מבני נתונים 6

שחר פרץ

23 באפריל 2025

1 תרגול?

מרצה: הפסיגולוג מליניארית 1א

עצים לא תמדי אצייר, תסתכלו במצגת של התרגול

אז מה זה עץ AVL?

- עץ חיפוש בינארי מאוזן
- הפקש הבהים לכל היותר 1

יש לו פעולות המאפשרות לממש Dict ADT. כל הפעולות כגבוה העץ – $O(\log n)$. המתרגל ממליץ לחשוב על הקו הכחול בעץ בתרגול. הסיבה: יש הרבה תזוזות וזה מבלבל. בקינטקס מלא, שני העצמים בקו הזה יגדירו את `Rotate(x, y)`.

נדבר על פעולה של `Select-All` שמחזירה מערך ממויין של m המפתחות הקטנים ביותר. נראה שאפשר להגדיר אות בסיבוכיות של $O(m)$. נבחין כי הבעיה היא בהגעה למינימלי כי משם אפשר לעשות מעבר in-order, וזה בעייתי רק כאשר $O(m+\log n) \neq O(m)$ כלומר $m < \log n$. נדבר על שני רעיונות:

- נוסיף מערך של $\log n$ איברים בנבה, ו־Insert ו־Delete יתאחלו כל פעם מערך חדש באורך $\log n$ בלי לפגוע בסיבוכיות. נבצע $\log n$ קריאות ל־successor מהמינימום על מנת לשים במערך את $\log n$ האיברים הכי קטנים. תוספת זמן הריצה היא $O(\log n)$.

  זה פתרון שפשוט להסביר והסיבוכיות תקינה על אף שהוא לא הכי מהיר.

- הפתרון השני הוא לשמור את אותו הפוינטר למינימו סופשוט משם לעשות successor (מעבר in-order). פרמול במצגת של התרגול. הבעיה העקרית: יותר קשה להראות שזה עומד בדרוש.

  נדבר על דרך ליניארית לבנות עץ AVL באמצעות רשימה ממויינת. חרבושים במצגת. זה די ברור. אפילו יש לי דרך יותר פשוטה מהמצגת להראות שאני באמת פולט AVL.

  "תפול בשקט".

  מיזוג שני עצי AVL לכדי יחיד: הפתרון הפשוט הוא להמיר את שניהם למערכים ממויינים, למזג מערכים ממויינים ואז להמיר לעץ. סה"כ $O(n)$. (יכול להיות שאפשר פחות אבל כנראה שלא, המתרגל לא בטוח).

  לסיכום – קל לעבור על עצי AVL. אפשר לעבור בין מערכים ממוייניס לעצים ולהיפך בזמן ליניארי.

  נתונים k מערכים ממויינים בגודל n ונרצה להמיר אותם לעץ של כולם. נוכל להפוך אותם למערך ממויין. נמזג בזוגות (קצת כמו האפמן), נצטרך $\log k$ איטרציות שיארכו $O(n)$ (מיזוג שני מערכים) וסה"כ ליניארי לעץ, כלומר $O(n \log k)$.

  באופן כללי כשמשתמשים במילון:

  - לוודא שהמפתחות שונים.
  - לציין באופן מפורש מי המפתחות ומי הערכים.

2 הרצאה

נדבר על שתי פעולות: `Select(D, k)` שמחזיר את המספר הכי הקטן מספר k ("hth smallest") במילון D, ו־`Rank(D, x)` שמחזיר את המיקום של x במילון מסודר. (הפעולות הפוכות אחד לשני). נפתור זאת באמצעות הרחבה ל־AVL שתקרא rank trees.

שאלות: איך? באיזו סיבוכיות? האם כל אחד מאותו הזמן?

רעיונות: חיפוש בינארי, שמירת אורך תתי העצים של כל צומת, שדה נוסף, רשימה מקושרת (עדכון עם successor).

נדבר על הרעיון של לשמור את השדה של גודל העצים מימין ומשמאל: איך נתחזק שדה כזה? נבין שצריך לעבור פעולה פעולה, בפרט בעבור כל סוג סיבוב אפשרי, ולהראות שאפשר לחשב ב־$O(\log n)$ את השדה. נסכם את השינויים:

- `Insert`: קודם: הוספנו רגיל. עכשיו: גם להוסיף 1 לכל צומת שעוברים בו. תיקונים ל־AVL: לטפל בגלגולים. ,
- כאן פחות או יותר נגמר הדיון בעפ

2.1 פרמול

מופיע במצגת. אין לי באמת כוח להעתיק. קוראים ל־AVL כזה rank tree. השורה התחתונה: השיטה שבה מחפשים משהו שעושה משהו: חיפשמנו מבנה נתונים ידוע (AVL), חשבנו איזה שינויים אפשר לעשות, איך ממשים, ואיך זה משפיע על דברים אחרים (או משהו כזה, תבדקו במצגת מה שיטת העבודה המומלצת).

נבחין שנוסף על כמות צמתים, אפשר לשמור גם גובה של כל תת־עץ. עומק אי־אפשר (כי בגלגול כל העומקים של תתי־העצים יצטכו להשתנות). שיעור הבא נראה משפט שיראה מתי אפשר לשמור פריט מידע בלי לפגוע בסיבוכיות ומתי לא.

המלצה של המתרגל: לפעמים לנסות להשתמש בדברים כ־black box, לא כל מבנה צריך לפרק. ואם משהו לא ברור כדאי לחזור על חומר.

. . . . . .

שחר פרץ, 2025

קומפל ב־LaTeX וגוצר באמצעות תוכנה חופשית בלבד