בס"ד

ממן 13

נדב מלרמי

שאלה 1.

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

א. נבטן:

נבחר: $\omega = e^{\frac{2\pi}{4}i}$ $\Downarrow$ $\omega = i$

להיות שורש פרימיטיבי. ($0 < k < 4$ $\omega^k \neq 1$)

הרצה בסמנך הבא:

1) FFT((-1, -3, 2, 1), $\omega = i$):

$-1-3x+2x^2+x^3$

1.1) call FFT((-1, 2), -1)

1.1.1) call FFT((-1), 1)

ret (-1) $\longrightarrow f_{e_{1.1}}$

1.1.2) call FFT((2), 1)

ret (2) $\longrightarrow f_{o_{1.1}}$

1.1.3) k=0: $-1+1\cdot 2 = 1$

k=1: $-1+(-1)\cdot 2 = -3$

ret (1, -3) $\longrightarrow f_{e_1}$

1.2) call FFT((-3, 1), -1):

1.2.1) call FFT((-3), 1)

ret (-3) $\longrightarrow f_{e_{1.2}}$

1.2.2) call FFT((1), -1)

ret (1) $\longrightarrow f_{o_{1.2}}$

1.2.3) k=0: $-3+1\cdot 1 = -2$

k=1: $-3+(-1)\cdot 1 = -4$

ret (-2, -4) $\longrightarrow f_{o_1}$

1.3) calc: k=0: $1+1\cdot -2 = 1-2 = -1$

k=1: $-3+i\cdot(-4) = -3-4i$

k=2: $1+(-1)\cdot(-2) = 1+2 = 3$

k=3: $-3+(-i)(-4) = -3+4i$

return $(-1, -3-4i, 3, -3+4i)$

ב- כדי נריץ על הפולינום $(-1, -3-4i, 3, -3+4i)$ ועל השורש

הדרמיטי: $\omega = \frac{1}{\omega} = -i$ את FFT:

1) $FFT((-1, -3-4i, 3, -3+4i), -i)$:

1.1) $FFT((-1, 3), -1)$:

1.1.1) $FFT((-1), 1)$:

ret $(-1) \to fe_{1.1}$

1.1.2) $FFT((3), 1)$:

ret (3) $fo_{1.1}$

1.1.3) $k=0$: $-1+1\cdot(3) = -1+3 = 2$

$k=1$: $-1+(-1)\cdot 3 = -1-3 = -4$

ret $(2, -4) \to fe_1$

1.2) $FFT((-3-4i, -3+4i), -1)$:

1.2.1) $FFT((-3-4i), 1)$:

ret $(-3-4i) \to fe_{1.2}$

1.2.2) $FFT((-3+4i), 1)$:

ret $(-3+4i) \to fo_{1.2}$

1.2.3) $k=0$: $(-3-4i)+1\cdot(-3+4i) = -6$

$k=1$: $(-3-4i)-1\cdot(-3+4i) = -8i$

$\omega = -i$

$(2, -4) \to fe_1$ ret $(-6, -8i) \to fo_1$

1.3) $k=0$: $2+1\cdot(-6) = 2-6 = -4$

$k=1$: $-4+(-i)(-8i) = -4-8 = -12$

$k=2$: $2+(-1)\cdot(-6) = 8$

$k=3$: $-4+(i)\cdot(-8i) = -4+8 = 4$

$\boxed{\text{ret } (-4, -12, 8, 4)}$ $\xrightarrow[n=4]{\text{נפלון ב-}}$ $(-1, -3, 2, 1)$

כמו שצריך! יש"ל

בס"ד

תשובה טובה מאוד נדב!
שמע, חל שיפור ניכר

שאלה 2.

רעיון האלגוריתם:

נפרק את שני המספרים ל- $\frac{n}{k}$ בלוקים בגודל של k ביטים.

כלומר עבור המספר α שמיוצג ע"י n ביטים, נייצג אותו באופן הבא:

$$\alpha = \sum_{i=0}^{\frac{n}{k}-1} \alpha_i \cdot 2^{i \cdot k}$$

כאשר α_i $0 \le i \le \frac{n}{k}-1$ להיות למעשה מספר בעל k ביטים.

יהי β מספר בעל n ביטים. אנו רוצים לחשב את המספר $\alpha \cdot \beta$.

נייצג גם את β באותו האופן: $\beta = \sum_{i=0}^{\frac{n}{k}} \beta_i 2^{ik}$, ולכן:

$$\alpha \cdot \beta = \left(\sum_{i=0}^{\frac{n}{k}-1} \alpha_i \cdot 2^{i \cdot k} \right) \left(\sum_{i=0}^{\frac{n}{k}-1} \beta_i 2^{ik} \right)$$

כדי לגדיר את הפולינומים (המוגדרים בייצוג הינארי) הבאים:

$$P_\alpha(x) = \alpha_0 \cdot x^0 + \alpha_1 x^1 + \dots + \alpha_{\frac{n}{k}-1} \cdot x^{\frac{n}{k}-1}$$

$$P_\beta(x) = \beta_0 \cdot x^0 + \beta_1 \cdot x^1 + \dots + \beta_{\frac{n}{k}-1} x^{\frac{n}{k}-1}$$

נשים לב כי: $\alpha = P_\alpha(2^k)$, $\beta = P_\beta(2^k)$

בס"ד

ולכן, $\alpha \cdot \beta = P_\alpha(2^k) \cdot P_\beta(2^k)$

לכן נידרש ל מצוא את הפולינום $P_\alpha \cdot P_\beta$ ולאחר שמצאנו

אותו, נציב $(P_\alpha P_\beta)(2^k)$ וזהו למעשה המכפלה הנדרשת. כל צורך שלנו

הוא למעשה מכפלה של חזקה של שתיים ושני מספרים בני k ביטים (לפי הנחה בשאלה

מספר הספרות של המספרים לא גדולים מעל k).

לכן בעזרת FFT על P_α ו- P_β נמצא את פולינום המכפלה, ולאחר

שנחזיר אותו ליצוג של מקדמים (InvFFT) נציב בו 2^k, נסכום את

המחוברים ונחזיר את הערך. את k נבחר ע"י $\log n$.

לכן האלגוריתם הינו:

Mult (α, β):

$n \leftarrow |\alpha|$

$k \leftarrow \lg n$

// Let P_α and P_β be the Polinomyals to multiply:

$P_\alpha \leftarrow$ split α to $\frac{n}{k}$ chunks of k bits and mult by 2^i $0 \le i \le \frac{n}{k}-1$

$P_\beta \leftarrow$ " β " " " " " " " " " "

// Let Q be the Polynom $P_\alpha \cdot P_\beta$:

$Q \leftarrow FFT(P_\alpha, \omega^{\frac{2n}{k}-2}) \bullet FFT(P_\beta, \omega^{\frac{2n}{k}-2})$

$Q \leftarrow InvFFT(Q, (\omega^{\frac{2n}{k}-2})^{-1})$

Divide all Q components with $\frac{2n}{k}-2$

sum $\leftarrow 0$

For $i \leftarrow 0$ to $\frac{2n}{k}-2$:

// $Q[i]$ is mult of $\alpha^* \cdot \beta^* \cdot 2^i$ Where α^* and β^* are

// numbers with k bits

sum $\leftarrow$ sum + $Q[i]$

return sum

הוכחת נכונות:

כפי שהסברנו בהקדמה ("רעיון האלגוריתם") המשימה שהאלגוריתם אכן מחזיר את המכפלה $\alpha\cdot\beta$ שכן:

$$\alpha\cdot\beta = \left(\sum_{i=0}^{\frac{n}{k}-1}\alpha_i\cdot 2^{ik}\right)\left(\sum_{i=0}^{\frac{n}{k}-1}\beta_i 2^{ik}\right) = \left(P_\alpha(2^k)\cdot P_\beta(2^k)\right) =$$

"גוש" ע"י $\frac{n}{k}$ בלוקים

לפי הגדרת הפולינומים P_α P_β:

$$P_\alpha(x) = \alpha_0\cdot x^0 + \alpha_1 x^1 + \dots + \alpha_{\frac{n}{k}-1}\cdot x^{\frac{n}{k}-1}$$

$$P_\beta(x) = \beta_0\cdot x^0 + \beta_1 x^1 + \dots + \beta_{\frac{n}{k}-1}\cdot x^{\frac{n}{k}-1}$$

לפי הגדרת מכפלת פולינומים

$$= (P_\alpha P_\beta)(2^k)$$

אבל זהו הערך שהאלגוריתם משיג. ולכן האלגוריתם מחזיר את המכפלה.

כעת בראה שהאלגוריתם אכן מחזיר את הערך $(P_\alpha P_\beta)(2^k)$:

בחלק הראשון אנו מחשבים את P_α ואת P_β (את המקדמים).

בחלק השני מבצעים FFT על כל אחד עם $2-\frac{2n}{k}$ שורשי יחידה. נאלץ מכרלים רביב-רביב כדי לקבל את $(P_\alpha P_\beta)$. לאחר מכן מבצעים InvFFT עם ההופכי של שורש היחידה מסדר $2-\frac{2n}{k}$ שמשתנים לטל. ולכן כעת יש לנו את פולינום המכפלה באופן הנכון.

בחלק השלישי אנו מחשבים את הערך $(P_\alpha P_\beta)(2^k)$ ע"י כפלה ולכסל מחזירים

אחרי הסדרן הגבוהים □

סיבוכיות:

חלק ראשון: מציאת הכפלינומים P_α ו- P_β – $O(n)$

מכיוון שאנו עובדים באופן לינארי על הדגים.

חלק שני: FFT על P_α ו- P_β מבחינת זמן ריצה:

נסמן: $a = \frac{n}{k}$. מכיוון שהגודל של הקלט הוא a סכומות

(שכל אחת בגודל k דגים) נקבל שנוסחת הנסיגה הינה:

$$T(a) = 2 \cdot T\left(\frac{a}{2}\right) + \underbrace{k^2 \cdot a}$$

העבודה הנעשית אחרי הקריאות הרקורסיביות.

כל פעולת כפל שנעשית על $O(a)$ המחברים אורכת

$\Theta(k^2)$ פעולות על דגים.

ולכן הדיתרון של נוסחה זו הוא:

$$T(a) = \underbrace{\log(a)}_{\text{רמות}} \cdot \underbrace{(k^2 \cdot a)}_{\text{עבודה בכל רמה}} \underset{a=\frac{n}{k}}{=} \log\left(\frac{n}{k}\right) \cdot \left(k^2 \cdot \frac{n}{k}\right) =$$

$$= \log\left(\frac{n}{k}\right) \cdot (n \cdot k) \underset{k=\log n}{=} \log\left(\frac{n}{\log n}\right) \cdot (n \log n) = (\log n - \log(\log n)) n \log n =$$

פה המעבר צריך להיות O גדול ולא טטא

$$(\log n - \log(\log n))\, n\log n = n\log^2 n - n\log n(\log(\log n)) = \boxed{\Theta\, n\log^2 n}$$

$\Leftarrow$ לכן ההרצות FFT על P_α , P_β עולות $\Theta\, n\log^2 n$

הכפל רכיב רכיב של $P_\alpha \cdot P_\beta$ עולה $k^2 \cdot \left(\frac{2n}{k} - 2\right)$

מספר הרכיבים

$$= k \cdot 2n - 2k^2 = 2n\log n - 2\log^2 n = \boxed{\Theta\, n\log n}$$

הרצה של InvFFT על $(P_\alpha P_\beta)$ באופן דומה עולה $\Theta\, n\log^2 n$.

בהמשך השאיפה אנו מחשבים את הערך $(P_\alpha P_\beta)(z^n)$, שצריך למעשה סכימה של כלל המחברים בפולינום המכפלה. בפולינום זה יש $2\frac{n}{k}$ איברים, כאשר בכל איבר יש מכפלה של שני מספרים בני k ביטים וכן הכפלה בחזקה b שתיים. הכפלה בחזקה של שתיים הינה למעשה "שיפוץ" של הביטים כמספר החזקה לכן ניתן בעצם להתייחס לכפל בשתיים כ- $\Theta(1)$. כלומר שוב זה עולה:

$$O\left(2 \cdot \frac{n}{k} \cdot k^2\right) = O(2n \cdot k) = O(nk) = \boxed{O(n\lg n)}$$

הכפלה של שני מספרים בני k ביטים

$\Leftarrow$ סה"כ הזמן ריצה הינו $\Theta n\lg^2 n$ $\square$

בס"ד

תשובה מעולה ומפורטת!
הכי טובה שראיתי עד כה!
שאפו!

לא יודע אם שמת לב אבל ביקשו תשובה בעלת 4-5 שורות בלבד...אבל אני לא אוריד לך על זה

שאלה 3-

תחילה נשים לב לעובדה הבאה. לכל $0 \le k \le n$, ניתן לבדוק את הנגזרת באופן הבא:

$$f^{(k)}(x) = \sum_{j=k}^{n} \frac{j!}{(j-k)!} \cdot a_j x^{j-k} \underset{\text{נפדר}}{=} \sum_{j=k}^{n} j!\, a_j \cdot \frac{x^{j-k}}{(j-k)!} \quad (*)$$

עלינו לחשב לכל $0 \le k \le n$ את הערך $f^{(k)}(x_0)$.

רעיון האלגוריתם:

נגדיר שני וקטורים, P ו-Q באורך $n+1$, שהקונבולוציה שלהם תיתן את הווקטור הבא:

$$P * Q = \left(f^{(n)}(x_0),\ f^{(n-1)}(x_0),\ \dots,\ f^{(1)}(x_0),\ f^{(0)}(x_0) \right)$$

כלומר, הווקטור שמתקבל ע"י הכפלת הקונבולוציה יהיו למעשה התשובה שאנו מחפשים!

איך נגדיר את P ו-Q?

לפי $(*)$:

$$\sum_{j=k}^{n} j!\, a_j \cdot \frac{x^{j-k}}{(j-k)!}$$

נבחין כי ישנם שני גורמים בכל מחובר. לכן נגדיר את הווקטורים באופן הבא:

$$P = (p_0, p_1, \dots, p_n) \quad \Big| \quad p_t = (n-t)! \cdot a_{n-t} \quad 0 \le t \le n$$

$$Q = (q_0, q_1, \dots, q_n) \quad \Big| \quad q_t = \frac{x^t}{t!} \quad 0 \le t \le n$$

בס"ד

נרצה כי $\bar{P}*\bar{Q}$ הוא הוקטור המקיים (שהרכיבים שלו הם חישובי הנגזרות):

יהי $0\le k\le n$. נתבונן ברכיב ה-k בווקטור $\bar{P}*\bar{Q}$:

לפי הגדרת קונבולוציה:

$$(\bar{P}*\bar{Q})_k=\sum_{\substack{s+t=k\\ 0\le s,t<n}} p_s\cdot q_t \underset{\substack{\text{הגדרת}\\ P\ \text{ו-}\ Q}}{=} \sum_{s+t=k}\left((n-s)!\cdot a_{n-s}\right)\cdot\left(\frac{x^t}{t!}\right) \underset{\substack{\text{מניפולציה על}\\ \text{האינדקסים:}\\ s=k-t}}{=}$$

$$\sum_{t=0}^{k}(n-(k-t))!\cdot a_{n-(k-t)}\cdot\frac{x^t}{t!}=\sum_{t=0}^{k}(n-k+t)!\,a_{n+t-k}\cdot\frac{x^t}{t!} \underset{\substack{\text{מניפולציה}\\ \text{על האינדקס}}}{=}$$

$$\sum_{m=n-k}^{n} m!\,a_m\cdot\frac{x^{m-(n-k)}}{(m-(n-k))!} \underset{\substack{\text{נשים לב שהביטוי}\\ \text{משמאל בדיוק למעשה}\\ \text{הנגזרת לפי } \circledast \text{ (מתמטיקת השאלה)!}}}{=} f^{(n-k)}(x)$$

$$\left[\ 0\le k\le n \Rightarrow \boxed{0\le n-k\le n}\ \right]$$

$\Rightarrow$ כלומר, הוכחנו כי הרכיב ה-k בקונבולוציה $\bar{P}*\bar{Q}$ הינו למעשה הערך של הנגזרת ה-$n-k$ המבוקשת (כמובן עבור x_0 נתון) כעת נתאר את האלגוריתם בשלבים - נקבל כקלט את וקטור המקדמים $(a_0,a_1,\dots,a_n)$ ואת x_0.

1) נחשב ונשמור את הערכים $(0!,1!,\dots,n!)\to$ Fact באופן איטרטיבי.

2) כעת נבנה את P: $\forall_{0\le j\le n}: P_j\leftarrow \text{Fact}[n-j]\cdot a_{n-j}$

3) " " " Q: $\forall_{0\le j\le n}: Q_j=\frac{x_0^j}{\text{Fact}[j]}$

4) א.1) נריץ:

$DFT_P \leftarrow FFT(P, \omega)$ עם $2n$ שורשי יחידה.

$DFT_Q \leftarrow FFT(Q, \omega)$ " " " "

ב.2) נכפל רכיב-רכיב:

$result \leftarrow DFT_P \cdot DFT_Q$

ג.3) נריץ: $result \leftarrow Inv\text{-}FFT(result, \omega^{-1})$

לאחר מכן נחלק ב-$2n$ את result

ונחזיר את result.

הוכחת נכונות:

בכדי שהנחנו לעיל, הקונבולוציה שמתקבלת היא אכן וקטור שערכיו הם הנגזרות של $f(x)$ בנקודה x_0. משוגלגין שהערך שבמקום ה-k בווקטור התוצאה הוא למעשה הערך $f^{(n-k)}(x_0)$.

כמו כן, ברור שהוקטור result הינו אכן קונבולוציה שכן הוא תוצאת הכפלת הפולינומים שמתאימים לוקטורים Q ו-P שהגדרנו, וכבר מקדמי הפולינום הוא מקרה פרטי של קונבולוציה.

ניתוח סיבוכיות:

שלב 1) לוקח $\Theta(n)$ (משום אינטואיציה של העדרות).

שלב 2) ו-3) לוקחים $\Theta(n)$ (בגלל הוקטורים P ו-Q באורך אינטרטיבי)

שלב 4.1) - הרצה של כמה FFT על $2n$ ערכים של הוקטורים P ו-Q $\Theta(n\log n)$

בס"ד

שלב 4.2) מכפלת שני וקטורים רכיב-רכיב - $\Theta(n)$

שלב 4.3) הרצת InvFFT על result באורך 2n - $\Theta(n\lg n)$

סיכום רכיבי הווקטור ב- 2n - $\Theta(n)$

$\Longrightarrow$ סה"כ $\Theta(n\lg n)$

שאלה 4-

נתאר את האלגוריתם של Strassen לפי המשוואות שנתונים בנתון.
הערה: האלגוריתם מקבל את המטריצות A ו-B מסדר חזקת 2 ומחזיר
את המטריצה A*B שמסדר חזקת 2:

```
Strassen (A, B):
if |A|=|B|=1:
    return |A|*|B|
Split A and B into four quarters:
    a,b,c,d ← A
    e,f,g,h ← B
P1 ← strassen(a, g-h)
P2 ← strassen(a+b, h)
P3 ← strassen(c+d, e)
P4 ← strassen(d, f-e)
P5 ← strassen(a+d, e+h)
P6 ← strassen(b-d, f+h)
P7 ← strassen(a-c, e+g)
s ← P1+P2
t ← P3+P4
r ← P4+P5+P6-P2
u ← P1+P5-P3-P7
C ← (r s; t u)
return C
```

בס"ד

תחילה נעיר:

* חיבור/חיסור של שני מטריצות A ו-B מסדר n פועל בזמן של $\Theta(n^2)$ שכן ישנם n^2 איברים בכל מטריצה שצריך לעבור אחד-אחד ולחבר/לחסר.

* אומנם השיטה מקבלת שתי מטריצות בגודל חצי, אך ניתן לגזור הקלט כ-n. כלומר זמן הריצה הינו כפונקציה של n.

כעת נבנה את זמן הריצה של השיטה strassen(A,B):

I) חלוקת המטריצות לרבעים אורכת $\Theta(n^2)$ (עוברים על n^2 האיברים של A. אותו דבר עבור B $\Leftarrow 2\cdot n^2=\Theta(n^2)$.)

II) ניתן לראות בבחול את העובדה שהשיטה מבצעת עבור חיבור או חיסור של מטריצות הרבעים שהן בגודל $\frac{n}{2}\times\frac{n}{2}$.
לכן כל פעולת חיבור/חיסור אורכת $\frac{n^2}{4}=\Theta(n^2)$.
ניתן לראות כי ישנן 18 פעולות כאלה $\Leftarrow 18\cdot n^2=\Theta(n^2)$

III) ניתן לראות בבאופן שבעת הקריאות הרקורסיביות ששולחות כפרמטר מטריצות בגודל $\frac{n}{2}\times\frac{n}{2}$ $\Leftarrow$ גודל הקלט הקטן בחצי. לכן, נסמן את זמן הריצה של השיטה עבור קלט בגודל n כך: $T(n)$

$\Leftarrow$ לכן שבעת הקריאות לה למשל העבודה הבאה:

$7\cdot T(\frac{n}{2})$

כס"ד

$\Leftarrow$ נוסחת הנסיגה עבור השגרה: $T(n) = 7 \cdot T(\frac{n}{2}) + 18 \cdot n^2$

I, II, III

נ'עצר בשגרה הזאת מקרה א'. נשים לב כי $f(n) = 18n^2 = \Theta(n^2)$

$a = 7$
$b = 2$

נבחר $\varepsilon = 0.1$:

$$n^{\log_2 7 - \varepsilon} = n^{2.8 - 0.1} = n^{2.7}$$

מתשמן...

כלומר:

$$f(n) = 18 \cdot n^2 = \Theta(n^2) = O(n^{2.7}) = O(n^{\log_2 7 - 0.1}) = O(n^{\lg_a b - \varepsilon})$$

לכן: $T(n) = \Theta(n^{\log_2 7})$ $\square$