \text{if } f = h, \\
\frac{\sin(2\pi (h-f)T)}{2\pi (h-f)T} + \frac{\sin(2\pi (h+f)T)}{2\pi (h+f)T}, & \text{otherwise}. 
\end{cases}
\]

\[
\Rightarrow \lim_{T \to \infty} \left( \frac{1}{T} \int_{-T}^{T} s(t) \cdot e^{-2\pi if t} \ dt \right) = \begin{cases} 
1, & \text{if } f = h, \\
0, & \text{otherwise}. 
\end{cases}
\]

So, the transform can detect if \( f = h \)!
Noise Reduction

Data Compression

− Discrete Cosine Transforms (DCT) are used for lossy data compression (e.g., MP3, JPEG, MPEG)

− DCT is a Fourier-related transform similar to DFT (Discrete Fourier Transform) but uses only real data (uses cosine waves only instead of both cosine and sine waves)

− Forward DCT transforms data from spatial to frequency domain

− Each frequency component is represented using a fewer number of bits (i.e., truncated / quantized)

− Low amplitude high frequency components are also removed

− Inverse DCT then transforms the data back to spatial domain

− The resulting image compresses better
Data Compression

Transformation to frequency domain using cosine transforms work in the same way as the Fourier transform.

Suppose: \( s(t) = \cos(2\pi h \cdot t) \)

\[
\frac{1}{T} \int_{-T}^{T} s(t) \cdot \cos(2\pi ft) \, dt = \begin{cases} 
1 + \frac{\sin(4\pi fT)}{4\pi fT}, & \text{if } f = h, \\
\frac{\sin(2\pi (h - f)T)}{2\pi (h - f)T} + \frac{\sin(2\pi (h + f)T)}{2\pi (h + f)T}, & \text{otherwise}.
\end{cases}
\]

\[
\Rightarrow \lim_{T \to \infty} \left( \frac{1}{T} \int_{-T}^{T} s(t) \cdot \cos(2\pi ft) \, dt \right) = \begin{cases} 
1, & \text{if } f = h, \\
0, & \text{otherwise}.
\end{cases}
\]

So, this transform can also detect if \( f = h \).
Protein-Protein Docking

- Knowledge of complexes is used in
  - Drug design
  - Studying molecular assemblies
  - Structure function analysis
  - Protein interactions

- **Protein-Protein Docking**: Given two proteins, find the best relative transformation and conformations to obtain a stable complex.

- Docking is a hard problem
  - Search space is huge (6D for rigid proteins)
  - Protein flexibility adds to the difficulty
To maximize skin-skin overlaps and minimize core-core overlaps

- assign positive real weights to skin atoms
- assign positive imaginary weights to core atoms

Let \( A' \) denote molecule \( A \) with the pseudo skin atoms.

For \( P \in \{A', B\} \) with \( M_P \) atoms, affinity function: 
\[
f_P(x) = \sum_{k=1}^{M_P} w_k \cdot g_k(x)
\]

Here \( g_k(x) \) is a Gaussian representation of atom \( k \), and \( w_k \) its weight.
Let $A'$ denote molecule $A$ with the pseudo skin atoms.

For $P \in \{ A', B \}$ with $M_P$ atoms, affinity function:

$$f_P(x) = \sum_{k=1}^{M_P} w_k \cdot g_k(x)$$

For rotation $r$ and translation $t$ of molecule $B$ (i.e., $B_{t,r}$),

the interaction score, $F_{A,B}(t,r) = \int_x f_{A'}(x) f_{B_{t,r}}(x) \, dx$
For rotation $r$ and translation $t$ of molecule $B$ (i.e., $B_{t,r}$),

the interaction score, $F_{A,B}(t,r) = \int f_A'(x)f_{B_{t,r}}(x) \, dx$

$\text{Re} \left( F_{A,B}(t,r) \right) = \text{skin-skin overlap score} - \text{core-core overlap score}$

$\text{Im} \left( F_{A,B}(t,r) \right) = \text{skin-core overlap score}$
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Docking: Rotational & Translational Search
Translational Search using FFT

∀z ∈ Ω = [−n, n]^3, h(z) = \int_{x \in \Omega} f_{A'}(x)f_{B'}(z - x)dx
## Multiplication of Bivariate Polynomials

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$y$</th>
<th>$x^2$</th>
<th>$x^3$</th>
<th>$y^2$</th>
<th>$x^2y$</th>
<th>$xy^2$</th>
<th>$x^3y$</th>
<th>$y^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{1,3}x^3y^2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{2,3}x^2y^2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{1,2}x^3y$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{1,1}x^2y$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{3,0}x^3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{2,0}x^2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{1,0}x$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{0,0}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Translation $\langle i, j \rangle$

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$y$</th>
<th>$x^2$</th>
<th>$x^3$</th>
<th>$y^2$</th>
<th>$x^2y$</th>
<th>$xy^2$</th>
<th>$x^3y$</th>
<th>$y^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_{0,0}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{1,0}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{2,0}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{3,0}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{0,1}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{1,1}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{2,1}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{3,1}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{0,2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{1,2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{2,2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{3,2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{0,3}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{1,3}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{2,3}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$a_{3,3}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Coefficient of $x^{i-1}y^{j-1}$
**Multiplication of Bivariate Polynomials**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$b_{0,0} x^3$</td>
<td>$b_{0,1} x^2$</td>
<td>$b_{0,2} x$</td>
<td>$b_{0,3}$</td>
<td>$a_{0,0} x^3$</td>
<td>$a_{1,0} x^2$</td>
<td>$a_{2,0} x$</td>
<td>$a_{3,0}$</td>
</tr>
<tr>
<td>1</td>
<td>$b_{1,0} x^3$</td>
<td>$b_{1,1} x^2$</td>
<td>$b_{1,2} x$</td>
<td>$b_{1,3}$</td>
<td>$a_{0,1} x^3$</td>
<td>$a_{1,1} x^2$</td>
<td>$a_{2,1} x$</td>
<td>$a_{3,1}$</td>
</tr>
<tr>
<td>2</td>
<td>$b_{2,0} x^3$</td>
<td>$b_{2,1} x^2$</td>
<td>$b_{2,2} x$</td>
<td>$b_{2,3}$</td>
<td>$a_{0,2} x^3$</td>
<td>$a_{1,2} x^2$</td>
<td>$a_{2,2} x$</td>
<td>$a_{3,2}$</td>
</tr>
<tr>
<td>3</td>
<td>$b_{3,0} x^3$</td>
<td>$b_{3,1} x^2$</td>
<td>$b_{3,2} x$</td>
<td>$b_{3,3}$</td>
<td>$a_{0,3} x^3$</td>
<td>$a_{1,3} x^2$</td>
<td>$a_{2,3} x$</td>
<td>$a_{3,3}$</td>
</tr>
</tbody>
</table>

Translation $\langle i, j \rangle$

Translation $\langle 1, 1 \rangle$

Coefficient of $x^{i-1} y^{j-1}$

Coefficient of $x^i y^j$
## Multiplication of Bivariate Polynomials

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>( b_{i,j} x^i y^j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>( b_{0,0} )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>( b_{0,1} x y )</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>( b_{0,2} x^2 )</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>( b_{0,3} y^3 )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>( b_{1,0} x^3 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>( b_{1,1} x^2 y )</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>( b_{1,2} x y^2 )</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>( b_{1,3} y^3 )</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>( b_{2,0} x^3 )</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>( b_{2,1} x^2 y )</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>( b_{2,2} x y^2 )</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>( b_{2,3} y^3 )</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>( b_{3,0} x^3 )</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>( b_{3,1} x^2 y )</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>( b_{3,2} x y^2 )</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>( b_{3,3} y^3 )</td>
</tr>
</tbody>
</table>

### Translation \( \langle i, j \rangle \)

### Translation \( \langle 3,1 \rangle \)

### Coefficient of \( x^{i-1} y^{j-1} \)

### Coefficient of \( x^2 y^0 \)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Translation \( \langle 5,1 \rangle \)

Coefficient of \( x^{i-1}y^{j-1} \)

Coefficient of \( x^4y^0 \)
Multiplication of Bivariate Polynomials

Translation $\langle i, j \rangle$

Translation $\langle 5, 2 \rangle$

Coefficient of $x^{i-1}y^{j-1}$

Coefficient of $x^4y^1$
Multiplication of Bivariate Polynomials

Translation \langle i, j \rangle

Translation \langle 2, 2 \rangle

Coefficient of \( x^{i-1} y^{j-1} \)

Coefficient of \( x^i y^j \)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Translation \( \langle 3,3 \rangle \)

Coefficient of \( x^{i-1}y^{j-1} \)

Coefficient of \( x^2y^2 \)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Translation \( \langle 7,3 \rangle \)

Coefficient of \( x^{i-1} y^{j-1} \)

Coefficient of \( x^6 y^2 \)
Multiplication of Bivariate Polynomials

Translation \(i, j\)

Translation \(4,4\)

Coefficient of \(x^{i-1}y^{j-1}\)

Coefficient of \(x^3y^3\)

Multiplication of Bivariate Polynomials
Multiplication of Bivariate Polynomials

Translation \(\{i, j\}\)

Translation \(\{1, 4\}\)

Coefficient of \(x^{i-1}y^{j-1}\)

Coefficient of \(x^0y^3\)
Multiplication of Bivariate Polynomials

Translation \( i, j \)

Translation \( 1, 5 \)

Coefficient of \( x^{i-1} y^{j-1} \)

Coefficient of \( x^0 y^4 \)
Multiplication of Bivariate Polynomials

Translation \(\langle i, j \rangle\)

Translation \(\langle 7, 5 \rangle\)

Coefficient of \(x^{i-1}y^{j-1}\)

Coefficient of \(x^6y^4\)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Translation \( \langle 7,6 \rangle \)

Coefficient of \( x^{i-1} y^{j-1} \)

Coefficient of \( x^6 y^5 \)
## Multiplication of Bivariate Polynomials

![Diagram of multiplication of bivariate polynomials]

### Translation \( (i, j) \)

### Translation \( (3, 6) \)

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( a_{0,0} )</td>
<td>( a_{1,0}x )</td>
<td>( a_{2,0}x^2 )</td>
<td>( a_{3,0}x^3 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>( a_{0,1}y )</td>
<td>( a_{1,1}xy )</td>
<td>( a_{2,1}x^2y )</td>
<td>( a_{3,1}x^3y )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>( a_{0,2}y^2 )</td>
<td>( a_{1,2}xy^2 )</td>
<td>( a_{2,2}x^2y^2 )</td>
<td>( a_{3,2}x^3y^2 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( b_{1,1}x^3y^3 )</td>
<td>( b_{1,1}x^2y^3 )</td>
<td>( b_{1,1}xy^3 )</td>
<td>( b_{1,1}y^3 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( b_{1,2}x^3y^4 )</td>
<td>( b_{1,2}x^2y^4 )</td>
<td>( b_{1,2}xy^4 )</td>
<td>( b_{1,2}y^4 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>( b_{1,3}x^3y^5 )</td>
<td>( b_{1,3}x^2y^5 )</td>
<td>( b_{1,3}xy^5 )</td>
<td>( b_{1,3}y^5 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>( b_{3,0}x^3 )</td>
<td>( b_{3,0}x^2 )</td>
<td>( b_{1,0}x )</td>
<td>( b_{0,0} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Coefficient of \( x^{i-1}y^{j-1} \)

### Coefficient of \( x^2y^5 \)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Translation \( \langle 2, 7 \rangle \)

Coefficient of \( x^{i-1}y^{j-1} \)

Coefficient of \( x^iy^j \)
Multiplication of Bivariate Polynomials

Translation \( \langle i, j \rangle \)

Coefficient of \( x^{i-1} y^{j-1} \)

Translation \( \langle 7, 7 \rangle \)

Coefficient of \( x^6 y^6 \)