17.11.16

אלגוריתמים - הרצאה 1

ספר הקורס: ←
Introduction to Algorithms

85% מבחן, 15% תרגילי בית.

moshe@cs.biu.ac.il

הפרד ומשול.

סוג פתרון בעיות כאשר לוקחים בעיה גדולה, מחלקים אותה לתתי בעיות קטנות, פותרים כל בעיה בנפרד ומאחדים לפתרון הגדול.

יתרונות וחסרונות:

← אין ספק שבעקביות הפרד ומשול בסופו של דבר הבעיה תיפתר.

← מצד שני יכול להיות שזה לא הפתרון הכי טוב מבחינת זמני ריצה, ניהול זיכרון וכו'.

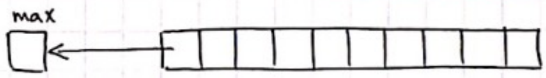
בעיית המקסימום:

קלט: מערך מספרים A באורך n.

פלט: האיבר המקסימלי.

מטרה: מס' השוואות מינימלי, כלומר, נחפש ונמצא את האיבר במינימום השוואות.

הצעות לאלגוריתמים:

1)


נבדוק איבר איבר ונשווה מול איבר מקסימום.
כלומר, נכניס את האיבר הראשון לתוך max ונעבור איבר איבר ונשווה מולו.

מס' ההשוואות יהיה n-1.

← במקרה הזה n-1 זהו החסם העליון והתחתון שלנו כי כל איבר חייב להשתתף לפחות בהשוואה אחת.

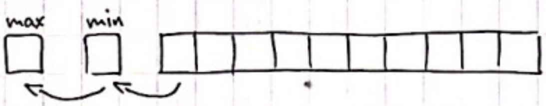
נסתכל על בעיה דומה כאשר צריך למצוא גם מינימום וגם מקסימום.

הצעות לאלגוריתמים:

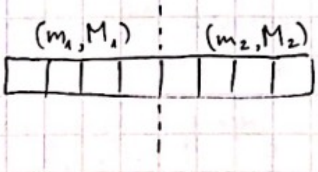
1) נריץ אלגוריתם שיחפש את המינימום ואלגוריתם שיחפש את המקסימום.

מס' ההשוואות יהיה $2(n-1) = 2n-2$.

2) נעבור עם כל איבר השוואה של בדיקה האם הוא מינימום ורק אם לא נבדוק אם הוא מקסימום.



באלגוריתם הזה גם במקרה הגרוע שהסדרה בסדר עולה לדוגמא נגיע ל- $2n-2$ השוואות.

3) שיטת הפרד ומשול.


נמצא לכל חצי מערך את המקסימלי והמינימלי ונמצא בגדול את:

$m \leftarrow \min(m_1,m_2)$

$M \leftarrow \max(M_1,M_2)$

נכתוב את הקוד בצורה רקורסיבית:

```
minmax(i,j)
    if(j=i+1):                      ⟶  תנאי עצירה.
                                        (מערך באורך 2)
        if A[j] > A[i]:
            return (A[i], A[j])
        if A[i] > A[j]:
            return (A[j], A[i])
    else:
        k ← ⌊(i+j)/2⌋
        (m1, M1) ← minmax(i,k)
        (m2, M2) ← minmax(k+1, j)
        return (min(m1,m2), max(M1,M2))
```

$T(n)$ = מס ההשוואות של minmax לקלט באורך n.

$$T(n) = \begin{cases} 1 & , n=2 \\ 2T(\frac{n}{2})+2 & , n>2 \end{cases}$$

$$T(n) = 2T(\tfrac{n}{2})+2 = 2(2T(\tfrac{n}{4})+2)+2 = 4T(\tfrac{n}{4})+6 = 4T(\tfrac{n}{4})+4+2 =$$

$$8T(\tfrac{n}{8})+8+4+2 = \dots = \boxed{2^i T(\tfrac{n}{2^i}) + \sum_{k=1}^{i} 2^k}$$

אנו טוענים בקצב שבה $T(n)$ ראינו שבה עבור ל- $i=1,2,3$ נוכיח באינדוקציה.

נניח שבה מתקיים ל- i ונראה שבה מתקיים ל- $i+1$. בקצב נפתח שוב את $T(\frac{n}{2^i})$

$$2^i\left(2T(\tfrac{n}{2^{i+1}})+2\right) + \sum_{k=1}^{i} 2^k = 2^{i+1}T(\tfrac{n}{2^{i+1}}) + \sum_{k=1}^{i+1} 2^k$$

ונראה שזו בדיוק המשוואה אם היינו מציבים $i+1$.

נרצה ש- $\frac{n}{2^i}=2$ כי אז נקבל *הקבועה: $T(\frac{n}{2^i}) = T(2) = 1$

$$i = \log(n)-1 \iff \log(n) = i+1 \iff n = 2^{i+1} \iff \frac{n}{2^i} = 2$$

נציב: $i=\log(n)-1$ (בקונסם)

$$T(n) = 2^i T(\tfrac{n}{2^i}) + \sum_{k=1}^{i} 2^k = 2^{\log(n)-1} T(\tfrac{n}{2^{\log(n)-1}}) + \sum_{k=1}^{\log(n)-1} 2^k = 2^{\log(n)-1} T(2) + \sum_{k=1}^{\log(n)-1} 2^k$$

$$\boxed{2^{\log(n)-1} = \frac{2^{\log(n)}}{2} = \frac{n}{2}}$$

המשך ←

17.11.16 בס"ד

$$2^{\log(n)-1}\cdot \underbrace{T(2)}_{1} + \sum_{k=1}^{\log(n)-1} 2^k = \frac{n}{2}\cdot 1 + n-2 = 1.5n-2$$

ניתן לראות שזהו סכום
של סדרה הנדסית:

$$S = 2+4+8+\dots+\frac{n}{2} \Rightarrow 2S = 4+8+\dots+\frac{n}{2}+n \Rightarrow S = 2S-S = n-2$$

לסיכום, ראינו שבאלגוריתם של הפרד ומשול התקבל זמן ריצה של $1.5n-2$ במקום $2n-2$, אם היינו מגדירים בסדרי גודל זה לא היה משנה אך רצינו למצוא מס' השוואות מדויק.

$\boxed{\text{מיון מיזוג:}}$

תזכורת: - ממיינים חצי ראשון של מערך, ממיינים חצי שני בנפרד. ($2T(\frac{n}{2})$)
- ממזגים את שני החצאים. (סדר גודל של n)

נחשב זמן ריצה: (אם השוואות עדיין המדד שלנו)

$$T(n) = \begin{cases} 0, & n=1 \\ 2T(\frac{n}{2})+n, & n>1 \end{cases}$$

$$T(n) = 2T(\tfrac{n}{2})+n = 2(2T(\tfrac{n}{4})+\tfrac{n}{2})+n = 4T(\tfrac{n}{4})+2n = \dots = 2^i\,T(\tfrac{n}{2^i}) + \underbrace{2^{i-1}n}$$

ניחשנו את $T(n)$,
נבדוק עכשיו עם $i+1$:

$$2^i\left(2T(\tfrac{n}{2^{i+1}})+\tfrac{n}{2^i}\right)+2^{i-1}n = 2^{i+1}\,T(\tfrac{n}{2^{i+1}}) + \underbrace{n+2^{i-1}n}$$

ניתן לראות שהנוסחא לא מסתדרת ולכן הניחוש לא היה נכון.

ננסה שוב:

$$T(n) = 2T(\tfrac{n}{2})+n = 4T(\tfrac{n}{4})+2n = 4(2T(\tfrac{n}{8})+\tfrac{n}{4})+2n = 8T(\tfrac{n}{8})+n+2n = 8T(\tfrac{n}{8})+3n = \dots$$

$$\boxed{\dots = 2^i\,T(\tfrac{n}{2^i})+in}$$

ניחוש שוב.
נבדוק אם הניחוש מסתדר עם $i+1$:

$$2^i\left(2T(\tfrac{n}{2^{i+1}})+\tfrac{n}{2^i}\right)+in = 2^{i+1}\,T(\tfrac{n}{2^{i+1}})+n+in = 2^{i+1}\,T(\tfrac{n}{2^{i+1}})+(i+1)n$$

ניתן לראות שהניחוש הנ"ל נכון כי כאשר פיתחנו את i עוד צעד אחד קיבלנו תוצאה זהה אם היינו מגדירים $i+1$ בנוסחא.

נרצה $\frac{n}{2^i}=1$ כי אז $T(\frac{n}{2^i})=T(1)=0$: $\frac{n}{2^i}=1 \iff n=2^i \iff i=\log(n)$

נציב: $i=\log(n)$

$$T(n) = 2^{\log(n)}\,T(1)+n\log(n) = n\log(n)$$

מיון מהיר:

הגדרה: לוקחים את האיבר הראשון למערך ומוצאים את מקומו כאשר כל אלה שקטנים ממנו נמצאים משמאלו וכל אלה שגדולים ממנו מימינו.
ממשיכים בצורה רקורסיבית עד שממיינים ושמים במקום את כל האיברים.

worst case: מערך ממוין, לוקחים את האיבר הראשון והוא "ישאר" למקומו לאחר בדיקה. ואז יוצא שבצורה הרקורסיבית לא הקטנו את המערך בצורה משמעותית אלא כל פעם הורדנו איבר אחד. ⟸ סדר גודל של n^2, כי:

$$T(n)=T(n-1)+n-1=(n-1)+(n-2)+T(n-2)=(n-1)\overset{+}{}(n-2)+\cdots+2+1=\Theta(n^2)$$

best case: אם כל פעם ה- pivot נמצא בדיוק באמצע המערך.

$$T(n)=2T(\tfrac{n}{2})+(n-1)=\Theta(n\log(n))$$

בעיית מכפלות מספרים ארוכים:

קלט: 2 מחרוזות בינאריות המייצגות מספרים. x,y (באורך n ביטים)

פלט: $x\cdot y$

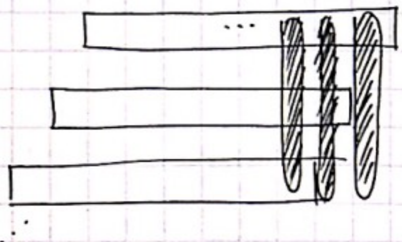
$x = $ 10111 ⋯

$y = $ 10010 ⋯

מדד: פעולות אריתמטיות בסיסיות, כמו כפל, חיבור וחיסור.

הצעות אלגוריתמיים:

1) כפל ארוך.



קיבלנו $O(n^2)$ פעולות אריתמטיות.

2) נחלק את המחרוזת לשני חלקים.

X [X_1 | X_2] ; Y [Y_1 | Y_2]

$X=X_1\cdot 2^{\frac{n}{2}}+X_2 \qquad Y=Y_1\cdot 2^{\frac{n}{2}}+Y_2$

$$X\cdot Y=(X_1\cdot 2^{\frac{n}{2}}+X_2)(Y_1\cdot 2^{\frac{n}{2}}+Y_2)=X_1 2^{\frac{n}{2}}\cdot Y_1\cdot 2^{n/2}+X_1\cdot 2^{\frac{n}{2}}\cdot Y_2+X_2\cdot Y_1\cdot 2^{n/2}+X_2Y_2=$$

$$=\underbrace{X_1Y_1}\,2^n+\underbrace{X_1Y_2}\cdot 2^{n/2}+\underbrace{X_2\cdot Y_1}\cdot 2^{n/2}+\underbrace{X_2Y_2}$$

$$\boxed{\begin{array}{ll} z^{(1)}\leftarrow X_1\cdot X_1 & ;\ z^{(3)}\leftarrow X_2Y_1 \\ z^{(2)}\leftarrow X_1\cdot Y_2 & \ \ \ z^{(4)}\leftarrow X_2Y_2 \end{array}} ; \boxed{\begin{array}{ll} z^{(5)}\leftarrow z^{(1)}\cdot 2^n & ;\ z^{(7)}\leftarrow z^{(3)}\cdot 2^{n/2} \\ z^{(6)}\leftarrow z^{(2)}\cdot 2^{n/2} & \end{array}}$$

$\downarrow T(\frac{n}{2})$ $\qquad\qquad \downarrow O(n)$

return $(z^{(5)}+z^{(6)}+z^{(7)}+z^{(4)})$

בס"ד.

17.11.16

לסיכום נראה מהו $T(n)$ במקרה זה:

$$T(n)=\begin{cases}1 & , n=1\\ 4T(\frac{n}{2})+cn & , n>1\end{cases}$$

$$T(n)=4T(\tfrac{n}{2})+cn=4(4T(\tfrac{n}{4})+c(\tfrac{n}{2}))+cn=4^2T(\tfrac{n}{4})+2cn+cn=4^2(4T(\tfrac{n}{8})+\tfrac{cn}{4})+2cn+cn=$$

$$=4^3(T(\tfrac{n}{2^3}))+cn(4+2+1)=\dots=\boxed{4^iT(\tfrac{n}{2^i})+cn\sum_{k=0}^{i-1}2^k}$$

ניחוש של i, נבדוק $i+1$:

$$=4^i(4T(\tfrac{n}{2^{i+1}})+\tfrac{cn}{2^i})+cn\sum_{k=0}^{i-1}2^k=4^{i+1}T(\tfrac{n}{2^{i+1}})+cn\sum_{k=0}^{i}2^k$$

ניתן לראות שהנוסחא נכונה.

$\frac{n}{2^i}=1 \iff i=\log(n)$, נציב ונקבל:

$$T(n)=4^{\log(n)}T(1)+cn\sum_{k=0}^{\log(n)-1}2^k=4^{\log(n)}+cn(n-1)=\Theta(n^2)$$

$$4^{\log(n)}=(2^2)^{\log(n)}=2^{2\log(n)}=(2^{\log(n)})^2=n^2$$

כאן קיבלנו סדר גודל של n^2 ולכן שיטת הפרד ומשול במקרה הספציפי הזה לא עזרה לנו.

ננסה להוריד את $4T(\frac{n}{2})$ ל- $3T(\frac{n}{2})$:

נזכיר את החישובים שאיתם הגענו ל- 4:

$Z^{(1)} \leftarrow X_1Y_1$ $\quad$ $Z^{(3)} \leftarrow X_2Y_1$

$Z^{(2)} \leftarrow X_1Y_2$ $\quad$ $Z^{(4)} \leftarrow X_2Y_2$

ננסה לרכז את $Z^{(2)}$ ו- $Z^{(3)}$ ובמקומם נציב:

$$Z^{(8)} \leftarrow \underbrace{\underbrace{(X_1+X_2)}_{O(n)}\underbrace{(Y_1+Y_2)}_{O(n)}}_{T(\frac{n}{2})} \quad (=X_1Y_1+X_1Y_2+X_2Y_1+X_2Y_2)$$

$$Z^{(9)} \leftarrow Z^{(8)}-Z^{(1)}-Z^{(4)} \quad (=X_2Y_1+X_1Y_2)$$

$$Z^{(10)} \leftarrow Z^{(9)}\cdot 2^{n/2}$$

נראה שבכך הוספנו המון פעולות נוספות של חיבור שלא משפיעים לנו על T אך הורדנו 2 פעולות כפל ($Z^{(2)}$ ו- $Z^{(3)}$) והוספנו רק פעולת כפל אחת $Z^{(8)}$.

לכן אולי הגדלנו את הקבוע של n עם כל החיבורים אך הקטנו מ- $4T(\frac{n}{2})$ ל- $3T(\frac{n}{2})$

$$T(n) = 3T(\tfrac{n}{2}) + cn = 3(3T(\tfrac{n}{4}) + \tfrac{cn}{2}) + cn = 3^2T(\tfrac{n}{2^2}) + \tfrac{3cn}{2} + cn = 3^2(3T(\tfrac{n}{8}) + \tfrac{cn}{2^2}) + cn(\tfrac{3}{2}+1) =$$

$$= 3^3T(\tfrac{n}{2^3}) + cn((\tfrac{3}{2})^2 + \tfrac{3}{2} + 1) = \cdots = \boxed{3^i T(\tfrac{n}{2^i}) + cn\sum_{k=0}^{i-1}(\tfrac{3}{2})^k}$$

$$S = 1 + \frac{3}{2} + (\frac{3}{2})^2 + \cdots + (\frac{3}{2})^{\log(n)-1}$$

$$T = \alpha^0 + \alpha^1 + \alpha^2 + \cdots + \alpha^{L-1}$$

$$\alpha T = \alpha^1 + \alpha^2 + \alpha^3 + \cdots + \alpha^L$$

$$\Rightarrow T(\alpha-1) = \alpha T - T = \alpha^L - 1 \Rightarrow \frac{\alpha^L - 1}{\alpha - 1}$$

$$S = \frac{(\frac{3}{2})^{\log(n)} - 1}{(\frac{3}{2}) - 1} = 2\left(\frac{3^{\log(n)}}{2^{\log(n)}} - 1\right) = 2\left(\frac{3^{\log(n)}}{n} - 1\right)$$

$$3^{\log(n)} = (2^{\log(3)})^{\log(n)} = (2^{\log(n)})^{\log(3)} = n^{\log(3)} \Rightarrow \Theta(n^{1.58\ldots})$$

מכפלת מטריצות:

קלט: $A = (\quad)$, $B = (\quad)$ מטריצות ריבועיות $n \times n$

פלט: $C = A \times B$.

__הצעת אלגוריתמים:__

1) חישוב ישיר של מכפלת מטריצה. לכל מיקום ב-C דרוש $O(n)$ פעולות אריתמטיות.
לחישוב המיקום יש n^2 מיקומים ב-C ולכן סה"כ $O(n^3)$

2) נחלק את המטריצה ל-4:

$$\begin{array}{c}A\\ \left(\begin{array}{c|c} a & b \\ \hline c & d\end{array}\right)\end{array} \begin{array}{c}B\\ \left(\begin{array}{c|c} e & f \\ \hline g & h\end{array}\right)\end{array} = \left(\begin{array}{c|c} ae+bg & af+bh \\ \hline ce+dg & cf+dh\end{array}\right) \Rightarrow T(n) = \begin{cases} 1 & n=1 \\ 8T(\frac{n}{2}) + c\cdot n^2 & n>1 \end{cases}$$

$T(n) =$ מס' הפעולות להכפלת 2 מטריצות $n \times n$.

הגענו לשיטת הפרד ומשול גם כן ל- $T(n)=\Theta(n^3)$ לכן שווה, השיטה עובדת
אך לא שיפרנו זמן ריצה.

3) __שטרסן__ - שיטה שמוצאת איך לבצע את המכפלות במקום ב-8 כפולות ב-7 כפולות.
ועם השיטה הזו מגיע לרמה של $O(n^{2.8\ldots})$

(*) היום הזמן הכי טוב שידוע למכפלת מטריצות הוא $O(n^{2.3\ldots})$