Advance Algorithm Analysis

(COMP 502 A)

Sharoon Nasim

sharoonnasim@fccollege.edu.pk

Office Hours:

Office: S-426 D

M,F 3 PM - 4 PM

TR 9:30 AM - 11:30 AM

Polynomial operations and representation

A polynomial $A(x)$ can be written in the following forms:

$$\begin{aligned} A(x) &= a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} \\ &= \sum_{k=0}^{n1} a_k x^k \\ &= \langle a_0, a_1, a_2, \ldots, a_{n-1} \rangle \quad \text{(coefficient vector)} \end{aligned}$$

The degree of A is $n - 1$.

Operations on Polynomials

- Evaluation
  - **Horner's Rule:** $A(x) = a_0 + x(a_1 + x(a_2 + \cdots x(a_{n-1}) \cdots))$. At each step, a sum is evaluated, then multiplied by x, before beginning the next step. Thus $O(n)$ multiplications and $O(n)$ additions are required.

- **Addition:** Given two polynomials $A(x)$ and $B(x)$, compute $C(x) = A(x) + B(x)$ $(\forall x)$. This takes $O(n)$ time using basic arithmetic, because $c_k = a_k + b_k$.

- Multiplication: Given two polynomials A(x) and B(x),

  compute C(x) = A(x)·B(x) (∀x).

The degree of the resulting polynomial is twice that of A or B. This multiplication is then equivalent to a convolution of the vectors A and reverse(B).

$$c_k = \sum_{j=0}^{k} a_j b_{k-j} \text{ for } 0 \leq k \leq 2(n-1)$$

Multiplication

Naive polynomial multiplication takes $O(n^2)$.

$A(x) = 6x^3 + 7x^2 - 10x + 9$ and $B(x) = -2x^3 + 4x - 5$ as follows:

$$
\begin{array}{rrrrrrr}
 & & & 6x^3 & +\ 7x^2 & -\ 10x & +\ 9 \\
 & & & -\ 2x^3 & & +\ 4x & -\ 5 \\
\hline
 & & & -\ 30x^3 & -\ 35x^2 & +\ 50x & -\ 45 \\
 & & 24x^4 & +\ 28x^3 & -\ 40x^2 & +\ 36x & \\
-\ 12x^6 & -\ 14x^5 & +\ 20x^4 & -\ 18x^3 & & & \\
\hline
-\ 12x^6 & -\ 14x^5 & +\ 44x^4 & -\ 20x^3 & -\ 75x^2 & +\ 86x & -\ 45
\end{array}
$$

Multiplication (divide-and-conquer approach)

The *Karatsuba algorithm* is a divide-and-conquer approach for fast multiplication of two polynomials (or integers). It significantly improves the naive $O(n^2)$ complexity to about $O(n^{\log_2 3}) \approx O(n^{1.585})$. Here's how it works:

$$A(x) = 3x^1 + 2x^0$$

$$B(x) = 5x^1 + 4x^0$$

$A_{\text{high}} = 3$ and $A_{\text{low}} = 2$

$B_{\text{high}} = 5$ and $B_{\text{low}} = 4$

$$P(x) = Z_2 \cdot x^2 + Z_1 \cdot x^1 + Z_0$$

$$P(x) = 15x^2 + 22x + 8$$

Z_0: Multiply the low parts:

$$Z_0 = A_{\text{low}} \cdot B_{\text{low}} = 2 \cdot 4 = 8$$

Z_2: Multiply the high parts:

$$Z_2 = A_{\text{high}} \cdot B_{\text{high}} = 3 \cdot 5 = 15$$

Z_1: Multiply the sum of the parts:

$$Z_1 = (A_{\text{high}} + A_{\text{low}}) \cdot (B_{\text{high}} + B_{\text{low}})$$

Calculating the sums:

$$(3 + 2) \cdot (5 + 4) = 5 \cdot 9 = 45$$

Now, subtract Z_0 and Z_2 from this result:

$$Z_1 = 45 - 8 - 15 = 22$$

Today, we will compute the product in

O(n.log(n))

time via Fast Fourier Transform!

Representation of polynomials

- Coefficient Representation
  - coefficient vectors a = $(a_0, a_1, \ldots, a_{n-1})$ and $b = (b_0, b_1, \ldots, b_{n-1})$
- Roots and a scale term • A(x)=(x − r0) · (x − r1) ····· (x − rn−1) · c • However, it is impossible to find exact roots with only basic arithmetic operations and kth root operations.
- Point Value Representation (Samples)

  (x0, y0), (x1, y1), ..., (xn−1, yn−1) with A(xi) = yi (∀i)

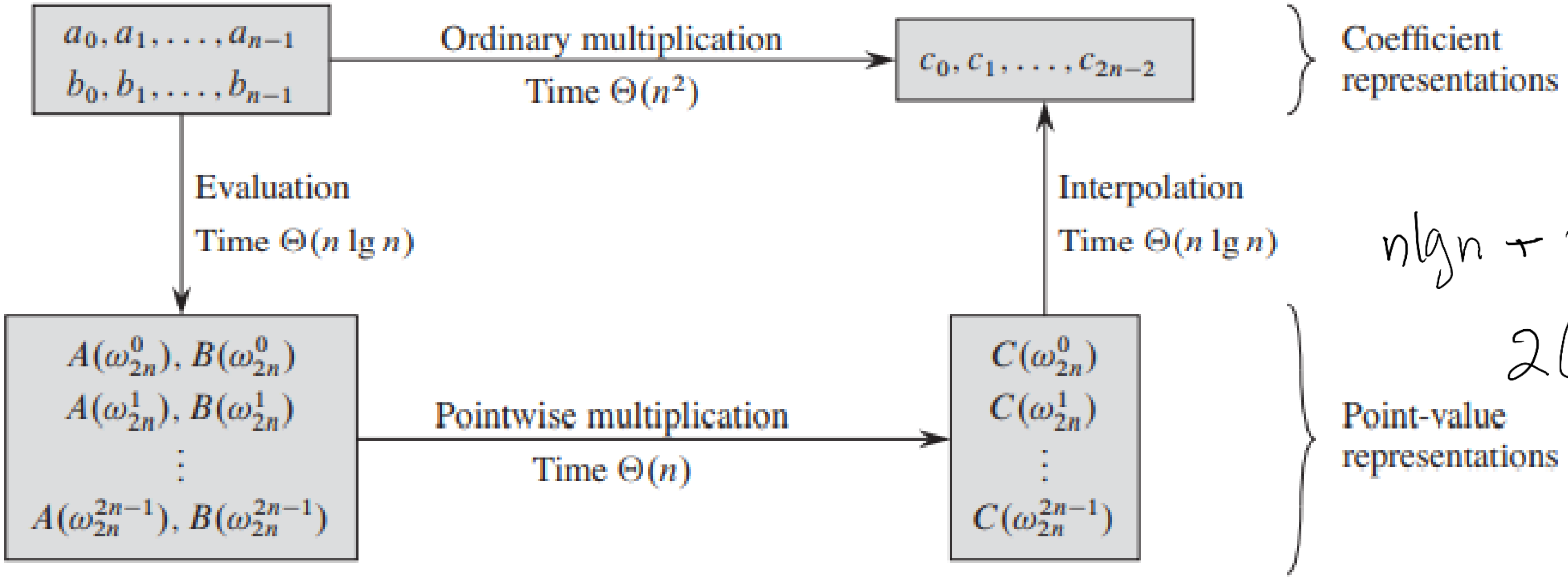
$\underline{2x^2 + 4x + 1} =$

$f(1) = 2+4+1 = 7$

$(1,7)(2,17)$

Representation	Evaluation	Addition	Multiplication
Coefficient	$O(n)$	$O(n)$	$O(n^2)$
Point-Value (Samples)	$O(n^2)$	$O(n)$	$O(n)$

Big Idea



$n\lg n + n + n\lg n$

$2(n\lg n) + n$

$O(n\lg n)$

Representation	Evaluation	Addition	Multiplication
Coefficient	$O(n)$	$O(n)$	$O(n^2)$
Point-Value (Samples)	$O(n^2)$	$O(n)$	$O(n)$

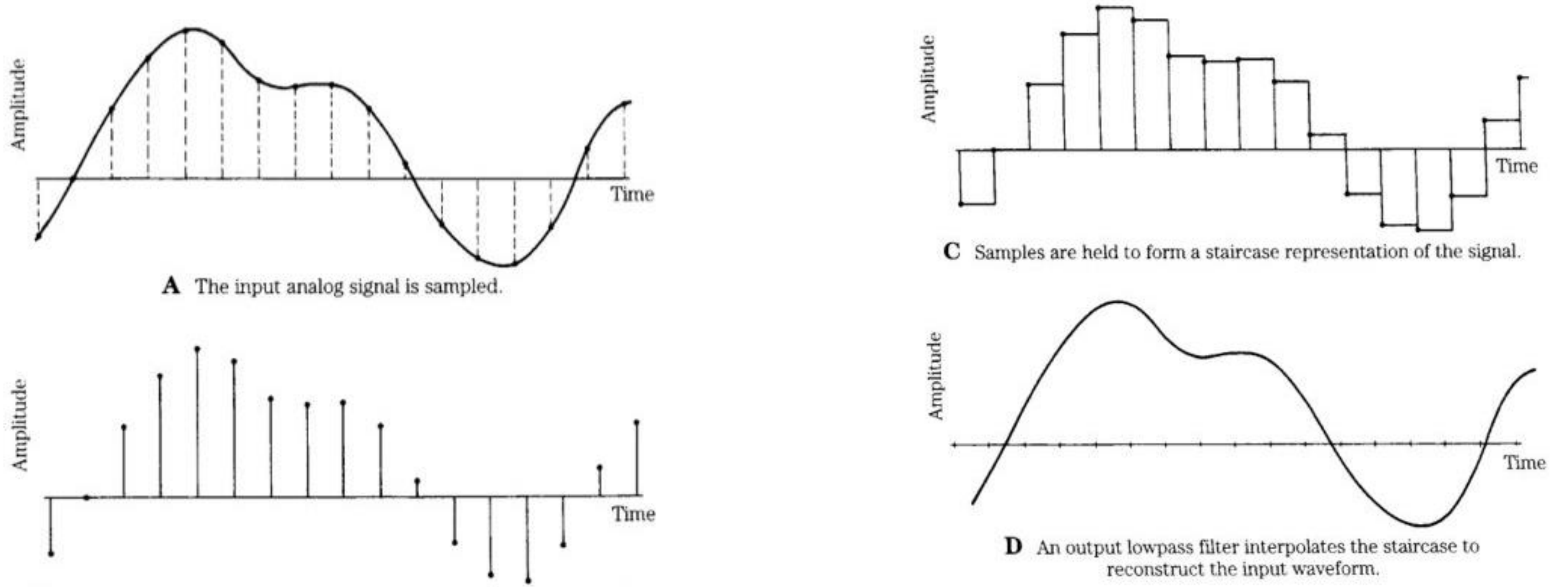
Fast Polynomial Multiplication

In order to compute the product of two polynomials A and B, we can perform the following steps.

1. Compute $A^* = FFT(A)$ and $B^* = FFT(B)$, which converts both A and B from coefficient vectors to a sample representation.
2. Compute $C^* = A^* \cdot B^*$ in sample representation in linear time by calculating $C_k^* = A_k^* \cdot B_k^*$ $(\forall k)$.
3. Compute $C = \mathrm{IFFT}(C^*)$, which is a vector representation of our final solution.

FFT (Fast Fourier Transform) and IFFT (Inverse Fast Fourier Transform) perform the conversion between Coefficient to Samples and Samples to coefficient representation respectively.

Continuous and Discrete Signals



A The input analog signal is sampled.

B The numerical values of these samples are stored or transmitted (effect of quantization not shown).

C Samples are held to form a staircase representation of the signal.

D An output lowpass filter interpolates the staircase to reconstruct the input waveform.

2-1 With discrete time sampling, a bandlimited signal can be sampled and reconstructed without loss because of sampling.

Fast Fourier Transform & Inverse Fast Fourier Transform

- FFT $x(n) \rightarrow X(k)$
- IFFT $X(k) \rightarrow x(n)$
- DIT > Decimation in Time n
- DIF > Decimation in Frequency k

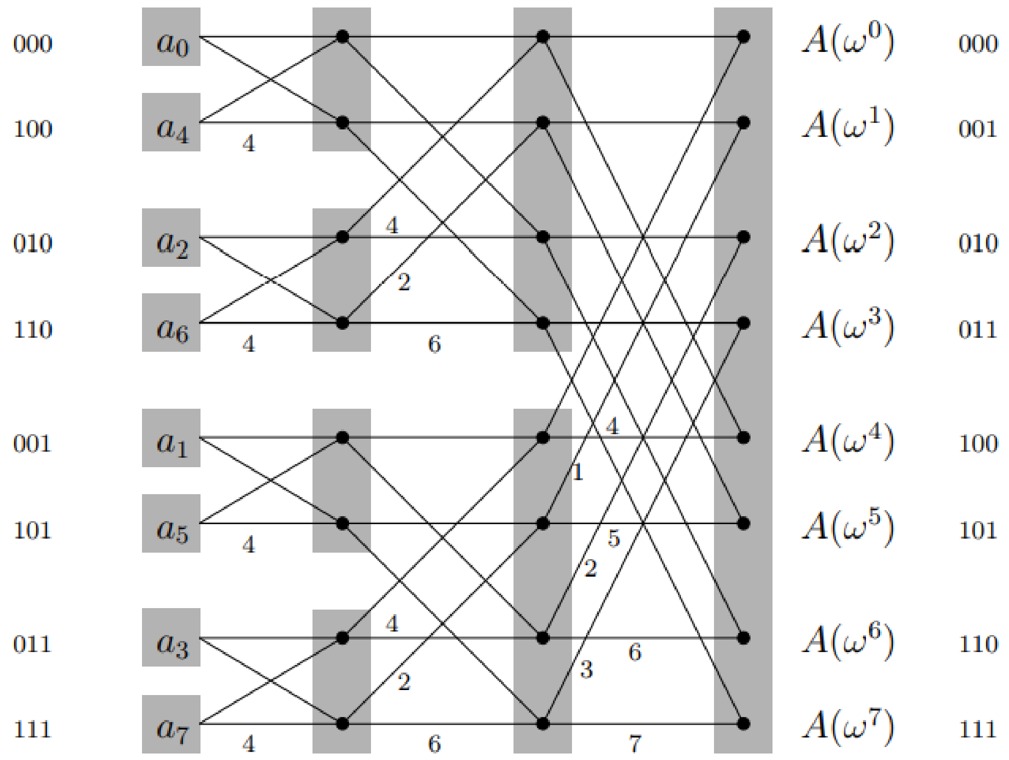
$$X(k) = \sum_{n=0}^{N-1} x(n).W_N^{nk} \qquad x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k).W_N^{-nk}$$

N=4 (0,1,2,3) **even:** 0,2 **odd:** 1,3

0,2,4,6

N=8 (0,1,2,3,4,5,6,7) **Even:** 0,4,2,6 **Odd:** 1,5,3,7

1,3,5,7



Twiddle Factor

$$W_N^L = e^{\frac{-j2\pi .L}{N}}$$

$W_4^0 = e^{\frac{-j2\pi 0}{4}} = e^0 = 1$

$W_8^0 = e^{\frac{-j2\pi \cdot 0}{8}}$

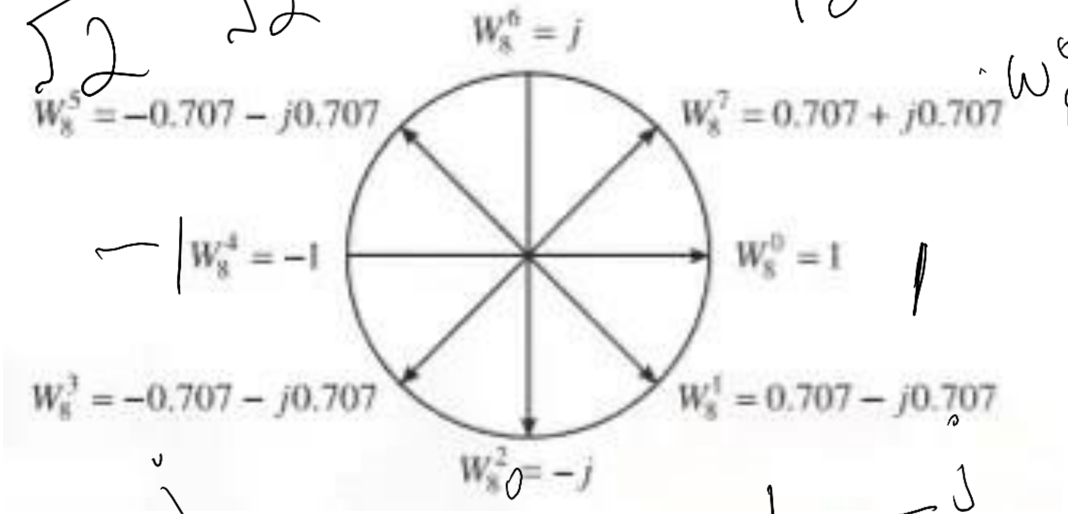
$W_4^1 = e^{\frac{-j2\pi(1)}{4_2}}$

$\cos\theta - j\sin\frac{\pi}{2}$

$W_8^0 = 1$

$e^{i\theta} = \cos\theta + i\sin\theta$

$-\frac{1}{\sqrt{2}} + \frac{j}{\sqrt{2}}$ $\quad j \quad$ $\frac{1}{\sqrt{2}} + \frac{j}{\sqrt{2}}$



$-1 \quad 1$

$-\frac{1}{\sqrt{2}} - \frac{j}{\sqrt{2}}$ $\quad -j \quad$ $\frac{1}{\sqrt{2}} - \frac{j}{\sqrt{2}}$

$W_8^7 =$

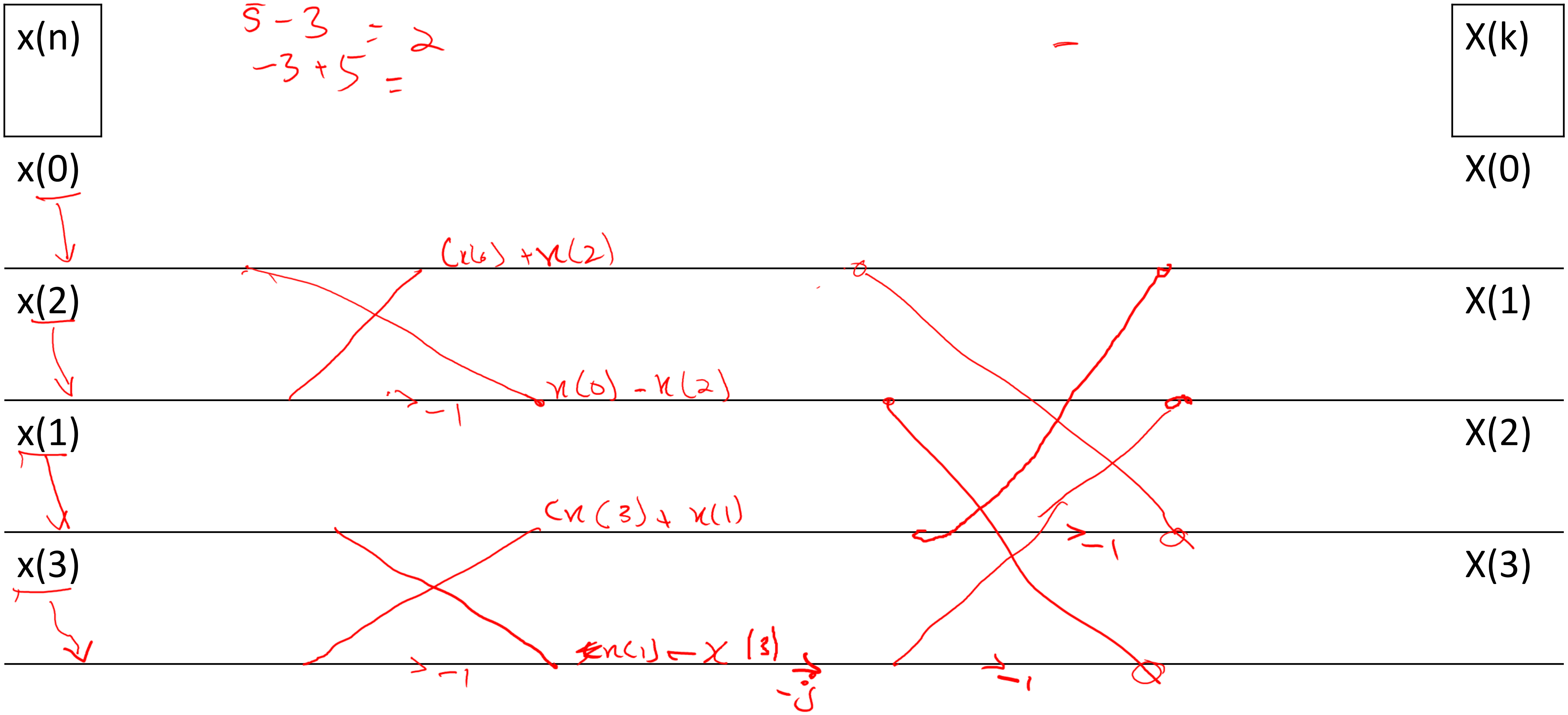
$W_8^0 = j$

$8\overline{)70}$
64
6

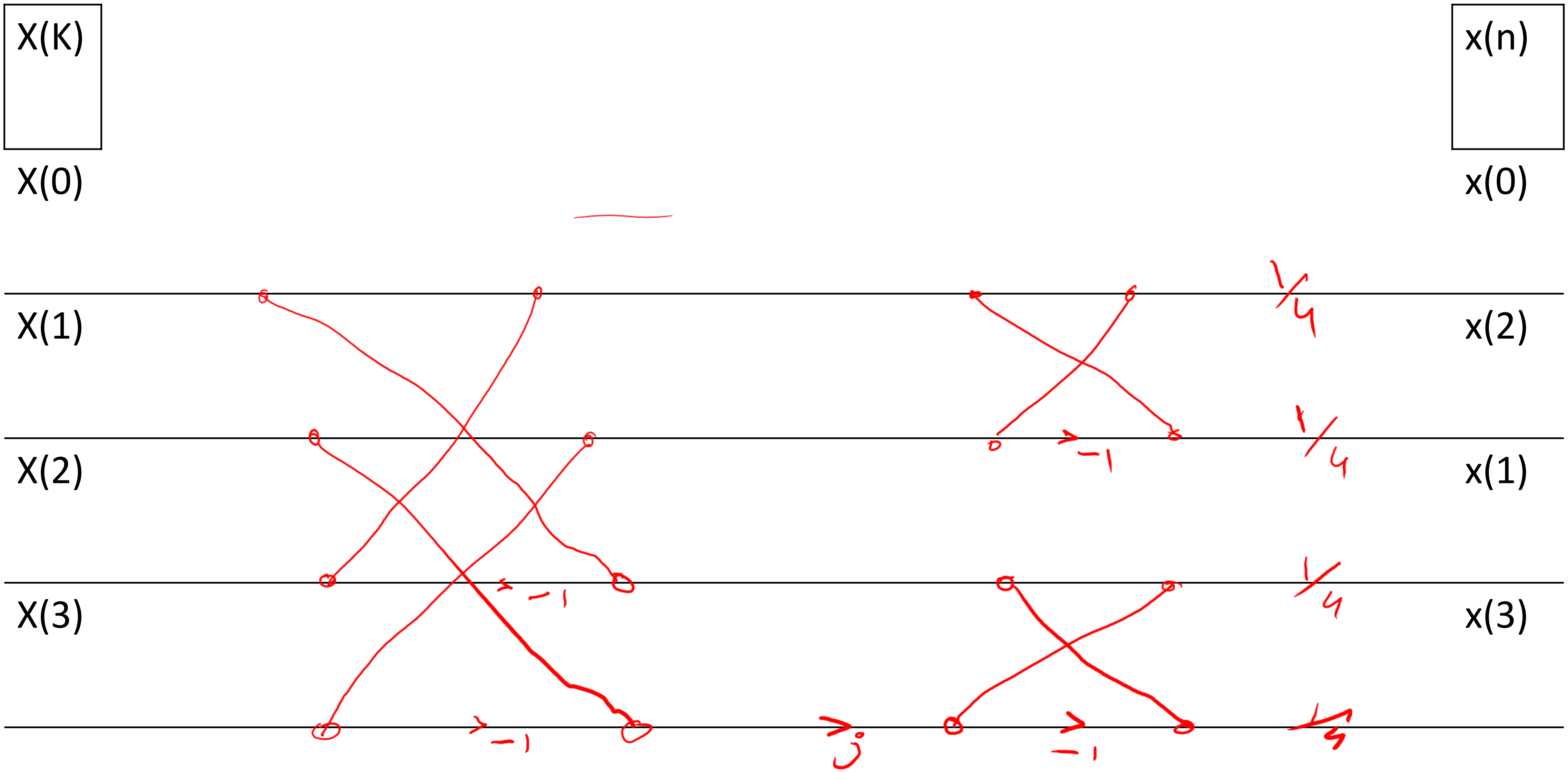
$$e^{i\theta} = cos\theta + isin\theta$$

$$e^{-i\theta} = cos\theta - isin\theta$$

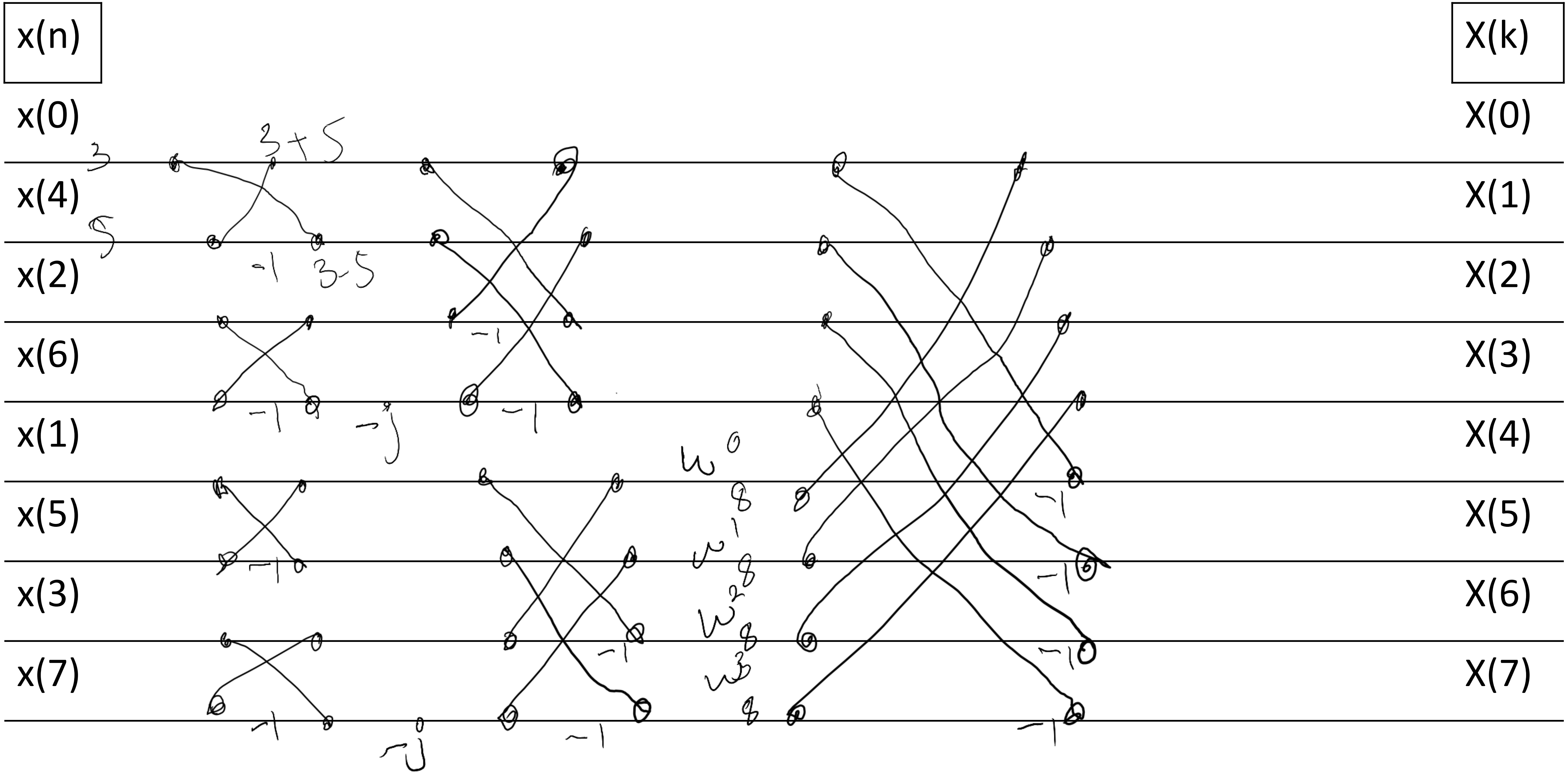
DIT x(n) → X(k) FFT ButterFly Diagram



DIT $x(n) \rightarrow X(k)$ IFFT ButterFly Diagram

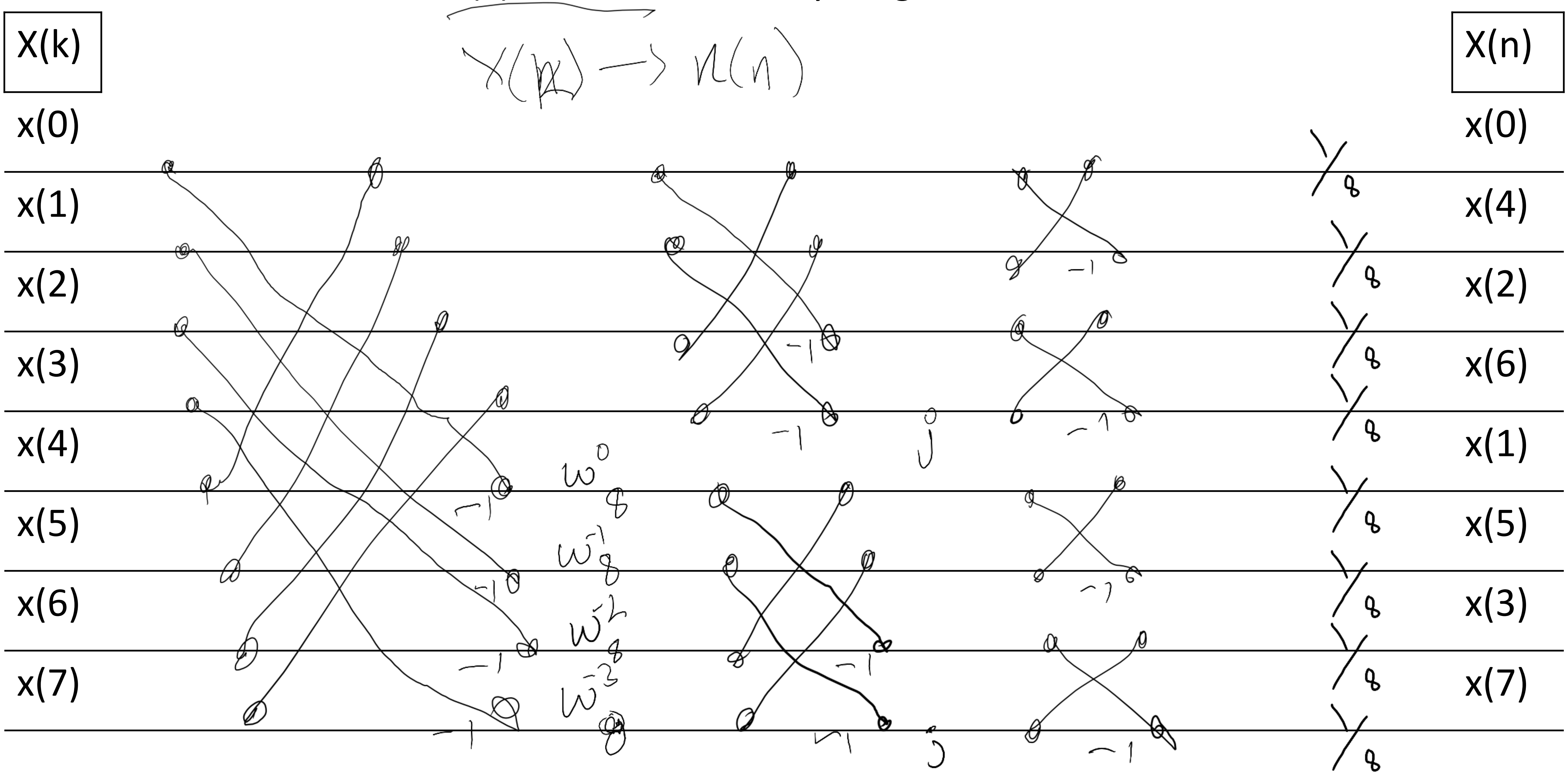


DIT x(n) → X(k) FFT ButterFly Diagram



DIT X(k) → IFFT ButterFly Diagram

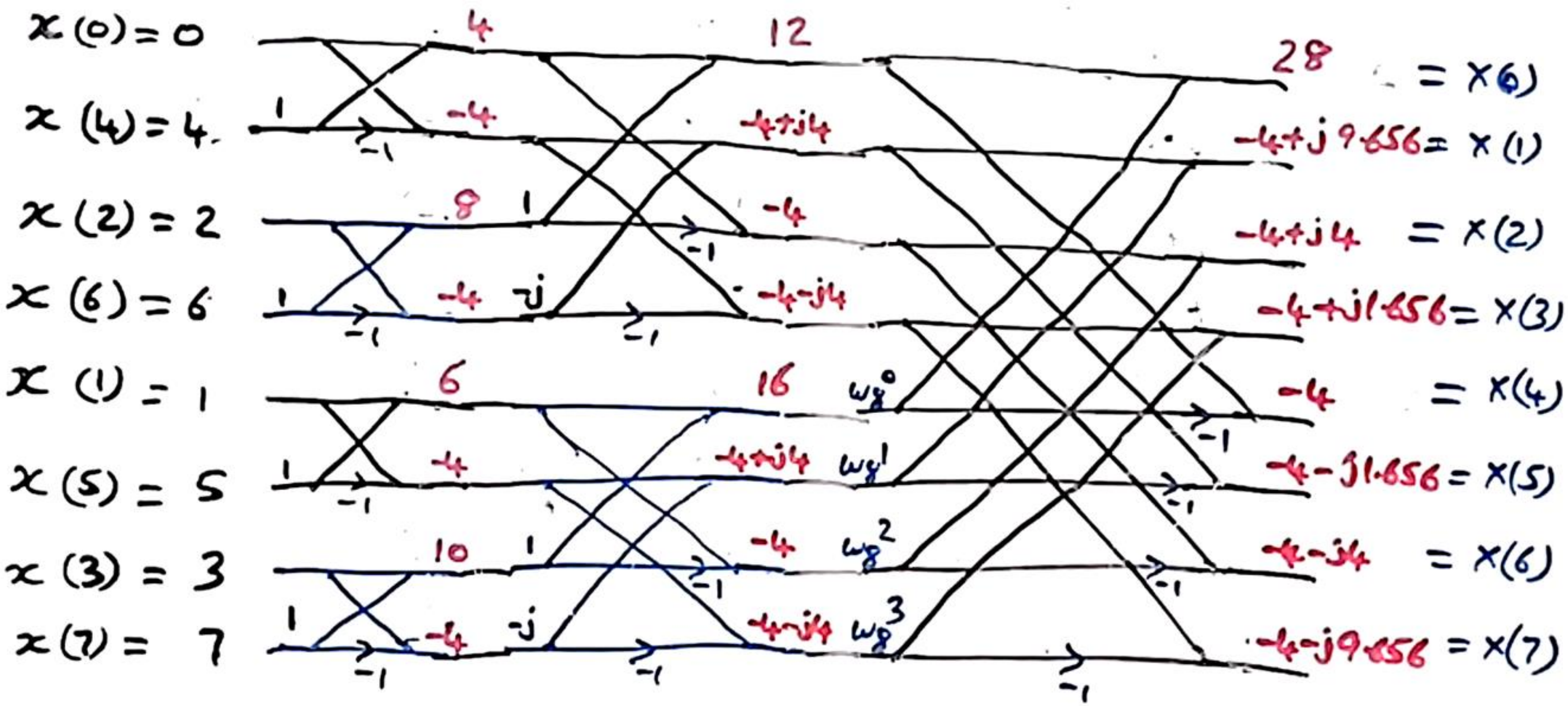
X(k) → x(n)



DIT FFT

$W_8^0 = e^0 = 1$

$W_8^1 = e^{-j\frac{2\pi}{8_4}(1)} = \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} =$



$x(n) = \{0, 1, 2, 3, 4, 5, 6, 7\}$

$X(k) = \{28, -4+j9.656, -4+j4, -4+j1.656, -4, -4-j1.656, -4-j4, -4-j9.656\}$

DIT X(k) → IFFT ButterFly Diagram

X(k) → x(n)