12.05.08 ①

מבנה נתונים - שיעור 2

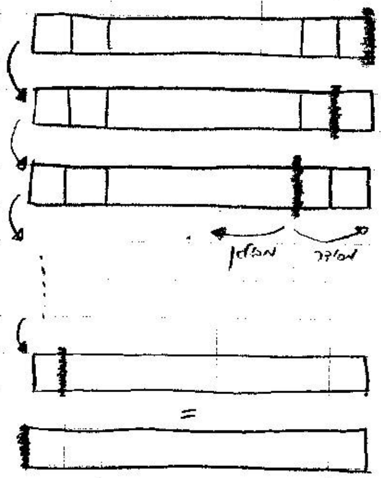
Selection Sort

$\theta(n-1)$ (למעשה בין $2(n-1)$ ל- $3(n-1)$)

$\theta(n-2)$ (למעשה בין $2(n-2)$ ל- $3(n-2)$)

$\theta(n-3)$

$\theta(1)$ (למעשה בין $2\cdot 1$ ל- $3\cdot 1$)



נסמן ב $T_i(n)$ את מספר הפעולות שמתבצעות בשלב ה-i של Selection sort כאשר ממיינים מערך בין n איברים.

$$2(n-i) \le T_i(n) \le 3(n-i) \Rightarrow T_i(n)=\theta(n-i)$$

לפיכך, מספר הפעולות הכולל ושמתבצע במהלך Selection Sort עבור מערך בין n איברים הוא

$$2\left((n-1)+(n-2)+\dots+1\right) \le T_1(n)+T_2(n)+\dots+T_{n-1}(n) \le 3\left((n-1)+(n-2)+\dots+1\right)$$

$$2\,\frac{(n-1)n}{2} \le \qquad\qquad \le 3\,\frac{(n-1)n}{2}$$

ולפיכך מספר הפעולות הכולל הוא $\theta\left(\frac{(n-1)n}{2}\right)$ (כלומר $\theta(n^2)$)

באופן כללי יותר, מתקיים שאם $g_1=\theta(f_1)$, $g_2=\theta(f_2)$, $g_3=\theta(f_3)$ אז $g_1+g_2+g_3=\theta(f_1+f_2+f_3)$.

<u>הסבר:</u> קיימים קבועים חיוביים $\underline{c_1}, \overline{c_1}, \underline{c_2}, \overline{c_2}, \underline{c_3}, \overline{c_3}$ כך שלכל n:

$$+\begin{cases} \underline{c_1}\, f_1(n) \le g_1(n) \le \overline{c_1}\, f_1(n) \\ \underline{c_2}\, f_2(n) \le g_2(n) \le \overline{c_2}\, f_2(n) \\ \underline{c_3}\, f_3(n) \le g_3(n) \le \overline{c_3}\, f_3(n) \end{cases}$$

$$\min\{\underline{c_1},\underline{c_2},\underline{c_3}\}\left(f_1(n)+f_2(n)+f_3(n)\right) \le g_1(n)+g_2(n)+g_3(n) \le \max\{\overline{c_1},\overline{c_2},\overline{c_3}\}\left(f_1(n),f_2(n),f_3(n)\right)$$

Insertion sort

<u>תאור מילולי:</u> בכל שלב מציבים את האיבר הימני ביותר בחלק הלא ממוין ימינה עד למקומו הנכון (בינארית) בחלק הממוין.

④ בדוגמת שיטת מיון זה, נוכל לספר את הבדלים עבור <u>קלטים מסוימים</u>. בכל מקרה הסיבוכיות תהיה $\theta(n^2)$. את החיפוש ניתן לעשות גם באמצעות חיפוש בינארי. אך לא יהיה שיפור משמעותי במצב הזה בגלל הצורך להזיז את האיברים,

מבנה נתונים - שיעור 2

פסאודו קוד

```
Insertion Sort (A[0... n-1])
    For i ← n-2 downto 0
        do  j ← i+1
            while A[j] < A[j-1] and j ≤ n-1
                do  exchange A[j] ↔ A[j-1]
                    j ← j+1
```

וריאציות

I לבצע את החיפוש של המקום הנכון (הזמני) בחלק הממוין של המערך באמצעות חיפוש בינארי.

II במקום למקם את האיבר החדש במקומו הנכון ע"י סדרת החלפות של איברים סמוכים (פעפוע), ראשית למצוא את המקום החדש ואז להזיז את מי שנדרש צעד אחד שמאלה.

פסאודו קוד של וריאציה II

```
For i ← m-2 downto 0
    do  temp ← A[i]
        j ← i+1
        while A[j] < temp and j ≤ n-1
            do  A[j-1] ← A[j]
                j ← j+1
        A[j-1] ← temp
```

ניתוח יעילות

באינטרציה ה-i של לולאת ה For תתבצענה בין 0 ל- $n-i$ איטרציות של לולאת ה while, בכל אחת מהן $\Theta(1)$ פעולות.

לכן באינטרציה ה i של לולאת ה For תתבצענה $O(n-i)$ פעולות. ובסך הכל $O(n^2)$ פעולות.

מספר הפעולות ב Selection תמיד $\Theta(n^2)$ מספר הפעולות ב Insertion תמיד $O(n^2)$ (ועבור קלטים מסויימים ממש פחות).

כמה פעולות השוואה בלבד מתבצעות?

ב Selection בדיוק $\frac{n(n-1)}{2}$ (כלומר $\Theta(n^2)$)

ב Insertion בגירסאות שניתנו (המקורית ו-II) - המקרה הגרוע ביותר $\frac{n(n-1)}{2}$ [illegible]

אבל, בוריאציה I (עם שימוש בחיפוש בינארי) מספר ההשוואות יהיה

$$\lceil \log_2 1 \rceil + \lceil \log_2 2 \rceil + \lceil \log_2 3 \rceil + \dots + \lceil \log_2 n-1 \rceil \le (n-1) \cdot \lceil \log_2 (n-1) \rceil$$

(למרות שמספר הפעולות הכולל עשוי להגיע לסדר גודל של n^2).

<u>טענה</u> כל אלגוריתם מיון, לצורך מיון מערך בן n איברים חייבת $\Omega(n\log_2 n)$ פעולות השוואה (הסתייגות מסויימת שנסביר בהמשך). במקרה הגרוע ביותר.

⊛ כלומר עבור כל אחד מהאלגוריתמים, במקרה הגרוע ביותר של החישוב באלגוריתם זה יבוצעו לפחות $n\log_2 n$ פעולות השוואה, גם עבור האלגוריתם הטוב ביותר.

<u>הוכחה:</u> נתבונן באלגוריתם מיון כלשהו שהתחייבו שהוא מבצע לכל היותר k פעולות השוואה. כאשר הוא מטפל במיון מערך בגודל n מספר "תסריטי" הריצה של האלגוריתם הוא לכל היותר 2^k. ⊛ (אחרי כל השוואה יש 2 אפשרויות - או להציג או לא להציג).
כדי שהאלגוריתם אכן יוכל למיין <u>כל</u> מערך בגודל n הכרחי שיהיו לו לפחות $n!$ תסריטי ריצה שונים (כדי להתמודד עם $n!$ הסדרים השונים של n מספרים שונים).

אם כן בהכרח $2^k \geq n!$

$$k \geq \log_2 n! = \log_2 1 + \log_2 2 + \dots + \log_2 n \geq \log_2 \tfrac{n}{2} + \log_2 \left(\tfrac{n}{2}+1\right) + \dots + \log_2 n \geq$$

$$\geq \frac{n}{2}\log\frac{n}{2} = \frac{n}{2}(\log_2 n - 1) \geq \frac{n}{2}\cdot\frac{1}{2}\log_2 n = \frac{1}{4} n\log_2 n$$

ל $n \geq 4$

<u>הערות:</u>

1. כמסקנה מהטענה נקבל שמספר הפעולות <u>הכולל</u> (לא רק השוואות) במקרה הגרוע של כל אלגוריתם מיון הוא $\Omega(n\log n)$.
2. אכן יש אלגוריתמי מיון (שנראה בשיעור הבא) שמספר הפעולות הכולל שלהם במקרה הגרוע הוא $\Theta(n\log n)$.
3. אם האלגוריתם לא אמור לטפל במערך כללי של n מספרים ממשיים אלא רק במערכים של n מספרים שידוע עליהם מידע נוסף, ניתן "לשבור" את מחסום ה $n\log_2 n$.