

מבני נתונים

תרגיל 7 - פתרונות

גלעד אשרוב

3 ביולי 2014

מיונים

שאלה 1. תארו כיצד ניתן להמיר כל אלגוריתם שממייין מערך A של n מספרים על ידי השוואות בזמן $O(f(n))$, לאלגוריתם שממייין מיון יציב בזמן $O(f(n))$. ניתן להניח ש- $n < f(n)$.

פתרון: הרעיון הוא לעבור פעם אחת על המערך, ולהעתיק אותו למערך חדש שבו לכל איבר נוסף גם את האינדקס במערך המקורי. כלומר, בהינתן מערך $A[1, \dots, n]$, נבנה מערך B כך ש- $B[i] = (A[i], i)$. כעת, נמייין את המערך B , כאשר השוואה בין שני תאים במערך B תעשה באופן הבא: בהינתן $(\alpha, i), (\beta, j)$, נשווה בין α ו- β . באם הם שונים - אנחנו יודעים מי קטן ממי. באם שניהם שווים - נכריע לפי (i, j) . בצורה כזו, יש לנו "הכרעה" בין כל זוג איברים, ולכן המיון הוא בהכרח יציב.

שאלה 2. במיון סלים ההנחה המוקדם הייתה שהמספרים במערך מתפלגים בצורה אחידה על פני הטווח החד מימדי שלהם. בשאלו זו, עליכם למיין ערכים בטווח תלת-מימדי. נתונות n נקודות במרחב $(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)$, כאשר ידוע שכל נקודה מתפלגת באופן אחיד בתוך כדור היחידה (ובפרט, לכל i מתקיים $\sqrt{x_i^2 + y_i^2 + z_i^2} \leq 1$). עליכם למיין את הנקודות לפי מרחקם מראשית הצירים, מהקטן לגדול. תארו אלגוריתם הממייין את הנקודות בזמן ריצה $O(n)$ (תחת הנחת התפלגות אחידה).

סקיצת פתרון: נפתור עבור n נקודות במרחב דו-מימדי $(x_1, y_1), \dots, (x_n, y_n)$. המעבר למרחב תלת מימדי ייעשה בצורה דומה.

המיון שנשתמש בו יהיה מיון סלים *bucket sort*, כאשר אנו יודעים שכל נקודה מתפלגת באופן אחיד בתוך מעגל היחידה. לפיכך, נחלק את המרחב (מעגל היחידה) לחלקים עם שטח זהה. ניתן לחלק לפי שורש היחידה ה- n (כלומר, כמו "פיצה" שמחולקת ל- n חתיכות) - בדקו שהנכם יודעים איך לעשות זאת. אפשרות נוספת, היא לחלק למעגלים וטבעות קטנים סביב הראשית, ונדע לאיזה סל שייך כל איבר ע"י סיווג מרחקו מהראשית. שטחו של המעגל הראשון הוא $r_1^2 \cdot \pi$, של הטבעת השניה היא $(r_2^2 - r_1^2) \cdot \pi$, ושל הטבעת האחרונה $(r_n^2 - r_{n-1}^2) \cdot \pi$. שטחו של כל המעגל הוא $1^2 \cdot \pi$, ולכן שטחו של כל טבעת כזו היא π/n . כאשר נציב במשוואות, נקבל כי $r_i^2 = i/n$, כלומר $r_i = \sqrt{i/n}$. כאשר נקבל איזושהי נקודה (x, y) , נחשב את מרחקה מהראשית $\sqrt{x^2 + y^2}$ ונבדוק באיזה תחום $(\sqrt{\frac{i}{n}}, \sqrt{\frac{i+1}{n}})$ היא נופלת, ונכניס אותה לסל ה- i המתאים.

שאלה 3.

1. תארו כיצד ניתן למיין n מספרים שלמים שכולם מהטווח $[0, \dots, n^2 - 1]$ בזמן $O(n)$.
2. תארו כיצד לכל קבוע c ניתן למיין n מספרים שלמים שכולם מהטווח $[0, \dots, n^c - 1]$ בזמן $O(n)$.

פתרון:

1. נבצע $Radix\ sort$ על בסיס n . נשים לב שכל מספר בטווח $[0, \dots, n^2 - 1]$ ניתן להביע כמספר בעל שתי ספרות בבסיס n (כלומר, כל ספרה היא איזשהו ערך בטווח $[0, \dots, n - 1]$). לדוגמה, המספר $(n - 10, 25) = (n - 10) \cdot n^1 + 25 \cdot n^0 = n^2 - 10n + 25$.

לאחר ההמרה, כאשר המספרים נתונים כזוג ספרות מבסיס n , נבצע $Radix\ sort$. עלות האלגוריתם היא $O(k \cdot (n + d))$, כאשר $k = 2$ ו- $d = n - 1$, ולכן העלות הכוללת היא $O(n)$. בנוסף, ישנה העלות של המרת המספרים לבסיס n (כיצד נעשה זאת וכמה עולה ההמרה?).

2. בדיוק אותו הדבר - נבצע $Radix\ sort$ על בסיס n . כאן נקבל c ספרות בכל מספר (ו- c קבוע). נקבל אם כן עלות $O(n)$. שוב - צריך להתייחס גם להמרת המספרים לבסיס המתאים, ולעלות ההמרה.

אלגוריתם Select

1. שאלה 1. נתון מערך A בעל n מספרים. עלינו לבצע עיבוד מקדים על A כל שלאחר מכן, בהינתן שאילתא של מספר שלם $1 \leq k \leq n$, נוכל להחזיר את k האיברים הקטנים ביותר בזמן $O(k)$.

פתרון: נשים לב שכאשר השאילתא היא על k קטן (כלומר, k קבוע) - אנחנו צריכים להחזיר את האיבר ממש ב- $O(1)$. לעומת זאת, כאשר שואלים על מספר גדול $k \approx n/2$, מותר לנו לענות ב- $O(n)$. לפיכך, נסדר את המערך בצורה הבאה: המספרים הקטנים ביותר יישמרו בתחילת המערך בצורה כמעט ממוינת, והמספרים הגדולים יישמרו בסוף המערך ללא סדר ביניהם. ככל ש- k קטן יותר, כך התחום $A[1, \dots, k]$ יחסית "מסודר" יותר, וככל ש- k גדול יותר - תת המערך $A[1, \dots, k]$ מסודר פחות.

האלגוריתם יעבוד בצורה הבאה: נחפש חציון בעזרת אלגוריתם $select$, ונבצע $partition$ לפיו. נקבל כי בחצי המערך $A[1, \dots, n/2]$ נמצאים האיברים הקטנים במערך, וב- $A[n/2 + 1, \dots, n]$ נמצאים האיברים הגדולים. כאמור, על שאילתא שבה $k > n/2$ מותר לנו לעבוד בזמן $O(n)$, לכן אין שום צורך לסדר את הצד הימני של המערך. לעומת זאת, נמשיך לסדר את החלק השמאלי של המערך - את החלק בו המספרים קטנים. באופן רקורסיבי - בעזרת $select$ נמצא את החציון של המערך $A[1, \dots, n/2]$ ונבצע $partition$ לפיו. נמשיך בצורה רקורסיבית כאשר בכל פעם נמצא חציון של תת המערך, נבצע $partition$ לפיו, ונמשיך רקורסיבית רק על הצד של האיברים הקטנים. נשים לב שבצורה כזו, לכל i מתקיים שבתת המערך $A[1, \dots, 2^i]$ נמצאים 2^i האיברים הקטנים ביותר במערך, ולכן, בהינתן שאילתא k כאשר $2^i \leq k \leq 2^{i+1}$, האיבר k בגודלו נמצא איפשהו בתת המערך $A[2^i, \dots, 2^{i+1}]$, מערך מגודל $O(k)$, וניתן לכן למצוא את האיבר הזה בזמן המבוקש. עלות העבודה המקדימה שבצענו היא:

$$T(n) = T(n/2) + O(n) = \dots = O(n) + O(n/2) + O(n/4) + O(n/8) + \dots = O(n).$$

2. שאלה 2. באלגוריתם $select$ שנלמד בכיתה, חלקנו את האיברים ל- $n/5$ קבוצות בנות חמישה איברים בכל קבוצה. ראינו שבמקרה כזה, "חציון החציונים" ממפה לפחות 30% מהאיברים, ובכל קריאה רקורסיבית נשאר במקרה הגרוע עם 70% מהאיברים.

1. ניח והיינו משנים את האלגוריתם כך שמחלקים ל- $n/3$ קבוצות עם שלושה איברים בכל קבוצה. כמה איברים ממפה כעת חציון החציונים? מהו זמן הריצה של האלגוריתם? הצג נוסחת נסיגה, ופתור אותה.

2. חזור על הסעיף הקודם כאשר מחלקים ל- $n/7$ קבוצות בנות שבעה איברים בכל קבוצה.

פתרון:

1. ניזכר שחציון החציונים גדול מחצי מהחציונים, וכל חציון גדול מחצי מהאיברים בקבוצה בה הוא נמצא. כאשר מחלקים לשלושיות, ישנם $n/3$ שלישיות, ולכן חציון החציונים גדול מ- $n/6$ חציונים. כל אחד כזה תורם עוד איבר אחד בלבד (עוד $n/6$), ונקבל אם כן שחציון החציונים גדול מ- $n/3$ מהאיברים. בצורה דומה, חציון החציונים קטן גם מ- $n/3$ מהאיברים, ונישאר במקרה הגרוע עם $2n/3$ מהאיברים. כלומר, חציון החציונים מנפה 33% מהאיברים, ובמקרה הגרוע נישאר עם 67% מהאיברים (לכאורה, אפילו טוב יותר מחמישיות!).

אבל... ניזכר במציאת חציון החציונים מתוך $n/3$ החציונים עולה $T(n/3)$. לפיכך, נוסחת הנסיגה תראה כך: $T(n) \leq T(n/3) + T(2n/3) + O(n)$ - כאשר $T(n/3)$ היא עלות מציאת חציון החציונים מתוך קבוצה החציונים, $T(2n/3)$ היא הקבוצה עמה נישאר בקריאה הרקורסיבית, ו- $O(n)$ היא עלות מיון השלישיות, וה- $partition$ נוסחא זו הולכת ל- $O(n \log n)$, וזה גרוע יותר מהחלוקה לחמישיות.

2. במקרה זה, יש לנו $n/7$ חציונים כאשר כל חציון גדול מ-3 איברים. לפיכך, חציון החציונים גדול מ- $4n/14 = 2n/7$ מהאיברים, וקטן מ- $2n/7$ מהאיברים. נישאר במקרה הגרוע עם $5n/7$ מהאיברים. נוסחת הנסיגה תהיה $T(n) = T(n/7) + T(5n/7) + O(n)$, שהולכת עדיין ל- $O(n)$. על כל פנים, כנראה שהקבועים גדולים יותר מאשר חלוקה לחמישיות.

ערימה

1. **שאלה 1.** בנו ערימת מקסימום לאיברים הבאים:

6, 5, 4, 2, 7, 8, 9, 1, 10, 54, 45, 65, 76, 25, 86, 123, 97, 22 ,

ע"י הרצת האלגוריתם $buildHeap$. לאחר מכן, בצעו שלוש פעמים את הפעולה $extractMax$. הדפיסו את הערימה המתקבלת לאחר כל שלב.

2. **שאלה 2.** נתונה ערימת מקסימום עם n איברים. תארו אלגוריתם המוצא את k האיברים הגדולים ביותר בערימה. האם הינכם יכולים לתכנן אלגוריתם הרץ בזמן $O(k \log n)$? (שימו לב - o קטן).

פתרון: ניתן לבנות אלגוריתם טוב יותר מאשר להוציא k פעמים את האיבר הגדול ביותר. נשים לב שהאיבר הגדול ביותר נמצא בשורש ה- $heap$. האיבר המקסימלי הבא - הוא אחד משני בניו. כלומר, לאיבר ה- 1 בגודלו יש שתי מקומות אפשריים. לאיבר ה- 2 בגודלו יש 3 מקומות אפשריים - האיבר שנותר מהאיטרציה הקודמת, ושני בניו של האיבר שיצא מהאיטרציה הקודמת. ממשיכים כך בצורה רקורסיבית. כלומר, נבנה $heap$ נוסף שיישמור את האיברים הפוטנציאליים. בהתחלה, ל- $heap$ זה נכניס את השורש של הערימה המקורית. בכל שלב - נוציא את האיבר המקסימלי מה- $heap$ ונכניס במקומות את שני בניו בערימה המקורית. בצורה כזו, בכל איטרציה יוצא איבר אחד (בעלות $O(\log k)$), ונכניס במקומו זוג איברים. נמשיך כך k איטרציות, ונקבל את k האיברים הגדולים בערימה בעלות $O(k \log k)$.