בס"ד

העבודה מועתקת משפיגלמן ירדן 200433332
ומי-משה מושקטל I ת"ז:025008384

יעל ויצמן 312232655

ממן 13 – פרק 5 הפרד ומשול

שאלה 1:

יהי הפולינום:

P(x) = x^3+2x^2-3x-1 → p(x) = -1-3x+$2x^2+x^3$

כאשר:

n=4, w=$e^{\frac{2\pi i}{4}} = i$

הסבר: 1. נפריד את הפונקציה הנתונה ל2 פולינומים המורכבים מהחזקות הזוגיות והאי זוגיות:

- עבור אינדקסים אי זוגיים: Podd (x) = -3+x^2 → Podd(x^2) = -3+x

- עבור אינדקסים זוגיים: Peven (x) = -1+$2x^2$ → Peven (x^2) = -1+2x

2. נראה את תוצאות הרצת האלגוריתם על שורשי היחידה:

	1	i	-1	-i
Peven (x^2) = -1+2x	1	-3	1	-3
Podd(x^2) = -3+x	-2	-4	-2	-4
$P(x) = P_{even}(x^2) + xP_{odd}(x^2)$	-1	-3-4i	3	-3+4i

3. נראה את תוצאות הרצת FFT - INVERSE

	1	-i	-1	i
$P_{even}(x^2) = (-3-4i)x - 3 + 4i$	2	-4	2	-4
$P_{odd}(x^2) = -x + 3$	-6	-8i	-6	-8i
$P(x) = P_{even}(x^2) + xP_{odd}(x^2)$	-4	-12	8	4

4. תיאור ריצת האלגוריתם:

1. Calculating FFT ((-1, -3, 2, 1), w)
   - 1.1. Calling FFT ((-1,3),w^2)
     - a. Calling FFT(-1), w^4) Return (-1)
     - b. Calling FFT((3), w^4) Return (2)
     - c. Return ((-1+1*2), (-1+w^2*2)) = (1, -3)
   - 1.2. Calling FFT((-3,1), w^2)
     - a. Calling FFT((-3), w^4) Return (-3)
     - b. Calling FFT ((1), w^4) Return (1)
     - c. Return ((-3+1*1), (-3+w^2*1)) = (-2, -4)
   - 1.3. Calculate: * f(1) = 1+1*-2 = **-1**
     f(i) = -3+i*-4 = **-3-4i**
     f(-1) = 1+-1*-2 = **3**
     f(-i) = -3+-i*-4 = **-3+4i**
   - 1.4. Return **(-1, -3-4i, 3, -3+4i)**

2. Calculating FFT (**(-1, -3-4i, 3, -3+4i)**, w^{-1})
   - 1.1. Calling FFT((-1, 3), w^{-2})
     - a. Calling FFT((-1), w^{-4}) Return (-1)
     - b. Calling FFT ((3), w^{-4}) Return (3)
     - c. Return ((-1+1*3), (-1+w^{-2}*3)) = (2, -4)

1.2. Calling FFT ((-3-4i, -3+4i) , w^{-2})
   a. Calling FFT((-3-4i), w^{-4}) Return (-3-4i)
   b. Calling FFT ((-3+4i), w^{-4}) Return (-3+4i)
   c. Return ((-3-4i+1*(-3+4i)), (-3-4i+w^{-2}*(-3+4i))) = (-6, -8i)

1.3. Calculating: f(1)= 2+1*-6 = -4
f(i) = -4+w^{-1}*-8i = -4-8 = -12
f(-1) = 2+w^{-2}*-6 = 8
f(-i) = -4+w^{-3}*-8i = -4+8 = 4

1.4. Return(-4, -12, 8, 4)

שאלה 2:

שלבי האלגוריתם:

הנחות:
1. שני המספרים המוכפלים הם שווי אורך.
2. שני המספרים חיוביים
3. ההכפלות שמתבצעות במהלך הקריאות הרקורסיביות אינן מגדילות את אורכם של המספרים.

כעת:
* נמיר לייצוג בינארי את שני הפולינומים שהתקבלו כקלט: X,Y .

* נמיר כל פולינום מייצוג מקדמים לייצוג ערכים - ע"י הרצת אלגוריתם FFT על בלוקים בגודל n/k על כל פולינום בנפרד.
נקבל שני ווקטורי תוצאות שכל אחד מהם הוא בגודל של n/k.
* נכפיל את שני הווקטורים שקיבלנו מהרצת האלגוריתם.
* על מנת לעבור בחזרה מייצוג ערכים לייצוג מקדמים נריץ על תוצאת ההכפלה את אלגוריתם FFT-INVERSE
* נמיר את הערכים המתקבלים מייצוג בינארי לייצוג השלם שלהם.
קיבלנו את התוצאה הסופית של הכפלת שני הפולינומים.

נכונות
האלגוריתם מתבסס על המרת מספרים לבסיס בינארי ולהפך , ועל אלגוריתם FFT
ולכן הנכונות שלו נובעת מנכונות שני אלגוריתמים אלו.

זמן ריצה
* ההמרה מייצוג בינארי ולהפך מתבצע תוך זמן של $\Theta(n)$

* הרצת האלגוריתם *FFT* על בלוקים בגודל *(n/k)* היא $\Theta(\frac{n}{k} * \log\frac{n}{k})$, עלות ההכפלות היא $\Theta(k^2)$ –כפי שנתון ולכן נקבל $\Theta(k^2 * \frac{n}{k} * \log\frac{n}{k}) = \Theta(k * n * log\frac{n}{k})$. נבחר $k = \log n$ ונקבל: $\Theta(\log n * n * log\frac{n}{\log n})$ ועל פי חוקי לוגריתמים:

$$\Theta(\log n * n * log\frac{n}{\log n}) = \Theta(\log n * n * (\log n - \log\log n)) = \Theta(\log n * n * \log n) = \Theta(n * \log^2 n)$$

לכן סה"כ זמן הריצה הוא: $\Theta(n * \log^2 n)$.

שאלה 3

אלגוריתם לחישוב ערכי הנגזרות בנקודה כלשהי:
נגדיר שני ווקטורים: A, B
כך ש:

$A = (n!\,a_n, (n-1)!\,a_{n-1}, \dots, 1!\,a_1, 0!\,a_0)$, ועבור כל i יתקיים: $A_i = (n-i)!\,a_{n-i}$

$B = (\frac{x_0^0}{0!}, \frac{x_0^1}{1!}, \dots, \frac{x_0^n}{n!})$ ועבור כל i יתקיים: $B_i = \frac{x_0^i}{i!}$

ווקטור A מכיל את כל המקדמים של x_0 לכל הנגזרות האפשריות בf.

בווקטור B עבור כל קורדינאטה: במונה יהיה החזקה של x_0 והמכנה יכיל את עצרת גובה החזקה.

על ידי שימוש ב FFT נכפול בין הווקטורים A,B את תוצאת ההכפלה נשמור ב: SUM שהוא הקונוולוציה בין שני הווקרטורים.
כעת על מנת לחשב את ערך נגזרת כלשהי k נחשב:

$$\sum_{0\le i,j\le k \wedge i+j=k} A_iB_j = A_0B_k + \cdots + A_kB_0 == f^{(k)}(x_0)$$

<u>נכונות + סיבוכיות</u>.

1. יצירת שני הווקטורים לוקחת זמן לינארי $O(n)$
2. חישוב ערך נגזרת כלשהי – הצגת התוצאה: $O(n)$
3. הכפלת הווקטורים בעזרת FFT - שני הרצות + הרצה נוספת של FFT^-1 ← $3 * \Theta(n\log n) = \Theta(n\log n)$.

סך הכל זמן הריצה הכולל: $\Theta(n\log n)$

שאלה 4

מימוש ישיר של כפל מטריצות כרוך ב-$\Theta(n^3)$ פעולות אריתמטיות. ביטוי זה נגזר מהעובדה שבכל שלב של הרקורסיה באלגוריתם מפרקים כל מטריצה ל-4 תתי מטריצות מסדר $\frac{n}{2} * \frac{n}{2}$ (כלומר 8 קריאות רקורסיביות – פעולות כפל) ו-4 פעולות חיבור (בין כל שני איברים במטריצה). על כן מתקבלת נוסחה לפי שיטת האב לחישוב זמן הריצה של האלגוריתם - $\Theta(n^{\log_2 8}) = \Theta(n^3)$. האלגוריתם הנתון שונה מאלגוריתם מימוש ישיר של כפל מטריצות בכך שמספר פעולות הכפל ירד ל7 (במקום 8) – ומספר פעולות החיבור-חיסור גדלו ל-18 (מספר הפעולות שיש ב-*P1-P7* ו-*s,t,r,u*). לכן, אם נשתמש שוב בשיטת האב לחישוב זמן הריצה של האלגוריתם נקבל את הנוסחה: $T(n) = 7T\left(\frac{n}{2}\right) + 18n^2 = \Theta(n^{\log_2 7}) = \Theta(n^{2.81})$. מה שצריך להוכיח.