מבני נתונים 10

שחר פרץ

14 במאי 2025

מרצה: עמית וינשטיין

לפי המרצה זה שיש הרצאה משולבת עם תרגול זה לא אידיאלי אבל לא ברור לי אם הוא בתחום שלמות או לא.

ביום שני דיברנו על ערימות. עם שנה ב' דיברנו על ערימה בינארית. בנושא הזה יש גם ערימה בינומית וערימת פיבונ'צי. סיכום של ערימות (שעתה אנו חוזרים עליו) מצוי בסיכום הקודם ועל כן לא אחזור עליו.

הטבלה הבאה מתייחסת למבני הנתונים והפעולות הנוספות שראינו בהרצאה הזו:

P. Queue	**Insert**	**Minimum**	**Delete-Min**	**Dec.-Key**	**Delete**	**Meld**	**Init**
AVL tree	$O(\log n)$	$O(1)$	$O(\log n)$	$O(\log n)$	$O(\log n)$		$O(n\log n)$
Binary Stack	$O(\log n)$	$O(1)$	$O(\log n)$	$O(\log n)$	$O(\log n)$	$O(n)$	$O(n)$
W.C Binomial Stack	$O(\log n)$	$O(1)$	$O(\log n)$	$O(\log n)$	$O(\log n)$	$O(\log n)$	
*Amort Binomial Stack**	$O(1)$	$O(1)_{W.C.}$	$O(1)$	$O(1)$	$O(1)$	$O(\log n)$	
Lazy Amortized Binomial Stack	$O(1)_{W.C.}$	$O(1)_{W.C.}$	$O(\log n)$	$O(\log n)$	$O(\log n)$		

$*$ – סיבוכיות amoritzed פר פעולה, כלומר **לא** בעבור רצף פעולות של כל הפעולות במבנה.

אם מצוין amort השאר $W.C.$ ולהפך.

HEAP SORT . . . . (1)

ב־1964 המדמ"חיסטים Williams & Floyd המציאו "מיון ערימה" – Heap Sort. הרעיון? נכניס ל־heap ונעשה delete min עד שהכל יהיה מסודר. באופן ברור זה יוצא $O(n\log n)$. הייתרון, הוא שבניגוד למיון עם AVL/B-tree אין I/O ובניגוד ל־merge sort לא צריך לחבר מערכים ולהקצות חלק ניכר מהזכרון למיון. זה הייתרון המשמעותי.

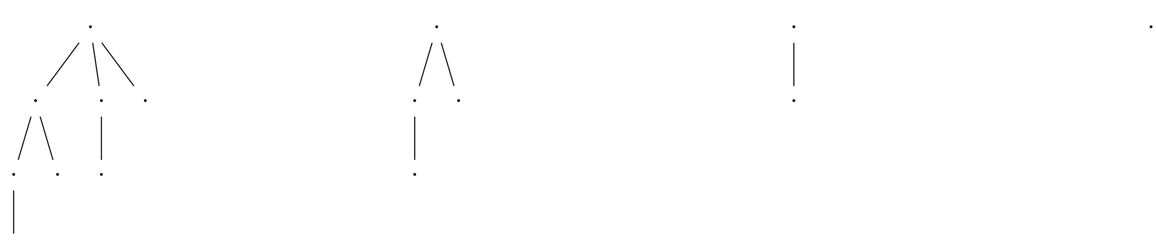
אלג': מכניסים את האיברים לערימה בינארית, ומוצאיים n פעמים לפי הסדר. ניתן למימוש ללא הקצאת זכרון נוסף בצורה יעילה (עם מקדמים קטנים בסיבוכיות). סיבוכיות $O(n\log n)$ במקרה הגרוע ביותר.

הערה: אפשר לממש delete-key ו־decrease-key אחת עם השניה ולכן הן די שקולות. אך יש פעולה שלא דיברנו עליה, היא $\texttt{Meld}(Q_1, Q_2)$, שילוב של שתי ערימות. בשיטה הנוכחית שראינו, שלא ליצור ערימה חדשה ולעשות הכל, לוקח $O(n)$ – כי צריך לכתוב הכל לערימה החדשה.

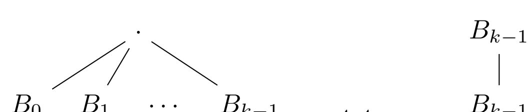
BINOMIAL STACK . . . . (2)

2.1 עצים בינארים

B_3 B_2 B_1 B_0



באופן כללי, B_k יהיה:



תכונותיו:

- לעץ B_k יש 2^k צמתים
- עומק k
- עומק לוגריתמי ביחס לכמות הקודקודים
- יש $\binom{k}{i}$ קודקודים בשכבה ה־i
- לשורש יש k בנים

הוא לא מאוזן בצורה טובה כמו AVL. אך, עדיין העומק הוא $\log n$ כמו ב־AVL.

אזי

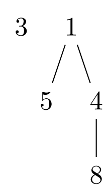
2.2 ערימה בינומית

רעיון של Vuillemin, 1978.

הצעה – נחזיק עץ בינומי בסידור ערימה (מפתות של בנים גדולים ממפתח של אבא). הבעיה, היא שעץ בינומי הוא מגודל 2^k ולכן לכל $k \geq 2$ לא נוכל להוסיף איבר אחד ולשמור על בינומיות.

אז מה עושים? הנ"ל, נחזיק לכל היותר עץ אחד מכל גודל (הרעיון – כל גודל של עץ, מספר טבעי, נוכל לייצג כערימה של $\log k$ עצים בינומים). משום שרוצים להחזיר מינימום ב־$O(1)$, נחזיק מצביע למינימום. נשים לב שיש לכל היותר $\lceil \log_2 n \rceil$ ערימות במבנה, בדיוק לפי הייצגו בינארי של n. העומק המקסימלי הוא גם $\log_2 n$, כי זה הגודל של הערימה המקסימלית בגודלה.

דוגמה.



עץ עם B_0 ו־B_2.

מציאת המינימום – נלך למצביע שמצביע על שורש העץ שנמצא בו המינימום וניקח את ערך השורש. נרצה עתה למחוק איבר מהעץ. הבעיה היא איך נשמור על יחידות גודל עץ בינומי. חיבור של שני עצים בינומים לא יהיה בעייתי, כי זהו כמו חיבור בינארי. לקחנו שני עצים של B_k, ויצרנו עץ חדש של B_{k+1} – ואם יש שארית, נמשיך עם השארית הזו. בזכות כך, בעת מחיקה נוכל להוציא את השורש, ובצע איחוד כמו לעיל ולקבל מחיקה ב־$Oo(\log n)$. הכנסת איבר חדש תתבצע באופן דומה – ניצור את B_0, ונבצע את אותו החיבור בדיוק (כלומר, אם כבר יש B_0 ניצור B_1, ונעביר כל הדרך עד למעלה את ה־carry).

מה שעמית כתב כדי שיהיה מסודר:

- פעולת Meld ניתן למימוש בזמן של $O(\log n)$, ע"י חיקוי של עצים בינארים, ע"י חיקוי חיבור בינארי ואיחוד עצים מגודל זהה לפי הבנייה של עצים בינומיים. העץ עם השורש הגדול יותר יהיה בן חד של שורש עם המפתח הקטן (כלומר חיבור כזה של שני עצים יהיה $O(1)$, וסה"כ worst case של $O(\log n)$ חיבורי עצים). בגלל שחיבורים הם amortized ברצף חיבורים ב־$O(1)$, גם כאן עלות החיבורים תהיה $O(1)$.
- עבור Decrease-Key נעשה headpify-up באותו העץ.
- insert/delete באופן דומה.

2.3 Lazy Binomial Stack

בגלל איך שמונים עובדים, באמורטייז רצף חיבורים או רצף חיסורים/הסרות יהיו ב־$O(1)$. אך נוכל לדאוג גם לרצף של הוספה ו־Delete-Min באמורטייז של $O(1)$. הפתרון הוא ערימה בינומית עצלה. במצב ה"יציב", יהיה רק עץ בינומי אחד מכל גודל. אך נוכל גם פשוט להוסיף את B_0 ולטפל בזה אחכ – איך? אחרי Delete-Min, כשנגיע ל־Minimum, יתכן שיש n ערכים של B_0 וצריך להתחיל לחבר אותם. אז פתאום דברים יכולים להיות יקרים. אבל, amoritized עדיין הכל $O(1)$ ובמקרה שב־2^k נעשה רצף הוספות והחסרות (כמו שראינו בתרגיל בית 2) אז המבנה שלנו לא ישתגע והכל יהיה $\text{amort}(O(1))$.

אופן התיקון של ערימה בינומית עצלה, הוא שיוצרים מערך באורך $\log n$ ואם יש שני עצים שצריך לדחוף לאותו התא, נעביר אותו לתא אחר. נראה $O(1)$ אמורטייז באמצעות חישוב הפוטנציאל הבא:

$$\text{cost} = O(T_0 + T_1 + L)$$

כאשר T_0 עצים בהתחלה, T_1 עצים בסוף ו־L תיקוני העצים. כל איחוד מקטין ב־1 את מספר העצים, כלומר כל איחוד הקטנו את כמות הפוטנציאל ב־1 והעלות שנשארת לנו היא כמות העצים שנשארים בסוף – היא $O(n)$, נחלק ב־n ונקבל אמורטייז $O(1)$. [הערה: במקרה הזה מספר העצים משמש כפונקציית הפוטנציאל]. האיפוס של המערך עצמו יהיה $O(\log n)$ ולכן סה"כ אמורטייז נמחק ב־$O(\log n)$ עבור רצף שכולל חיסורים (טיב הרצף הוא מדוע האמורטייז של הערימה העצלה חזק יותר).

שחר פרץ, 2025

קומפל ב־$\LaTeX$ ונוצר באמצעות תוכנה חופשית בלבד