**מסמך תיעוד וניתוח סיבוכיות**

**מגישות:** ליאור פרניק ואלין לושבסקי

**המחלקה BinomicalHeap –**

המחלקה מתארת ערימה בינומית, המחלקה מממשת פעולות הכנסה, מחיקה, מחיקת מינימום, הפחתת מפתח של צומת נתון, מיזוג ערימות, חישוב גודל הערימה, בדיקה האם הערימה ריקה וחישוב מספר השורשים בערימה.

שדות המחלקה הם גודל הערימה, מצביע לצומת עם הדרגה הגבוהה ביותר ומצביע לצומת עם המפתח המינימלי.

**הפונקציות המרכזיות הנמצאות במחלקה הן:**

* constructors
* meld
* insert
* deleteMin
* decreaseKey
* Delete
* findMin
* Size
* numTrees
* empty

**פונקציות העזר לפונקציות המרכזיות –**

* Size\_of\_ranks
* Update\_fields
* meld2Roots
* connectRoots
* switchItems

**תיאור הפונקציות וניתוח סיבוכיות זמן הריצה שלהם –**

1. **הבנאים של המחלקה –**

ישנם 3 בנאים במחלקה:

* 1. **בנאי ריק –**

הבנאי בונה ערימה ריקה ומאתחל את הגודל שלה ל-0

* 1. **בנאי המקבל צומת –**

הבנאי בונה עץ עם שורש יחיד שהוא הצומת שהתקבלה ומעדכן את שדה השורש בהתאם לדרגת הצומת. בנאי זה משומש בפעולות Insert ו-meld.

* 1. **בנאי המקבל צומת ודרגה –**

הבנאי משומש בפעולה deleteMin לבניית ערימת ילדי המינימום.

הדרגה שמתקבלת היא הדרגה של this.min.child, במידה והדרגה גבוהה מ-0 (יש יותר מילד אחד) נקרא לפעולה size\_of\_ranks שתחשב את גודל הערימה ול-update\_fields, לעדכון שדות הערימה.

* + 1. **הפעולה size\_of\_ranks –**

הפעולה מקבלת כקלט צומת ועוברת על כל האחים שלה תוך כדי סכימת כמות הצמתים שהם כוללים. הפעולה מחזירה את גודל הערימה שהם מייצגים. ← O(Log(n))

1. **הפעולה meld –**

הפעולה מקבלת כקלט ערימה heap2 למיזוג, ומאחדת את הערימה הבינומית הנוכחית עם heap2. הפעולה ממזגת ביעילות את שתי הערימות על ידי ביצוע השלבים הבאים:

* טיפול בערימות ריקות: אם אחת הערימות ריקה, המתודה מתאימה את הערימה הנוכחית בהתאם. ← O(1).
* הכנה ואחסון עצים: נוצר מערך לאחסון עצים בינומיים ברמות שונות. מוסיפים למערך במקומות המתאימים את העצים הקיימים בערימה הנוכחית.
* עיבוד עצי heap2: כל עץ מ-heap2 מתמזג עם העצים המתאימים במערך על פי הרמות שלהם בעזרת הפעולה meld2Roots.

בסוף האיחודים, נקראת הפעולה connectRoots עם המערך הסופי של העצים.

סה"כ 🡨 O(log(n))

* 1. **הפעולה meld2Roots –**

הפעולה מקבלת כקלט 2 צמתים f,s מעדכנת את f להיות ההורה של s ומעדכנת את הדרגות של כל הילדים של f. ← O(deg(f))=O(log(n)).

* 1. **הפעולה connectRoots –**

הפעולה מקבלת כקלט מערך של עצים ומחברת אותם יחד כדי ליצור את הערימה המאוחדת הסופית. בנוסף במהלך המעבר על השורשים, מעדכנת את המינימום של הערימה אם יש צורך.  מספר העצים לוגריתמי ביחס למספר הכולל של הצמתים בערימה.← O(Log(n))

1. **הפעולה insert –**

הפעולה מקבלת כקלט מפתח ומידע, יוצרת צומת חדש עם הערכים, ומוסיפה אותו לערימה, ישנם מספר מקרים נפרדים:

1. קיים בערימה עץ B0 ויש צורך במיזוג הצומת החדש לערימה.
2. לא קיים בערימה עץ B0 או שהערימה ריקה, נשרשר את הצומת כשורש חדש בערימה.

במקרה ב, נשתמש בפונקציה update\_fields לעדכון שדות הערימה.

* 1. **הפעולה update\_fields –**

הפונקציה מקבלת כקלט צומת ו-size, הפונקציה מוסיפה לשדה גודל הערימה את size ומעדכנת את שדה המינימום וה-last בהתאם לצורך. ← O(Log(n))

1. **הפעולה deleteMin –**

הפעולה לא מקבלת קלט, מוחקת את המינימום הנוכחי ומתקנת את מבנה העץ בהתאם.

ישנם 3 מקרים אפשריים:

1. העץ הוא צומת אחת - והמחיקה תוביל לעדכון השדות של העץ כדי שיהיה ריק.
2. קיים שורש יחיד בעץ, במקרה זה הערימה תוחלף בערימת ילדי המינימום.
3. אחרת, נבצע meld בין השורשים בעץ לילדי המינימום.

סה"כ ← O(Log(n)).

1. **הפעולה decreaseKey –**

הפעולה מקבלת כקלט צומת ומספר diff, הפעולה מחסרת מערך מפתח הצומת את diff ועושה heapify\_Up בעזרת לולאה שבודקת כל עוד הצומת ההורה גדול יותר לאחר שינוי המפתח, נבצע את פעולת SwitchItems שנמצאת במחלקת HeapItem, ולבסוף, במידה והצומת הוא המינימום החדש, נעדכן את השדה min. העומק המקסימלי של עץ בערימה הוא Log(n) ← O(Log(n))

* 1. **הפעולה SwitchItems –**

הפעולה מקבלת item ומחליפה בין הערכים של key,info של this, item. 🡨 O(1)

1. **הפעולה delete –**

הפעולה מקבלת כקלט צומת למחיקה v, מבצע עליה פעולת decreaseKey(v,inf)

ולאחר מכן, מבצעת את פעולת deleteMin. סה"כ ← O(Log(n))

1. **הפעולה findMin –**

הפעולה לא מקבלת קלט, מחזירה מצביע למינימום של הערימה.←(1)O

1. **הפעולה size –**

הפונקציה לא מקבלת קלט, מחזירה את גודל העץ בעזרת גישה לשדה גודל של שורש העץ. ←(1)O.

1. **הפעולה numTrees –**

הפעולה לא מקבלת קלט, עוברת וסופרת את כל השורשים של העץ לפי סדר הדרגות, קיימים לכל היותר (log(n שורשים. ← O(Log(n))

1. **הפעולה empty –**

הפונקציה לא מקבלת קלט, מחזירה האם שדה גודל העץ הוא 0 או לא. ← (O(1

**חלק תיאורטי**

**טבלאות תוצאות ניסויים 1-3:**

**ניסוי 1:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **מספר סידורי** | **זמן ריצה (מילישניות)** | **מספר החיבורים הכולל** | **מספר העצים בסיום** | **סכום דרגות הצמתים שמחקנו** |
| 1 | 6 | 9836 | 6 | - |
| 2 | 14 | 39360 | 6 | - |
| 3 | 58 | 127931 | 9 | - |
| 4 | 134 | 393656 | 4 | - |
| 5 | 254 | 1190813 | 9 | - |

**ניסוי 2:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **מספר סידורי** | **זמן ריצה (מילישניות)** | **מספר החיבורים הכולל** | **מספר העצים בסיום** | **סכום דרגות הצמתים שמחקנו** |
| 1 | 33 | 43619 | 5 | 37063 |
| 2 | 76 | 193350 | 6 | 167112 |
| 3 | 160 | 689154 | 8 | 603870 |
| 4 | 413 | 2337830 | 4 | 2075396 |
| 5 | 1079 | 7677960 | 8 | 6884089 |

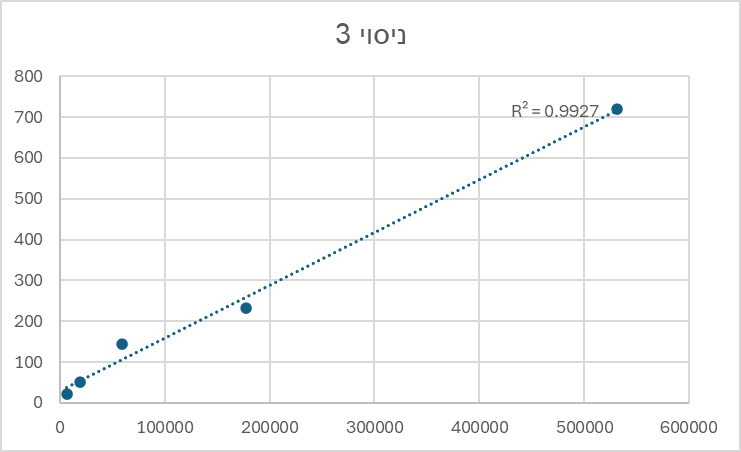
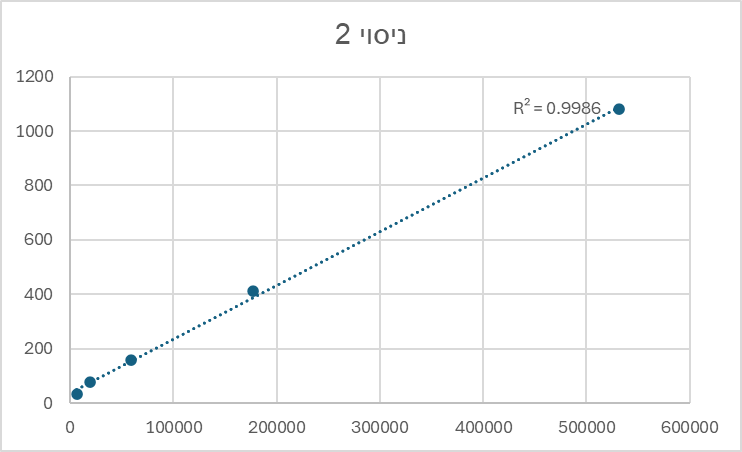
**ניסוי 3:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **מספר סידורי** | **זמן ריצה (מילישניות)** | **מספר החיבורים הכולל** | **מספר העצים בסיום** | **סכום דרגות הצמתים שמחקנו** |
| 1 | 22 | 9835 | 5 | 6529 |
| 2 | 50 | 57424 | 5 | 44277 |
| 3 | 143 | 203725 | 5 | 161054 |
| 4 | 232 | 653503 | 5 | 522259 |
| 5 | 720 | 2038326 | 5 | 1641362 |

**ניתוח זמן הריצה כתלות במספר האיברים בגרף עבור ניסויים 1-3:**

כמתואר בגרפים, האלגוריתם רץ בזמן לינארי עבור סדרת n הכנסות וגם עבור סדרת n הכנסות שלאחריה סדרת מחיקות, בדיוק כפי שראינו בהרצאה.

תמונה שמכילה טקסט, קו, צילום מסך, עלילה

התיאור נוצר באופן אוטומטילהלן גרפים המתארים את זמני הריצה כתלות בגודל הקלט ב-3 הניסויים:

**תלות סכום הדרגות שנמחקו כתלות בכמות הלינקים וגודל העץ הסופי:**

נשים לב שעבור כל מחיקת צומת מדרגה  נוריד ב1 את כמות הצמתים בעץ והילדים שלו (יש לו k ילדים) הופכים לשורשים של עצים

כאשר מוחקים צומת מדרגה k, יש לה k ילדים שאותם נצטרך למזג לערימה הקיימת, גודל הערימה יקטן ב-1 ויתבצעו לכל היותר 2k מיזוגים סה"כ.

