Hw3_212077333

Monday, May 1, 2023 3:59 PM

שאלה 1

א/ <u>הוכחה</u>

$$27^{\log_3 n} = O(n^4)$$

בתרגול ראינו כי

$$f = O(g) \iff \limsup_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

אז

$$\limsup_{n\to\infty} \frac{27^{\log_3(n)}}{n^4} \overset{\text{שינוי בסיס לוגים}}{=} \limsup_{n\to\infty} \frac{27^{\frac{\log_{27}(n)}{\log_{27}3}}}{n^4} =$$

$$= \limsup_{n\to\infty} \frac{27^{\log_{27}(n)\cdot 3}}{n^4} = \limsup_{n\to\infty} \frac{n^3}{n^4} = 0 < \infty$$

ב. $\log(n) = O(\log(\log n^2))$ <u>הפרכה</u>

נניח בשלילה לכן קיים c וn_0 כך שלכל מתקיים

בגלל השלילה לכן קיים c ו-n_0 כך שלכל $n \geq n_0$

$$\log(n) \leq c\log(\log(n^2))$$

נעלה בחזקת e

$$e^{\log(n)} \leq e^{c\log(\log(n^2))}$$

$$\Downarrow$$

$$n \leq c\log(n^2) = c\cdot 2\log(n) \Rightarrow$$

$$n = O(\log(n))$$

וזו סתירה [illegible]

<u>הפרכה</u>

3. אם $f_1(n) = O(g_1(n))$ $f_2(n) = O(g_2(n))$

אזי $f_1 \circ f_2(n) = O(g_1 \circ g_2(n))$

$f \circ h(n) = f(h(n))$ סימון

$f_1(n) = e^n$ $f_2(n) = 2\ln(n)$

$g_1(n) = e^n$ $g_2(n) = \ln(n)$

דוגמא

$$e^{2\ln(n)} = n^2 = O(e^{\ln(n)} = n)$$

$$\Downarrow$$

$n^2 = O(n)$ וזה ממש לא נכון

$n^2=O(n)$ וזה משהו לא נכון

4 לא $\xrightarrow{\text{הפרכה}}$

$f=O(2^n)$ ו $g=2^{O(n)}$

נבחר $2n=O(n)$ אזי $g=2^{2n}=2^n\cdot 2^n$

בנוסף $f=c2^n$ במובן כלל

$$\limsup_{n\to\infty}\frac{2^n\cdot 2^n}{c2^n}=\infty$$

ולכן $g\neq O(f)$ הפרכה

ב 1.

דרוש $0\le a_1,\dots,a_n$ מספרים אי שליליים אם יש כמה

שני המספרים $0<b,c\le 1$ רק שלפעמים n-ם מאיברי

דרוש כדי הרמו הבובול של לפחות $c\cdot\max\{a_1,\dots,a_n\}$

אנשתקירים

$$\sum_{i=1}^{n} a_i = \Theta\left(n \cdot \max\{a_1, \dots, a_n\}\right)$$

סל הספומרים סהומינכרים

ברכת 13 כיונגי $f=\Theta(g) \iff g=O(f) \mid f=O(g)$

מספרת ⇐

$$\sum_{i=1}^{n} a_i \le n \cdot \max\{a_1, \dots, a_n\} \in O\left(n \cdot \max\{a_1, \dots, a_n\}\right)$$

$$\sum_{i=1}^{n} a_i \ge n \cdot b\, c \cdot \max\{a_1, \dots, a_n\}$$

מצית ⇐

$$\limsup_{n\to\infty} \frac{n \cdot \max\{a_1, \dots, a_n\}}{\sum_{i=1}^{n} a_i} \le \limsup_{n\to\infty} \frac{n \cdot \max\{a_1, \dots, a_n\}}{n \cdot b \cdot c \cdot \max\{a_1, \dots, a_n\}}$$

$$= \frac{1}{b \cdot c} \in \mathbb{R} < \infty$$

לכן $\Omega\left(\sum_{i=1}^{n} a_i\right) = n \cdot \max\{a_1, \dots, a_n\}$

$$\Downarrow$$

$$\sum_{i=1}^{n} a_i = \Theta\, n \cdot \max\{a_1, \dots, a_n\}$$

2) הוכיחו כי מתקיים $n\log(n) = \Theta\log(n!)$

$$\log(n!) = \log(n\cdot(n-1)\cdot(n-2)\cdots 2\cdot 1) =$$

$$= \log(n) + \log(n-1) + \dots + \log(2) + \log(1)$$

$$= \sum_{i=1}^{n} a_i \qquad a_i = \log(i)$$

בסעיף א' הראינו כי

$$\sum_{i=1}^{n} a_i = \Theta\, n\cdot\max\{a_1, \dots, a_n\} = \Theta\, n\cdot\log(n)$$

לכן גם $n\log(n) = \Theta(\log(n!))$

∎

3. בהינתן שתי [illegible] נגדיר

$\sum^{n}$ א

$$p_\alpha(n) = \sum_{i=1}^{n} i^\alpha$$

הוכיחו לכל קבוע $1 \le \alpha$ שמתקיים

$$p_\alpha(n) = \Theta(n^{\alpha+1})$$

בסעיף 1 הוכחנו לכל n מתקיים

$$\sum_{i=1}^{n} a_i = \Theta\ n \cdot \max\{a_1, \dots, a_n\}$$

נציב i^α

$$\sum_{i=1}^{n} i^\alpha = \Theta n \cdot \max\{1^\alpha, \dots, n^\alpha\}$$

$$\max\{1^\alpha, \dots, n^\alpha\} = n^\alpha$$

לכן

$$p_\alpha(n) = \sum_{i=1}^{n} i^\alpha = \Theta\ n \cdot n^\alpha = \Theta n^{\alpha+1}$$

4.

$$\sum_{i=1}^{n} 2^i \cdot i^k = \Theta(2^n \cdot n^k)$$

נשים לב כי

$$\sum_{i=1}^{n} 2^i \cdot i^k = 2^n \cdot n^k + \sum_{i=1}^{n-1} 2^i \cdot i^k > 2^n \cdot n^k$$

ולכן

$$2^n \cdot n^k = O\left(\sum_{i=1}^{n} 2^i \cdot i^k\right)$$

כעת נוכיח כי לכל n

$$\sum_{i=1}^{n} 2^i \cdot i^k = O(2^n \cdot n^k)$$

מקרה בסיס n=1

$$2^1 \cdot 1^k = 2^1 \cdot 1^k$$

לכן נכון

נניח n-1

$$\sum^{n-1} 2^i \cdot i^k = \Theta(2^{n-1} \cdot (n-1)^k)$$

$i=1$

ברכיות ל מ

$$\sum_{i=1}^{n} 2^i \cdot i^k = \sum_{i=1}^{n-1} 2^i \cdot i^k + 2^n \cdot n^k$$

$f_1 + f_2 = O(g_1 + g_2)$

$$\downarrow$$

$$= O\left(2^{n-1} \cdot (n-1)^k + 2^n \cdot n^k\right) \overset{1}{=} O(2^n \cdot n^k)$$

$$\left(2^{n-1} \cdot (n-1)^k + 2^n \cdot n^k \le 2^n \cdot (n)^k + 2^n \cdot n^k = 2(2^n \cdot n^k)\right) \overset{1}{}$$

$$= O(2^n \cdot n^k)$$

לכן

$$\sum_{i=1}^{n} 2^i \cdot i^k = O(2^n \cdot n^k)$$

2) ניתוח סיבוכיות במקרה הגרוע וביטוי הדוק ככל הניתן

1.

נבדוק כמה פעמים נבצע append שבכל

צריך לבצע את ה if הזה ובנוסף לכך ברשימה

והוא תלוי באורך הרשימה n שמתחלק

הרשימה ב2 כל פעם

$$\sum_{i=1}^{\log_2(n)} \frac{n}{2^i} = \Theta(n)$$

לכן סיבוכיות זמן הריצה של ה if הוא $O(n)$ בנוסף למדוד

כמה פעמים שבו מבצעים for בעבור על לולאת $\frac{n}{2}$

פעמים שוב $\frac{n}{4}$ פעמים וכן הלאה ומכל איטרציה נבצע

את פעולת ה if שעולה בסיבוכיות $\Theta(n)$

לכן סה"כ הסיבוכיות היא

$$\sum_{i=1}^{\log_2(n)} \frac{n}{2^i} \Theta(n) = \Theta(n) \cdot \Theta(n) = \Theta(n^2)$$

2.

לולאת ה for מתבצעת כמס-n פעמים

ועוד מכן לולאת for פנימית שרצה $\log_2(n)$

לאחר מכן לולאת for רצה כמויות משתנות של $\log_2(500)$ פעמים ובאיטרציה השנייה $\log_2(501)$ וכך הלאה עד $\log_2(n)$

בכל איטרציה כמויות קיימות לולאת while שרצה $\log_2(n)$ פעמים לכן

$$\log_2 n \cdot \log_2 500 + \dots + \log_2 n \cdot \log_2 n = \log n \cdot \log n \cdot (n-500)$$

הוכחנו ש $n\log n = \log(n!)$ ולכן נקבל

$$O\left(n(\log_2(n))^2\right)$$

■

3.

נשים לב כי לולאת ה for מתבצעת $n+1$ פעמים ובתוכה ישנה לולאת while שמתנהגת היא מתבצעת $\log_2(n)$ פעמים ובתוכה שלו פעולות שבכל איטרציה עוברת פעולות וכן $O(n)$

לכן

$\log_2(1)\cdot n + \log_2(2) n + \dots + \log_2(n+1) n$

$n\log(n) = \log(n!)$

$= (n+1)\log((n+1)!) = O(n \cdot n \log_2 n) = n^2 \log_2(n)$

2

a. b.

$\text{sign} = 0 \quad \text{exp} = 1024$

$\Downarrow$

$+$

$\square . \square \dots . \square$

$\text{value } 110 \dots 0 = 1\cdot\frac{1}{2} + 1\cdot\frac{1}{4} + 0 \dots 0$

$+ \cdot 2^{1024-1023}\left(1+\frac{1}{4}+\frac{1}{2}\right) = 2\left(1+\frac{3}{4}\right) = \boxed{3.5}$

b.

$\text{sign} = 1 \quad \text{exp} = 1024 + 2$

ל

fraction $= \frac{1}{2}$

$$-\cdot 2^{1026-1023}\left(\frac{1}{2}\right) = -2^{3}\left(1+\frac{1}{2}\right) = -8\cdot 1.5 = \boxed{-12}$$

ב. sign=0 | n מספר שלם ש $-1023 \le n \le 1024$

הכוון שמוגדרת $[2^n, 2^{n+1})$ בין כל שני מספרים סמוכים

הכי קרובים לידיד ושווה אותו הפרש (ותשמש אותו כברגינגרפ

של n

לפי הפרוטוקול IEEE 754 בשיטה שהוצגה בה

כל שני מספרים סמוכים חייבי הופכוון הסמוך של

ביט בודד לווווים 1 ל 0 0

ל fraction

ולפי הפרוטוקול מרחק כה הוא $2^n \cdot \frac{1}{2^{52}}$

ג. סיכום באך הרווח בתחום $[2^{n+1}, 2^{n+2})$ שווה

כתוצאה הטווח

הרווח ב $[2^{n+1}, 2^{n+2})$ לכן $\frac{2^{n+1}}{2^{52}}$

גדול פי 2 מ $[2^{n}, 2^{n+1})$

c. איך סעיפים a ו b היו משתנים אם היינו מוסיפים ביט ל fractal

אני הרווח בסעיף א היה $\frac{2^{n}}{2^{53}}$

אך סעיף b לא היה משתנה כלל

d. הסבירו מדוע יש הבדל בין שתי התוצאות

החישובים האחרונים בבדיקת הטבלה

בשורה השנייה אנו משלבים את המספר בשלם וכך

טיפוס של משתמש בפונקציות IEEE 754 לייצוג המספר

לעומת זאת בשורה השלישית

השינוי נובע מכך שלא ניתן לייצג את המספר

$900719925474099\underline{3}.0$

כי מכפילים a המספר שיכול לצאת $[2^{53}, 2^{54})$ והחישוב של 2 עוקבים

בשונים הוא $2^{53-52} = 2$ לכן בשורה השלישית הוא

לא תקין במדויק

3 ס ii הסבר הפתרון

התוכנית עוברת על כל זוג איברים קרובים ושומרים

הפרשים ובודקת את הפער שלהם כמון את הגדול

כמון הוא מחזירה ב-ניקוד ובסוף מחזירה את ההפרש

סיבוכיות זמן הריצה היא $O(n^2)$ בגלל ש הכפולות

ערך לבדוק ה- הפרשים כפי $O(n)$ שנעשו עושים לולאה

פנימית לכן $= 2O(n)$ $O(n)$ ובכולל יש בחות עשה

שגעו עושים מעדיפים לשון $O(n^2)$

חלק 2

3 ס

מ

כאשר אני מגדיר את lst2 בס k

ב עלות

לאחר מכן קומפו for loop שרוקת רק k בעלות

ובתוכה while שלוקח $\log_2(n//k)$

לסיכום

$$O(k)+O(k)\cdot\log_2(n//k)=O(k\log_2(n//k))$$

$$=O(k\log_2(n))$$

ש 3

הפונקציה מפעילה ב עומדת בקריטריון סיבוכיות הזמן משום ש

מימוש רשימת העזר הינו 5^k

לאחר מכן אנו לפל n שאנו מפעילים את הפונקציה

string_to_int לכן nk

מכיוון שאנו עוברים על כל איברי רשימת העזר ומבצעים
כך n פעמים שאנו מבצעים המרה מ int_to_string
שלוקחות $O(k)$ לכן $5^k + nk$ לסיכום

$$25^k + 2nk = O(5^k + nk)$$

4 ו

אנו עוברים על כל הערכים מ 1 עד 5^k

לכל ערך i אנו עוברים על n ערכים

ממירים כל ערך בעזרת string_to_int שלוקחת

זמן קבוע ולא תופסת בזיכרון

בודקים if ועושים append $O(1)$ לכן

לכן סיבוכיות זמן $O(5^k \cdot nk)$

סיבוכיות מקום $O(k)$