תשס"ח סמסטר ב'
תרגול 2
Sorting, Compatison Model, and Lower Bounds

מבני נתונים

מתרגל:
דן פלדמן

השלמה מהשיעור הקודם,
כידוע שינוי הבסיס מעבר בסיסים.

$\log_a b = \frac{\log b}{\log a}$

פרוט תרגיל 1 משיעור קודם

$$9^{\log_7 n}+n\left(\frac{9}{7}\right)^{\log_7 n} = 9^{\log_7 n}+n\left(\frac{9^{\log_7 n}}{7^{\log_7 n}}\right)$$
$$=2\cdot 9^{\log_7 n} = 2\cdot 9^{\log_9 n\cdot\log_7 9} = 2\cdot n^{\log_7 9} \; (*)$$

(*) צריך לשים לב לא לשנות ב-(*) את הביטוי ל-$\Theta(1)$ כיוון שיש הבדל אם מדובר על 1 או 2 לדוגמה.

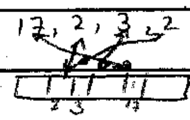
$\Theta(2^n) \neq \Theta(3^n)$ טעויות נפוצות
(1)

(2) לא לשכוח אם אני מפתחים נוסחת נסיגה אסור לי להפוך משתנים $\Theta(..)$ רק בשלב האחרון ($T(1)$).

מיונים

- Counting Sort
  - ממיין n מפתחות של שלמים בין 0 ל-n בזמן לינארי.
  - זאת בעזרת שיבוץ המפתחות למערך בגודל n+1.
- Radix Sort
  - מיון יציב לפי ספרת האחדות, עשרות וכו' עד הספרה האחרונה (Most Significant Bit - MSB)
  - אפשר גם רקורסיבית מה-MSB ל-LSB
  - זמן: O(nk) כש-k זה האורך המקסימלי של מספר
- טוב גם למיון לקסיקוגרפי וכדומה

ספר טלפונים



תרגיל 1 להיום

- כיצד ניתן למיין בזמן $O(n)$, n מספרים שלמים בטווח $1..n^2$?
- פתרון:
  - נהפוך כל מספר לבסיס n בזמן $O(n)$. נקבל n מספרים בין 2 ספרות.
  - נפעיל Radix Sort בזמן $O(n)$
- עבור מספרים בתחום $1..n^c$ נמיין בזמן $O(nc)$

4

מיון יציב - היא מיון אשר בו במידה ויש 2 מספרים עם ערך שווה, הם יופיעו אחד ליד השני אך לפי הסדר המקורי.

מספר הספרות של x בבסיס B היא $O(\log_B x)$.

מודל ההשוואות

- הגישה היחידה לקלט היא בעזרת פונקציה Comapre(index i, index j)
- מחזירה בזמן $O(1)$ תשובה בוליאנית
- רוב האלגוריתמים שנלמד עובדים תחת הנחות מודל זה
- יעיל גם למיונים וחיפושים תחת אילוצים שונים

5

תרגיל

- נתונים n מטבעות זהב שאחת מהן מזוייפת (שוקלת פחות).
- כתבו אלגוריתם שמוצא אותה בכמה שפחות שקילות מאזניים (אסימפטוטית) במקרה הגרוע (WC).
- מצאו חסם עליון ותחתון לזמן הריצה של האלגוריתם.
- מצאו חסם עליון ותחתון לכמות השקילות ההכרחיות WC לפיתרון הבעיה.
- הוכיחו שאין אלגוריתם יעיל יותר (אסימפטוטית) WC

6

תשובה 1

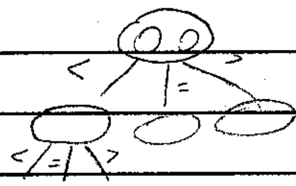
- עבור n זוגי: נשים חצי מהמטבעות בכל מאזן, ונמשיך רקורסיבית עם החצי הקל.
- עבור n אי זוגי: נשים מטבע בצד ואת השאר נחלק לשניים. נמשיך רקורסיבית עם החצי הקל. במקרה של שיווין - המטבע המזוייף הוא המטבע שהוצאנו.

7

חסמים לזמן ריצה של האלגוריתם

- חסם עליון: $T(n)=1+T(n/2)=O(\log n)$
- חסם תחתון: $O(1)$ – למשל, אם המזוייפת היתה המטבע שהוצאנו, באחת מ-$O(1)$ השקילות הראשונות
- חסם הדוק : אין.
- אם n הוא חזקה של 2: נבצע $\Omega(\log n)$ שקילות.
- עבור n כזה יש חסם הדוק לכמות ההשוואות של האלגוריתם: $\Theta(\log n)$

8



חייבים להיות לפחות n עלים כיוון שאחרת
אין עלים לכך שכל המטבע ה K
לעולם לא נגיע.

הוכחת חסמים לזמן ריצה של אלגוריתם

- חסם עליון: הוכחה לזמן ריצה מקסימלי עבור כל קלט. למשל: נוסחאות נסיגה, פונקציית פוטנציאל
- חסם תחתון: קלט לדוגמא עבורו האלגוריתם לוקח הרבה זמן.
- נניח שהוכחנו שאלגוריתם מסויים פותר את הבעיה לכל קלט בזמן $O(t)$
- אזי $O(t)$ הוא חסם עליון לזמן הנדרש לפתור את הבעיה WC

9

מצאנו חסם עליון לבעיה

- הוכחנו ש:

קיים אלגוריתם שפותר את הבעיה לכל קלט בגודל n, בזמן $O(\log n)$.

כדי למצוא חסם תחתון לבעיה יש להוכיח ש:

לכל אלגוריתם שפותר את הבעיה קיים קלט שעבורו זמן הריצה הוא $\Omega(\log n)$.

10

אבחנות

- לעץ עם n עלים ומס' בנים קבוע לכל צומת, יש גובה $\Omega(\log n)$
  - כי: בסידרה הנדסית עם מנה קבועה, שבה האיבר הראשון הוא 1 והסכום הוא n, יש $\Omega(\log n)$ איברים.
- כל אלגוריתם שפותר את הבעיה ניתן להצגה כעץ החלטות
- בעץ יש לפחות n עלים, שמתאימים ל-n פלטים אפשריים
- לכל צומת בעץ ההחלטות יש מס' קבוע של בנים

11

מצאנו חסם תחתון לבעיה

- לכל אלגוריתם שפותר את הבעיה קיים קלט שעבורו זמן הריצה הוא $\Omega(\log n)$
- מצאנו חסם עליון לפתרון הבעיה, וכיוון שהוא שווה לחסם התחתון – זהו פיתרון הדוקי.
- זה לא אומר שאין אלגוריתם שפותר את הבעיה בזמן $O(1)$ עבור קלטים מסויימים

12

חסמי זמן ריצה לבעיה

- חסם עליון $O(t)$: אלגוריתם לדוגמא שפותר את הבעיה לכל קלט בזמן $O(t)$
- חסם תחתון $\Omega(t)$:
  - מיפוי כל אלגוריתם לעץ החלטות בגובה $\Omega(t)$
  - רדוקציה לבעיה שהוכחנו בכיתה עם חסם תחתון על זמן הריצה (מיון או חיפוש בינארי במודל ההשוואות).
  - זמן קריאת הקלט. למשל: כל אלגוריתם לבעיית מציאת המינימום מ-n מספרים חייב לעבור על כל הקלט ולכן לוקח זמן $\Omega(n)$

13

חסם תחתון על חיפוש במודל ההשוואות

- מה החסם התחתון על חיפוש במערך ממוין?
  - במקרה הגרוע ביותר נשווה לכל איבר
  - ז"א בעץ יש n עלים
  - מה העומק המינימלי של עץ בינארי בעל n עלים?
    - זמן לוגריתמי $O(\log n)$

14

מס

פתק למבחן

- נניח שקיים אלגוריתם שפותר את הבעיה בזמן $O(f(n))$.
- נציג אלגוריתם A שמוצא מפתח נתון במערך ממוין בגודל n בזמן WC $o(\log n)$ במודל ההשוואות.
- זוהי סתירה למה שהוכחנו בכיתה. לכן לא קיים אלגוריתם שפותר את הבעיה בזמן WC $O(f(n))$. מ.ש.ל.
- האלגוריתם A:
  - קלט: מערך של n מספרים ממוינים ומפתח x שקיים במערך.
  - פלט: אינדקס המפתח x במערך
  - מימוש: ...

15

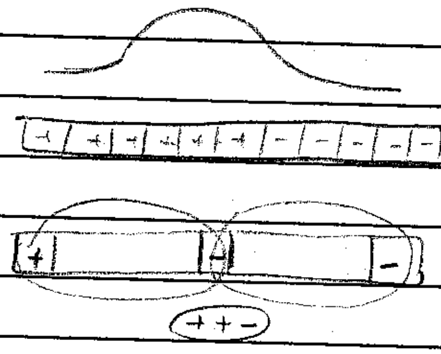
- נתונים n אלמנטים מתחום סדור במערך A. ידוע כי קיים אינדקס j כך שלכל i≤j מתקיים A[i]<A[i+1] ולכל i>j מתקיים A[i+1]<A[i].

דוגמה: j=6

1	2	4	5	6	9	26	15	14	3

תארו אלגוריתם יעיל למציאת j.

16



⊛ הסיבה שבזמן החיפוש חייב היה להיות logn
הוא בגלל הסבר של חיפוש
כמוך השאלה.

- נתאר אלגוריתם הפועל בסיבוכיות O(logn).
- האלגוריתם מאוד דומה לחיפוש בינארי במערך ממוין.
- נסתכל על האבר ה-n/2. אם זהו האבר המתאים אזי סיימנו (גדול משני שכניו), אחרת אם הוא גדול משכנו הימני וקטן משכנו השמאלי, נחזור על הפעולה בחצי השמאלי של המערך רקורסיבית (על n/2 אברים).
- אחרת נחזור על הפעולה בחצי הימני של המערך רקורסיבית.

17

- בכל שלב אנו מקטינים את מס' האברים הנבדקים פי 2, ולכן כפי שראינו פעמים רבות הסיבוכיות לוגריתמית.
- טעויות נפוצות: לחפש בכל שלב את החציון ולא את התא האמצעי. חציון ניתן למצוא רק בזמן לינארי ולכן זה לא מתאים!!

18

- הוכיחו כי $\Omega(\log n)$ הוא חסם תחתון על מספר ההשוואות הנדרש למציאת j.
- אופציה א': נבנה עץ השוואות, סה"כ יש n אפשרויות עבור המיקום של j, ולכן בעץ צריכים להיות לפחות n עלים. בכל צומת בעץ ההשוואות אנו יכולים להשוות רק בין שני אברים ולכן אם העץ מאוזן עומקו הוא לוגריתמי

19

- אופציה ב': רדוקציה לחיפוש בינארי. נניח כי ניתן למצוא את j בזמן $o(\log n)$ (ממש קטן מ-$\log n$). אזי בהינתן מערך ממוין A ואבר x אותו אנו מחפשים נפעיל אלגוריתם למציאת j, כאשר נתייחס לכל אבר גדול מ-x כאל מספר שלילי. כך לדוגמה נתייחס למערך 1,3,5,7,9,15 כאשר x=7 כאל המערך -1,3,5,7,-9,-15. כעת נפעיל את האלג' ונקבל את המיקום של x. אך אנו יודעים שהחסם התחתון על חיפוש בינארי הינו לוגריתמי, וזאת בסתירה להנחה.

20

- הערות בדיקה: היו כאן כמה טעויות חוזרות
- הוכיחו שהאלגוריתם בסעיף א' עובד בזמן לוגריתמי במקרה הגרוע ביותר (נוסחת רקורסיה). זה היה צריך להיכתב בסעיף א' ולא כאן.
- ניסו לעשות רדוקציה למיון של מערך. זה התחלק לשתי טעויות עיקריות
  - הניחו שA מערך כמו שמתואר והראו שניתן למיינו בזמן ממש קטן מ- ($n\log n$), אבל מערך כמו שמתואר ניתן למיין בקלות בזמן לינארי (למצוא את j ואז למזג את שני תתי המערכים)
  - הניחו שA מערך כללי, אבל אז לא ניתן להפעיל את הפונקציה, היא מסתמכת על כך שלA מבנה מאוד מסוים!

21

הסוף

שבוע נעים

22