מבני נתונים 14

שחר פרץ

26 במאי 2025

SORTING . . . . (1) . . . .

פורמלית, עד עתה לא ראינו חסם תחתון למיון. נראה שהוא $\Theta(n \log n)$. נגדיר את בעית המיון:

קלט: n איברים שהם totally ordered (יחס סדר מלא). **פלט:** אותם איברים לפי הסדר, מהקטן לגדול. אפשר לחשוב על זה אכל תמורה/פרמוטציה.

יש $n!$ אפשרויות למה התמורה בין הקלט לפלט.

יש תכונה של מיון – stable sort (מיון יציב). בעת מיון כזה אם שני איברים שמבחינת הסדר הם שקולים, יש צורך לשמור על הסדר המקורי שלהם בקלט. אנו מכירים כבר מספר מיונים:

bubble sort insertion sort selection sort	heap sort merge sort binary search tree	quick sort
$O(n^2)$	$O(n \log n)$	worst $O(n^2)$ avg. $O(n \log n)$

הנחה: השוואה בין שני איברים לוקחת $O(1)$, וכל האל' הנל תשתמש בפעולה זו.

ראוי לציין שב־quicksort אין קלט "רע", רק בחירות אקראיות רעות. המשמעות של זה היא שאי אפשר להנדס שלט שיגדיל בבת אחת את הסיבוכיות.

שאלה: האם קיימים אל' מהירים יותר?

חסם עליון הראינו בעזרת דוגמה של אלגו', לעומת חסם תחתון שנצטרך להראות ביחס לכל אלגו' שיכול להיתכן. לדוגמה, נוכל לתת חסם תחתון של $O(n)$ משום שחייבים לבחון כל ערך בקלט.

משפט 1. תחת מודל ההשוואות, כל אלג' מיון אץ בזמן $\Omega(n \log n)$ במקרה הגרוע ביותר.

הוכחה. "הוכחה" האלגוריתם יכול אך ורק להשוות. יהיו x_i, x_j כלשהם שהשותי. נתבונן במרחב $n!$ הפרמוטציות, אזי פמעשה פיצלנו אותן ל־$x_i < x_j$ פרמוטציות והמקרים בהם $x_i > x_j$. בגלל שהאלגוריתם מבוסס השוואות, נוכל להמשיך את המיון הזה לפי אותו הרעיון, כלומר נריץ את אחד המקרים. אידיאלית, נרצה לבחור x_i, x_j כך שננחצה בדיוק לשניים את מרחב הפרמטוציות (ספציפית לגבי הזוג הראשון זה יהיה נכון לכל זוג).נמשיך ב"עץ החלטות", כלומר באחד מהמקרים, או $x_i < x_j$ או להפך.

מסתכלים על עץ ההחלטות של האלגו'. בכל השוואה לפי התוצאה (הבינארית), פוצים ללק מהפרטמוטציות. עלה בעץ מייצג פתרון של המיון עאש נשארה פרמוטציה אחת בלבד.

יש לפחות $n1$ עלים בעץ עבור כל תוצאה אפשרית. לכן, העומד לפחות $\Omega(n!)$. כל ריצה של האלג'ו שקולה למעבר שולש ועד עלה. העץ כולו מאוד גדול, אבל לא בהכרח מחזיקים אותו באמת. זהו עץ אך ורק לניתוח. ראינו ש־$\log nn! = \Theta(n \log n)$. ■

אז ניתוח זמן ריצה ב־worst case הוא $O(n \log n)$, עומק עץ החיפוש. זמן הריצה הטוב ביותר הוא $O(n)$ – והוא לעלה עם העומק המינימלי. לדוגמה ב־bubble sort יש ענף כזה בעץ, שמאפשר $O(n)$ לקלט מסודר. זמן ריצה ממוצע שקול לעומק הממוצע של עלה.

נשאלת השאלה, האם ה־average case הוא $\Omega(n \log n)$ גם כן?

משפט 2. על אלג' במודל ההשוואות חייב להיות $\Omega(n \log n)$

הוכחה. באינדוקציה על מספר העלים ℓ.

בסיס $\ell = 1$. יהיו 0 השוואות. עבור ℓ כללי, הוא מתפרק ל־$\ell = \ell_1 + \ell_2$ כך שעץ החיפוש:־

$$\begin{array}{c} \cdot \\ / \quad \backslash \\ \ell_1 \qquad \ell_2 \end{array}$$

כאשר ℓ_1 מייצג תת־עץ עם ℓ_1 עלים. אז העומק הממוצע (נשקלל כי בכל עץ כמות אחרת של עלים):

$$\text{avg. case} \overset{\text{ה.א.}}{\geq} \frac{\ell_1}{\ell} \cdot (1 + \log \ell_1) + \frac{\ell_2}{\ell}(1 + \log \ell_2) = 1 + \frac{\ell_1 + \ell_1 \log \ell_1 + \ell_2 \log \ell_2}{\ell} \overset{(1)}{\geq} 1 + \frac{2 \cdot \frac{\ell_1 + \ell_2}{2} \log\left(\frac{\ell_1 + \ell_2}{2}\right)}{\ell} = 1 + \log \frac{\ell}{2} = \log \ell$$

(1) – ידוע שהפונקציה $\log x$ קמורה. היא מקיימת שקו בין שתי נקודות נמצא מעליה, ומא"ש ינסן נקלב שהנקודה $f\left(\frac{\ell_1+\ell_2}{2}\right)$ קטנה מ-$\frac{f(\ell_1)+f(\ell_2)}{2}$. ■

הערה לעצמי: לבדוק איך אני יכול להגדיר לעצמי משהו יותר הדוק מממוצע חשבוני ביחס לאינדוקציה ההיא.

משפט 3. גם עבור אלגו' לא דטרמיניסטי, מתקיים $\Omega(n\log n)$ זמן מדוסה (תלות).

מסקנה. לא ניתן לבנות עץ חיפוש בינארי על n איברים כלליים בזמן $o(n\log n)$. זוהי רדוקציה לבעיית המיון – אם קיים עץ כזה אזי אפשר בזמן לינארי לעבור עליו ולמצוא מיון.

ניתן לרדת המסיבוכיות של $O(n\log n)$ כאשר יש מידע נוסף על הקלט. לדוגמא, אם אנחנו חסומים ב־r כלשהו, אפשר לספור כמה פעמים הופיע כל איבר. סיבוכיות $O(n+r)$. פרטים לגבי אופן המימוש + פסאדו קוד שהמרצה לשנו לא מראה נמצאים במצגת, וזהו יוצא מיון יציב.

נרצה לשפר עבור המקרה של n איברים עד n^k. נוכל למיין על סמך ספרת האחדות, ואז עשרות, מאיות וכו'. בטרמינולוגיה של הקורס התחלנו בלמיין ב־least significant digit עד ה־most significant digit. דרך המיון תהיה באמצעות count sort כי אנו חסומים ב־k, שהוא מיון יציב (דרוש לתקינות האלגו'). העלות הסופית תהיה $d(b+b)$ כאשר d כמות הספרות ו־b הבסיס שאנו עובדים בו. בפרט עבור n^k חסם עליון נוכל לבחור לעבוד בבסיס n, ונקבל k ספרות, וסה"כ $k(n+n)=O(nk)$. נקרא לכך Radix Sort.

(יואב עירוני לוחש – least significant digit – LSD).

ניסוח של המרצה: עבור n ממספרים עד n^k נבחר $b=n$, ונקבל $d=k$ ולכן $O(kn)$ זמן.

....

שחר פרץ, 2025