**תרגיל בית מעשי 2 – תיעוד פונקציות**

**עומר פלביץ 206840126  
אלעד שבא 207909409**

מחלקת FibonacciHeap

getFirst – מחזיר את האיבר הראשון בערימה O(1)

getLast – מחזיר את האיבר האחרון בערימה O(1)

isEmpty – בודק האם הערימה ריקה לפי איבר המינימום שחייב להיות בכל ערימה בעלת איבר אחד. O(1)

Insert- קודם כל נשים לב כי יש שני מימושים לפונקציה זו (העמסה), שניהם עובדות באופן זהה, אחת מקבלת ככקלט ערך מספרי והשנייה מקבלת ככקלט heapNode (לשימוש אחרי החיתוך).  
אם קיבלנו ערך – קודם ניצור את הצומת. כעת אם הערימה ריקה, נאתחל את הצומת להיות השורש, האיבר הראשון ואיבר המינימום. אם לא , נכניס את הצומת החדשה משמאל לצומת האחרונה שהוכנסה (מחזיק first - האיבר הראשון בערימה), נעדכן את השדות של האיבר הראשון, החדש (שיהפוך לראשון) והאחרון בערימה. לאחר מכן נבדוק את הערך החדש אם הינו קטן מהמינימום, אם כן נחליף את המצביעים. לאחר מכן נחזיר את הצומת החדשה. כלל הפעולות שביצענו הינם פשוטות וקשורות לעדכון שדות ויצירת משנה פשוט, כמו כן השתמשנו בפונקצית עזר isEmpty בסיבוכיות O(1) לכן סיבוכיות כוללת הינה O(1).

FindMin – נחזיר את המצביע לאיבר המינימום (המצביע מתוחזק בכל ההכנסות והמחיקות), O(1). אם הערימה ריקה נחזיר null.

Size- נחזיר את השדה של גודל הערימה, O(1).

Delete – נקבל צומת שברצוננו למחוק, נקח את הערך של האיבר המינימלי ברשימה, ואת ערך הצומת הנוכחית ונסמן ונקרא לפעולה decrease key עם ערך הדלתא הנוכחי, נשים לב כי בהכרח נקבל שערך הצומת הנוכחית הינה מינימלית ! כעת שהצומת הינה המינימלית ועלתה לראש הערימה, נבצע מחיקת איבר המינימום.  
הסיבוכיות של פונקציה זו נובעת מקריאתה לפעולות העזר , הפחתת ערך מפתח שהיא ב WC O(n) אבל באמורטייז O(1) ומחיקת איבר מינימום שהיא במקרה הגרוע O(n) ואמורטייז O(log n) לכן סה"כ סיבוכיות הפעולה הינה O(n) במקרה הגרוע, ו O(log n) באמורטייז.

deleteMin – מוחקת את איבר המינימום. תחילה נבדוק האם איבר המינימום הוא האיבר היחיד – אם כן נהפוך את הערימה לריקה. אחרת, נבדוק אם יש לו בן. אם יש לו, נחבר את הבן כשורש חדש ונחפש מינימום חדש. אם אין לו בן, נמחק אותו ונחבר בין האח הימני שלו לאח השמאלי שלו. בסוף נעשה consolidate . סה"כ זמן הריצה במקרה הגרוע הוא O(n), ואמורטייז O(log n ), כפי שראינו בכיתה.

Meld – מחברת בין שתי ערימות, כך שהערימה השנייה מתחברת לערימה הראשונה. הפונקציה בודקת אם הערימה ה2 ריקה, אם כן משאירה את הערימה הראשונה איך שהיא. אם היא לא ריקה, נחבר את האיבר הראשון בערימה הראשונה לאיבר האחרון בשנייה, נתאים את הגדלים וזהו. סך הכל ביצענו פעולה של חיבור צמתים, זמן הריצה הוא O(1).

Max\_rank – פונקציית עזר, מחזירה את הrank הגבוה ביותר בערימה. עוברת על השורשים ובודקת rank. O(n).

countersRep – מחזירה מערך בו כל כניסה במערך הוא כמות העצים בדרגה המתאימה לכניסה. הפונקציה בודקת האם הערימה ריקה ואם כן מחזירה מערך ריק. אם לא, עוברת על השורשים ומוסיפה 1 לכל מקום במערך בו יש שורש עם הדרגה המתאימה. סה"כ עוברים במקרה הגרוע O(n) שורשים (גם בפוקנציית עזר max\_rank) לכן זהו הזמן ריצה.

decreaseKey – מורידה מצומת מסוים מספר שלם מהמפתח. לאחר מכן מארגנת את העץ בהתאם. הפונקציה בודקת האם הצומת הוא שורש. אם הוא שורש אז הוא יישאר במקומו, נבדוק אם הוא נהיה המינימום ואם כן נחליף את המינימום בהתאם. אם הוא לא שורש, נבדוק האם הוא עדיין גדול מההורה שלו. אם כן – לא נעשה כלום. אם הוא יותר קטן, נבצע חיתוך בעזרת פונקציות העזר. במקרה הגרוע החיתוכים יכולים לעלות ולהגיע עד O(n), אבל בזמן אמורטייז זה O(1).

Non marked: נשתמש בשדות המתוחזקים ונחזיר את ההפרש בין this.size לבין this.sum\_marked, ההפרש בין גודל הערימה לבין מספר הצמתים המסומנים. מכיוון שזוהי רק קריאה לשדות הסיבוכיות הינה O(1).

Potential – פונקציה המחזירה פוטנציאל. מאחר ואנחנו מחזיקים ומתחזקים שדות של מספר העצים ומספר הצמתים המסומנים זה יהיה רק קריאה של שדות ולכן O(1).

totalLinks – נחזיר את השדה הסטטי המתוחזק של מספר ה link שבוצעו בערימה, סיבוכיות O(1).

totalCuts – מחזיר את מספר החיתוכים. זהו שדה שמתחזקים לכן O(1).

Cuts+cascadingCut – פונקציות שחותכות צמתים כפי שנראה בשיעור. כפי שראינו זמן הריצה הוא במקרה הגרוע O(n) וO(1) אמורטייז.

Consolidate + connect\_same פונקציות עזר לdelete min שמטרתן לאחד את העצים הזהים לעצים גדולים יותר. ניצור מערך של rank של עצים שבכל מקום במערך נאחסן עץ אם הוא בrank של המקום במערך. אם יהיו שני עצים באותו מקום נשתמש בconnect ונאחד בינהם.

Kmin- הפונקציה מקבלת ערימה ומספר k. נטפל קודם במקרי הקיצון , אם הערימה בגודל 0 או שk=0, נחזיר רשימה ריקה, אם גודל הערימה או k שווים ל1 נחזיר רשימה בגודל 1 עם מפתח איבר המינימום בערימה.  
נממש את הפונקציה הזו, כהרחבה של מה שראינו במטלת הבית. מימשנו ערימת עזר KminHeap, נכניס אליה את הערך של איבר המינימום בערימה המקורית, ונשמור מצביע בkminHeap לאיבר הזה (המצביע הינו השדה location), כמו כן נתחזק צומת בשם curr ששווה לאיבר המינימום בערימה המקורית בהתחלה.  
כעת נעבור בלולאה מ0 עד k, כל פעם נכניס במקום הi לרשימה את curr (ז"א שבהתחלה נכניס את איבר המינימום מהרשימה המקורית), כעת נמחק את איבר המינימום מ kminHeap ונעבור בלולאה על הילדים של curr, נכניס כל פעם ילד ל kminHeap ונתחזק את שדה המיקום שיצביע לילד הנוכחי שהכנסנו , נשתמש בדגל כדי לשים לב מתי חזרנו שוב לילד הראשון שהכנסנו בסיבוב הנוכחי.  
בצורה כזו אנו מכניסים ל KminHeap את כל הילדים של האיבר המינימום שמחקנו, כעת לאחר שהכנסנו את כל ילדיו (ה"מועמדים"), נעדכן את curr להיות האיבר בערימה המקורית, שהוא איבר המינימום בkminHeap (בעזרת שדה המיקום), כעת נתחיל את הלולאה שוב כאשר i התקדם ב1 ,נכניס את איבר המינמום החדש למערך, ונמחק את איבר המינימום החדש בערימה בדיוק כפי שביצענו את כל המהלך הנוכחי.

הוכחנו במטלה 4 כי בשיטה כזו זמן הריצה יהיה O(klogk) , מאחר ו k קטן ממש מ size(H), ומאחר וH הוא עץ בודד בערימה נקבל:

ולכן סיבוכיות זמן הריצה הינה O(k\*deg(H))

מחלקת HeapNode

נחזיק שדות של מפתח, בן, הורה, אח ימני ושמאלי, האם הצומת מסומן ומה הrank שלו. בנוסף שדה של מיקום ( עליו יש פירוט ב kmin).

לכן כל הget של כל השדות האלה הוא O(1).

חלק תיאורטי:

שאלה 1:

אנו מבצעים סדרת הכנסות (של m איברים), פעולת הכנסה היא בעלת סיבוכיות O(1), לכן סיבוכיות הכנסת הסדרה בכללותה היא O(m), לאחר מכן מבצעים delete min -> פעולה המבצעת m-1 פעולות link, סה"כ סיבוכיות של כ O(m)  
לאחר שאיחדנו הכול לעץ אחד, אנו מבצעים logm פעולות של decrease key, פעולות אלה מתרחשות על הצמתים שמחוברים לצמתים על הענף השמאלי ביותר, במקרה הרע מבצעים cut יחיד בשלב זה, סיבוכיות כוללת הינה O(logm)  
לכן סה"כ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| potential | Total Cuts | Total Links | Run time(ms) | M |
| 14 | 5 | 31 | 0.0871 |  |
| 29 | 10 | 1023 | 0.511 |  |
| 44 | 15 | 32767 | 8.51 |  |
| 59 | 20 | 1048575 | 116 |  |

1. במצב הסטנדרטי – כשm עלה בחזקת 2, מספר הcuts עלה פי 2, לכן הסיבוכיות הינה   
   כמות הlink שביצענו היא תמיד 1 פחות מm, ולכן סיבוכיות כתלות בm הינה .

סעיפים ג'-ו':

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Decrease key cost max | potential | Total Cuts | Total Links | case |
|  |  |  | m-1 | original |
|  | 1 | 0 | m-1 |  |
|  | m+1 | 0 | 0 |  |
|  |  |  | m-1 | Added line #4 |

נימוקי סיבוכיות:

בסיטואציה המקורית:  
• link –לאחר מחיקת המינימום, נותרו צמתים, ולכן יתבצעו מתבצעות m-1 פעולות link.

נוכיח באינדוקציה כי בconsolidate נקבל עץ מלא ונבצע בדיוק m-1 פעולות link, נשים לב כי m הוא חזקה של 2

בסיס :כשיש ערימה בגודל 2 ז"א שיש שני עצים בגודל 0 ז"א שלאחר ביצוע link יחיד   
= 2-1 נקבל עץ יחיד.

צעד: נניח כי יש עץ שנוצר לאחר link ל צמתים נתבונן על מהנחת האינדוקציה יווצרו שני עצים בגודל m, ואז נבצע link בין שניהם, ונקבל עץ יחיד, נשים לב כי בוצעו לינקים ועוד לינק יחיד.  
סה"כ:

• פעולות cut – מבצעים logm פעולות על הצמתים הנמצאים על הענף השמאלי ביותר של העץ, לאחר ביצוע הפעולה מבצעים חיתוך יחיד (כי הפעולה מתבצעת על בן יחיד של כל צומת בענף השמאלי), לכן מבצעים log m חיתוכים, לאחר הdecreasekey כל צומת הופכת לאיבר המינימום ויש צורך לבצע חיתוך. אנו עושים decrease key לתתי העצים מדרגה 0 של העץ על הענף השמאלי ביותר, מהגדרה לכל צומת יהיה לכל היותר רק בן יחיד מדרגה 0, לכן חותכים בדיוק בן אחד ולא שניים. לכן עבור כל decreasekey מבצעים חיתוך.

• פונקציית הפוטנציאל:

צמתים מסומנים – יהיה מספר החיתוכים שעשינו פחות 1 : לכן סה"כ log m-1

מספר העצים – אחרי מחיקת איבר המינימום, יהיה לנו עץ יחיד. לאחר כל חיתוך צומת, נקבל עץ נוסף, לכן נקבל log m +1  
נזכור כי בנוסחת הפוטנציאל מכפילים את מספר הצמתים המסומנים ב2

לכן לפי נוסחת הפוטנציאל ערכה הוא -

decKey(m-2^i):

במקום לבצע את הפחתת המפתח לעלים שעל הענף השמאלי, אנו מבצעים אותם על הצמתים עצמם שבענף השמאלי ( ה"אבות" של מי שעשינו במקור").  
מתחילים בלהפחית את ערך איבר המינימום (מה שמשאיר אותו מינימלי), הולכים שמאלה ומקטינים את האיבר הבא. לאחר ההקטנה הולכים שוב שמאלה לענף השמאלי וממשיך בהקטנה בצורה רקורסיבית, למעשה אין כאן מעבר על כללי הערימה:

לינקים – יהיה כמו במצב המקורי m-1  
חיתוכים – אין הפרה לכללי הערימה, לכן גם לא נחתוך : 0

פוטנציאל- אין חיתוכים, אזי אין צמתים מסומנים, כמו כן יהיה לנו רק עץ אחד, ולכן הפוטנציאל הינו 1.

Remove line #2:

לא ביצענו מחיקה לאיבר, לכן הערימה תהיה למעשה m+1 עצים מדרגה 0, לכן:

מספר הלינקים – יהיה 0 (לא מחקנו איבר לכן בערימת פיבונאצי גם אין לינק)

מספר החיתוכים – יהיה 0 (כלל העצים הם למעשה שורשים בודדים, אין הפרה לכלל הערימה בהפחתת ערך המפתחות שלהם ולכן אין חיתוכים)  
פוטנציאל – אין חיתוכים אזי אין צמתים מסומנים, כמות העצים הינה m+1 ולכן פוטנציאל הינו m+1

Added Line #4

נשים לב כי התהליך שמתרחש הינו שחתכנו את כל העצים מדרגה 0, ואז מבצעים הפחתת ערך מפתח על הצומת האחרונה ושמאלית בענף השמאלי של העץ. פעולה זו תגרום להפרה ולכן:

מספר הלינקים : לא ישתנה, יהיה כמו בסעיף המקורי m-1

מספר החיתוכים :חתכנו בן יחיד מכל צומת בענף השמאלי, כל הצמתים האלו יסומנו (כי איבדו בן יחיד) ובשלב הבא למעשה יתרחש cascading cut.  
העץ היה בהתחלה logm גובה, הסרנו את העלה העמוק ביותר, מה ששינה את גובהו ל logm-1. וזה יהיה מספר הפעולות הנוספות שיתרחשו בגלל הפחתת ערך המפתח. סה"כ 2log m -1 חיתוכים.

פוטנציאל: בcascading cut אנו למעשה מורידים את מספר הצמתים המסומנים ומעלים את מספר העצים. מספר הצמתים המסומנים יהפוך ל0 ואזי פונקצית הפוטנציאל תהיה סכום העצים מהמודל המקורי ועוד העצים החדשים שנוצרו בחיתוך