ממ"ן 13

מגיש: רן אלגאוי, 305243115

שאלה 1

סעיף א

מספר סימונים:

i ן- -i הם מספרים לא אמיתיים במישור המספרים המרכובים(C) ולא משתנים.

פולינום אי-זוגי: עבור הערכים -i, -1, i, 1 נקבל את התוצאות -3, 1, -3, 1 בהתאמה.

פולינום זוגי: עבור הערכים -i, -1, i, 1 נקבל את התוצאות -4, -2, -4, -2 בהתאמה.

|  |  |  |
| --- | --- | --- |
| צעד מס' X | פונקציה | מחזיר ערך |
| 1 | FFT((-1,-3,2,1), w= i) | (-1,-3,-4i,3,-5+4i) |
| 2 | FFT((-1,2), w2=-1) | (1, -3) |
| 3 | FFT(-1, w4=1) | -1 |
| 4 | FFT(2, w4=1) | 2 |
| 5 | FFT((-3,1), w2=-1) | (-2, -4) |
| 6 | FFT(-3, w4=1) | -3 |
| 7 | FFT(-1, w4=1) | 1 |
| 8 | לולאה ראשונה | V1 = 1 + w0 \* -2 = -1  V3 = 1 – w0\*-2 = 3 |
| 9 | לולאה שניה | V2 = -3 + w1 \* -4 = -3 – 4i  V4 = -3 – w0\* -4 = -3 + 4i |

סעיף ב

פולינום אי-זוגי: עבור הערכים -i, -1, i, 1 נקבל את התוצאות 4i, -8, 12i, -4 בהתאמה.

פולינום זוגי: עבור הערכים -i, -1, i, 1 נקבל את התוצאות 8i, -6, 8i, -6 בהתאמה.

|  |  |  |
| --- | --- | --- |
| צעד מס' X | פונקציה | מחזיר ערך |
| 1 | FFT((-1,-3,2,1), w= i) | (-4, -12, 8, 4) |
| 2 | FFT((-1,3), w-2=-1) | (2, -4) |
| 3 | FFT(-1, w-4=1) | -1 |
| 4 | FFT(2, w-4=1) | 3 |
| 5 | FFT((-3,1), w2=-1) | (-6, -8i) |
| 6 | FFT((-3-4i), w4=1) | -3 + 4i |
| 7 | FFT((-3+4i), w-4=1) | -3 + 4i |
| 8 | לולאה ראשונה | V1 = 2 + w0 \* -6 = -4  V3 = 2 – w0\*-6 = 8 |
| 9 | לולאה שניה | V2 = -4 + w-1 \* -8i = -12  V4 = -4 – w-1\* -8i = 4 |

שאלה 2

אלגוריתם

1. נחלק את הקלט(גודל n) לn/k בלוקים.
2. עבור כל בלוק:
   1. נציב בX את סכום הX-ים מX0 ועד Xn/k-1.
   2. נציב בY את סכום הY-ים מY0 ועד Yn/k-1.
3. נריץ FFT(X) ו- FFT(Y) ועבור כל ריצה נקבל נקודות שמייצגות אותו.
4. נכפול את שני התוצאות שקיבלנו בשלב (3) ונציב את התוצאה ב-S.
5. נריץ Inverse-FFT(S) ונציב אותם בS\_coefficients.
6. נסכום את כל האיברים בS\_ coefficients ונציב בS\_sum.
7. אם התוצאה לא בבסיס שלם אז נבצע המרה.
8. נחזיר את התוצאה מ7.

הוכחות נכונות

קל לראות שנכונות האלגוריתם נגרר מתוך המרת המקדמים של הפולינומים X ו-Y וכמובן מתוך נכונות האלגוריתם FFT.

חישוב סיבוכיות(לפי שלבי האלגוריתם)

* שלבים 1 ו-2 מתבצעים בזמן גודל הקלט(n).
* שלב 3: הרצת FFT מתבצע בזמן של O(n\*logn).
* שלב 4: הכפלת התוצאות מתבצע בזמן גודל הקלט.
* שלב 5: הרצת Inverse-FFT מתבצע בזמן של O(n\*logn).
* שלב 6: סכימת האיברים מתבצע בזמן גודל הקלט.
* שלב 7: בדיקה והמרת לבסיס אם יש צורך מתבצע בזמן גודל הקלט.
* שלב 8: מתבצע בזמן קבוע( O(1))

שאלה 3

נגדיר שני וקטורים i ו j ונשתמש בהם עם FFT ולמען הנוחות נסמן את x­0­ בX.

וקטור i יכיל את כל המקדמים האפשריים של X­ עבור כל הנגזרות האפשריות של f, ווקטור j יכיל עבור כל נקודה(על המישור) שנמצאת במונה תהייה החזקה של ­­X­ אך במכנה תהייה עצרת בגובה החזקה לפחות.

i = (n! \* a­n­, (n-1)! \* a­(n-1)­, … , 0! \* a­0­)

j = ( [X0/0!], [X1/1!], … , [Xn/n!])­

כעת שהגדרנו את שני הוקטורים שאנחנו צריכים נעבור לשלב החישוב.

נכפול את שני הוקטורים שהגדרנו הנ"ל ע"י שימוש בFFT ונגדיר את התוצאה להיות וקטור k.

וקטור k הוא בעצם כל המחוברים האפשריים שמתקבלים מריצת FFT(הכפלת הוקטורים) שמכילים את הנגזרות של f.

נניח ונרצה לחשב את ערך הנגזרת הS-ית בנקודה X כל שנצטרך לעשות זה להריץ את ריצת FFT עד המספר S, בצורה יותר מפורטת:

נסכום את מכפלת הוקטורים מ0 עד S כל עוד סכום המיקומים של התא הנוכחי בוקטורים לא עולה על S.

סיבוכיות:

* שלב ראשון: בניית הוקטורים בגודל n יתבצע בזמן של O(n).
* שלב שני: הכפלת הוקטורים ע"י שימוש בFFT יתבצע בזמן של O(n\*logn).

בסה"כ אנחנו מקבלים זמן של O(n\*logn).

שאלה 4

לפי הגדרת האלגוריתם(Strassen), הוא מחלק את הבעיה מקורית(כפל מטריצות) ל7 תת-בעיות בסדר גודל של חצי מהבעיה המקורית.

בנוסף מהגדרת האלגוריתם נובע שבכל בעיה מופיע בדיוק פעולת כפל יחידה ובין 1 ל2 פעולות חיבור/חיסור.

אנחנו מקבלים שבכל קריאת רקורסיה אנחנו מבצעים:

מאחר וכל חילוק מתחלקת לחצי מהבעיה המקורית אנחנו מקבלים את הנוסחה הרקורסיבית הבאה:

על מנת לפתור את הנוסחה הנ"ל נשתמש במשפט האב:

נתונים התחלתיים: a=7, b=2.

קל לראות ש f(n) < g(n) וזה עונה על הדרישות למקרה הראשון של משפט האב ולכן אנחנו מקבלים: