מבני נתונים 2020 סמסטר א מועד ב

שאלה 1

א. האלגוריתם יחלק את המערך ל-$\frac{n}{k}$ תת מערכים באורך זהה וימיין אותם בסדר עולה/יורד לפי הזוגיות של האינדקס ההתחלתי שלהם. נשים לב שהתת מערך הi הוא $A[(i-1)k, ik)$ ולמיין אותו לוקח (עם מיון $mergesort$) $O(klogk)$ ולכן נקבל סה"כ זמן ריצה של $O\left(\frac{n}{k}klogk\right) = O(nlogk)$.

ב. נשיג חסם תחתון לגובה של עץ ההשוואות של האלגוריתם, נשים לב שמספר העלים בעץ ההשוואות הוא (בכל פעם בוחרים k מפתחות והסדר שלהם נקבע כי הוא מונוטוני) –

$$Num(leaves) = \prod_{i=0}^{\frac{n-k}{k}} \binom{n-ki}{k}$$

אבל היות ולעץ בינארי בגובה h יש לכל היותר 2^h עלים (במקרה והוא עץ מלא) נסיק שהגובה של עץ ההשוואות הוא לכל הפחות $\log(Num(leaves))$ כי לולא היה קטן יותר מספר העלים היה קטן מ$2^{\log(Num(leaves))} = Num(leaves)$ והיינו מקבלים סתירה. לכן נקבל שמתקיים –

$$height \geq \log\left(\prod_{i=0}^{\frac{n}{k}-1} \binom{n-ki}{k}\right) = \sum_{i=0}^{\frac{n}{k}-1} \log\left(\binom{n-ki}{k}\right)$$

נבחין ש$\log\left(\binom{n}{k}\right) = \log(n!) - \log\big((n-k)!\big) - \log(k!) = \Omega(log(n!))$ (עבור $1 \leq k \leq n$) ולכן מתקיים (החל מn מספיק גדול) –

$$\sum_{i=0}^{\frac{n}{k}-1} \log\left(\binom{n-ki}{k}\right) \overset{c_i \in \mathbb{R}}{\geq} \sum_{i=0}^{\frac{n}{k}-1} c_i(\log((n-ki)\,!)) \geq \sum_{i=0}^{\frac{n}{k}-1} c_{\frac{n}{k}-1}(\log(k\,!)) = c_{\frac{n}{k}-1}\frac{n}{k}(\log(k\,!))$$

$$\overset{\log(n!)=\Omega(n\log n), c\in\mathbb{R}}{\geq} c_{\frac{n}{k}-1}\frac{n}{k}ck\log k = c_{\frac{n}{k}-1}cnlogk \overset{M := c_{\frac{n}{k}-1}c}{=} Mnlogk$$

וזה אומר בדיוק ע"פ הגדרה ש$height = \Omega(nlogk)$ לכן חסם תחתון (אסימפטוטי) הדוק למספר ההשוואות לבעיה הוא $\Omega(\text{nlogk})$.

נשים לב שמקרה פרטי של תוצאה זו כאשר $k = \frac{n}{logn}$ הוא שהחסם התחתון הוא $\Omega(\text{nlog}\frac{n}{logn})$ כלומר $\Omega(\text{nlog}n)$. לחלופין אפשר לעשות רדוקציה למיון – להניח בשלילה שאפשר להפוך מערך ל$zigzag(A, logn)$ ב$o(nlogn)$ עבודה ואז למיין אותו ב$O(nloglogn)$ כפי שראינו בתרגול כיצד למזג k מערכים ממויינים ב$O(nlogk)$ (כאשר n הוא מספר האיברים הכולל) ולקבל סתירה לחסם התחתון לבעיית המיון.

שאלה 2

א. רכיבים:

נתחזק עץ AVL שבנוסף לשדות הרגילים יש לו גם שדות $evensum, oddsum, evensize, oddsize$ ונשתמש גם בשדה גלובלי $counter$. השדות $evensum, oddsum$ הם הסכום של האיברים הזוגיים/אי זוגיים בתת העץ הנוכחי ובדומה $evensize, oddsize$ הם מספר האיברים הזוגיים/אי זוגיים בתת העץ הנוכחי. $counter$ הוא צובר גלובלי ששומר את סכום הערכים של הקריאות ל$shift$. נשים לב שאפשר לעדכן את כל השדות $evensum, oddsum, evensize, oddsize$ בזמן קבוע מאותם השדות של הילדים ולכן שדות אלו ניתנות לתחזוקה באופן יעיל (כלומר מבלי לפגוע בסיבוכיות הרגילה של פעולות העץ AVL).

מתודות:

$Insert(x)$:

נכניס את האיבר עם המפתח $x.key - counter$ (הכנסה רגילה לעץ AVL) ולכן זה $O(logn)$.

$Delete(k)$:

נמחק את המפתח $k - counter$. הסיבה שמחקנו את האיבר הנכון היא ש$k - counter$ בעץ מייצג ערך "אמיתי" k' שהוכנס כ$k' - k'.counter$ כלומר $k' - k'.counter = k - counter$ ולכן מתקיים $k = k' + counter - k'.counter$ ולמעשה מחקנו את k' מהמבנה (כי מבחינת המשתמש הכנסנו את k' ולא את $k' - counter$) והיות ו$k = k' + counter - k'.counter$ זה בדיוק האיבר שהיה צריך להימחק. המחיקה עצמה רגילה ולכן זה $O(logn)$.

$Find(k)$:

מאותו סיבה כמו מתודה קודמת נחפש את המפתח $k - counter$ ונחזיר את הערך שלו. זה חיפוש רגיל ולכן זה $O(logn)$.

$Shift(d)$:

נבצע $counter += d$. זה כמובן $O(1)$.

$EvenSum()$:

אם $counter$ זוגי נחזיר $root.evensum + root.evensize * counter$.

אם $counter$ אי זוגי נחזיר $root.oddsum + root.oddsize * counter$.

הסיבה שזה נכון היא כמובן שאם $counter$ אי זוגי אז למעשה צריך להחזיר את הסכום של האי זוגיים (אבל מתוקן, כלומר להוסיף $root.oddsize * counter$ שכן הכנסנו את הערכים במקור עם הזזה מטה ולכן על כל ערך אי זוגי צריך להוסיף את $counter$ הנוכחי שיפצה על ההזזה ואולי אפילו יוסיף יותר אם מאז ההוספה של האיבר היו עוד *shift*ים). זה כמובן $O(1)$.

ב. רכיבים:

נתחזק טבלת האש בגודל n שעובדת בשיטת הchaining. בנוסף נתחזק שדות גלובליים $counter, oddsize, evensize, oddsum, evensum$ שמייצגים את סכום המפתחות הזוגיים/אי זוגיים ומספר המפתחות הזוגיים/אי זוגיים בטבלה בכל רגע נתון. השדה $counter$ יהיה צובר גלובלי לערכים של הקריאות ל*shift*.

המתודות עצמן יעבדו באופן זהה לקודם רק שעכשיו ההכנסה היא לטבלת האש ולא לעץ *avl* . נשים לב שבגלל שאנחנו עובדים עם טבלה בגודל n ויש לכל היותר n איברים בטבלה נקבל זמן ריצה קבוע בתוחלת לכל הפעולות (בנוסף כמובן שאפשר לתחזק את כל השדות בזמן קבוע ע"י בדיקות פשוטות של הזוגיות של המפתחות שמכניסים ועדכון השדות המתאימים).

שאלה 3

א. הרעיון יהיה לחקות את האלגוריתם $quickselect$, נחלק את המפתחות ל$\sqrt{n}$ קבוצות שבכל אחת יש $\sqrt{n}$ מפתחות, ועל כל קבוצה נפעיל את השגרה A . לבסוף נפעיל פעם אחת נוספת את השגרה A על $\sqrt{n}$ המפתחות שיתקבלו ונחזיר את החציון שהתקבל מהפעלה זו. ראשית נשים לב שאלגוריתם זה עובד בזמן $O\left(\sqrt{n}\sqrt{n}+\sqrt{n}\right)=O\left(n+\sqrt{n}\right)=O(n)$ ולכן סיבוכיות הזמן שמתקבלת מתאימה לדרישות השאלה. נשאר רק להסביר למה הדרגה של המפתח שהוחזר היא באמת בין $n/4$ ל$3n/4$. נשים לב שחציון החציונים גדול מלפחות $\frac{\sqrt{n}}{2}\cdot\frac{\sqrt{n}}{2}=\frac{n}{4}$ איברים (על כל איבר שקטן ממנו יש $\frac{\sqrt{n}}{2}$ שקטנים מהאיבר עצמו שקטן ממנו). באופן סימטרי הוא קטן מלפחות $\frac{\sqrt{n}}{2}\cdot\frac{\sqrt{n}}{2}=\frac{n}{4}$ איברים. לכן סה"כ הדרגה של מפתח זה היא באמת בין $n/4$ ל$3n/4$.

ב. באופן דומה לסעיף א', קודם נחלק ל$n^{2/3}$ קבוצות בגודל $n^{1/3}$ ונמצא את החציון של כל אחת מהם ע"י הפעלה של השגרה A. לאחר מכן נחלק שוב את קבוצת החציונים שהתקבלה ל$n^{1/3}$ קבוצות בגודל $n^{1/3}$ ונמצא את החציון של כל אחת מהן (שוב ע"י הפעלה של השגרה A). לבסוף נחזיר את החציון של הקבוצה שהתקבלה (ע"י הפעלה בודדת נוספת של השגרה A). סה"כ סיבוכיות הזמן תהיה $O\left(n^{\frac{2}{3}}n^{\frac{1}{3}}+n^{\frac{1}{3}}n^{\frac{1}{3}}+n^{\frac{1}{3}}\right)=O\left(n+n^{\frac{2}{3}}+n^{\frac{1}{3}}\right)=O(n)$ כלומר הסיבוכיות מתאימה לדרישות השאלה. בנוסף נשים לב שהמפתח הנתון באמת מקיים את הדרישות הרצויות שכן הוא גדול מלפחות $\frac{n^{1/3}}{2}\frac{n^{1/3}}{2}\frac{n^{1/3}}{2}=\frac{n}{8}$ איברים (על כל חציון של הקבוצות יש $\frac{n^{1/3}}{2}$ שהוא גדול מהם ויש $\frac{n^{1/3}}{2}$ חציונים שעל כל אחד מהם יש עוד $\frac{n^{1/3}}{2}$ חציונים שהוא גדול מהם). באופן סימטרי הוא קטן מלפחות $\frac{n}{8}$ איברים ולכן הדרגה שלו היא בין $n/8$ ל$7n/8$.

<u>שאלה 4</u>

א. נתאר אלגוריתם שעובד ב$O(n)$ בתוחלת. נאתחל טבלת האש בגודל n שעובדת בשיטת הchaining. נכניס כל מפתח לטבלה ב$O(1)$ בתוחלת. עד כה הסיבוכיות היא $O(n)$ בתוחלת. כעת נעבור על כל מפתח בטבלה ונחפש את העוקב שלו ב$O(1)$ בתוחלת ואם קיים אז נדפיס/נוסיף לתוצאה שנחזיר את הזוג $(k, k+1)$ ואחרת נעבור למפתח הבא בתור. זה עוד $O(n)$ בתוחלת ולכן סה"כ כל האלגוריתם עובד ב$O(n)$ בתוחלת.

ב. נתאר אלגוריתם שעובד ב$O(n)$ בתוחלת. נאתחל טבלת האש בגודל n שעובדת בשיטת הchaining. נאתחל גם משתנה לשמירת הגודל של הקבוצה המקסימלית הנוכחית (מאותחל בהתחלה ל1) ועוד משתנה זמני. נכניס את כל המפתחות לטבלה ב$O(n)$ בתוחלת. כעת נסרוק את הטבלה, ועל כל מפתח k שבה נחפש את העוקב ואם הוא קיים נעלה את המשתנה זמני ב1 (הוא ישמור את הגודל של הקבוצה הנוכחית שנבדוק) ונמשיך כך עד שהעוקב לא קיים יותר בטבלה. נעשה כך גם לקודם (ובכל פעם שקיים נעלה את המשתנה הזמני ב1). לאחר מכן נבדוק אם המשתנה הזמני גדול מגודל הקבוצה המקסימלית ואם כן הוא הגודל המקסימלי החדש, ונמחק את כל המפתחות שעברנו עליהם (אין צורך בהם יותר כי לולא היו חלק מקבוצה אחרת שני הקבוצות היו מתמזגות לאחת). לבסוף נחזיר את הגודל המקסימלי. סה"כ האלגוריתם עובד ב$O(n)$ בתוחלת כי על כל איבר בטבלה עשינו $O(1)$ עבודה בתוחלת ויש לכל היותר $2n$ איברים שלא בטבלה שבדקנו האם הם בטבלה. כלומר נקבל $O(n + n \cdot 1 + 2n) = O(n)$ עבודה בתוחלת.

שאלה 5

רכיבים:

מערך בוליאני (של 0/1) באורך n ששומר באינדקס i האם הרקדן הi מזווג או לא – 0 אם הוא לא ו1 אם הוא כן. בנוסף נשמור ערימת מקסימום ששומרת בתור מפתח את המספר האי שלילי שמייצג את האיכות של זוג רקדנים ובתור ערך את הזוג עצמו כזוג מזהים (i,j). כאשר המבנה מאותחל, מאתחלים את המערך להיות מערך של 0ים ב$O(1)$ (כפי שנלמד בכיתה) ואת הערימה להיות ערימה ריקה בעוד $O(1)$ זמן. סה"כ האתחול עובד ב$O(1)$ זמן.

מתודות (הגענו אפילו לסיבוכיות worst case):

$SetMatchValue(i,j,v)$:

נבדוק האם אחד מהרקדנים כבר מזווג (ע"י גישה למערך הבוליאני שלנו באינדקסים שערכם הם המזהים של הרקדנים) ונכניס לערמת המקסימום את המפתח v עם הערך (i,j) אם ורק אם שני הרקדנים לא מזווגים. אחרת לא נכניס אותם. סה"כ זה $O(logn)$ במקרה הגרוע.

$Match()$:

נחזיר את מפתח של השורש והערך שלו כשלשה (i,j,v). אם לא קיים שורש (הערימה ריקה) נחזיר null. בנוסף נסמן את i,j כמזווגים. סה"כ זה $O(1)$ במקרה הגרוע.