**מבני נתונים – פרויקט מעשי 2 - קובץ תיעוד**

**פרטים אודות מגישות-**

הגר פייתן

* שם משתמש - hagarleap
* תעודת זהות - 206825176

גל קריאל

* שם משתמש - galkariel
* תעודת זהות - 318459666

**חלק ניסויי/תיאורטי**

**שאלה 3 – השוואה בין quadratic probing ל-alternating quadratic probing**

**שאלה 3 - סעיף 1**

גדלי הקבוצות:

( נבחין כי הגודל של זה בדיוק וכי הגודל של זה בדיוק q )

**שאלה 3 - סעיף 2**

מדדנו עבור quadratic probing את כמות השגיאות הנובעות משגיאה כי הטבלה מלאה ומשגיאה כי מפתח כבר קיים במערך.

ראינו כי אין שגיאות שנובעות ממפתח שקיים כבר במערך, אבל קיבלנו בין 0 ל-3 שגיאות בכל ניסוי שנובעות מכך שהטבלה מלאה.

לעומת זאת עבור alternating quadratic probing לא נזרקו שגיאות כלל וכל האיברים הוכנסו לטבלה.

הסבר לשוני בין התוצאות-

ראינו מהסעיף הקודם כי כאשר אנחנו משתמשים במודל של quadratic probingאז כתוצאה סדרת הבדיקות שלנו עוברת רק על קבוצת מספרים בגודל חצי מהמספר הראשוני q () . כלומר פונקציית ה-hash במודל של quadratic probing לא נותנת לנו את כל האינדקסים בטבלה ולכן ייתכנו סדרות של בדיקות שלא עוברות על כל האינדקסים בטבלה כלומר נקבל שגיאות כי הטבלה מלאה למרות שבפועל יש בה עוד מוקמות להכניס איברים.

לעומת זאת כאשר אנחנו משתמשים במודל של alternating quadratic probing ראינו בסעיף הקודם שסדרת הבדיקות שלנו עוברת על קבוצת מספרים בגודל q () כלומר אין חזרות של מספקים ואנחנו עוברים **על כל** האינדקסים בטבלה– כלומר בהכרח נמצא מקום פנוי אם קיים לטובת הכנסת מפתח חדש לכן לא נקבל שגיאות ונצליח להכניס את כל המפתחות.

**שאלה 3 - סעיף 3**



**שאלה 4 – נשווה בין המימושים השונים ל-open addressing**

**שאלה 4 - סעיף 1**

|  |  |
| --- | --- |
| Class | Running Time |
| LPHashTable | 0.675 |
| QPHashTable | 0.560 |
| AQPHashTable | 0.688 |
| DoubleHashTable | 1.181 |

הסבר מילולי להבדלים בזמני הריצה בין סוגי הטבלאות-

נבחין כי זמני הריצה של DoubleHashTable יותר ארוכים באופן יחסי לעומת שאר המימושים. ושאר המימושים רצים בזמנים מאוד דומים.

ראינו בהרצאה כי הסיכוי להתנגשויות כאשר משתמשים בפונקציות hash מהמשפחות האוניברסליות הנו נמוך ( כאשר m זה גודל טבלת ה-hash). בנוסף אנחנו ממלאים רק את חצי הטבלה, ולכן נעריך כי אפילו אם יש התנגשויות לא יהיו "רצפים" גדולים מידי של איברים בטבלה (שעלולים היו לפגוע בזמני הריצה).

לכן אנו מעריכות כי בניסוי זה השוני נובע מהחישובים שקוראים בעת הכנסת איבר –

ב- DoubleHashTable יש יותר חישובים כבדים כי אנחנו מפעילים שתי פונציות hash עבור כל הכנסת איבר (וראינו במבוא מורחב כי mod הנה פעולה יחסית יקרה) לכן זמני הריצה שלה באופן יחסי ארוכים יותר.

**שאלה 4 - סעיף 2**

|  |  |
| --- | --- |
| Class | Running Time |
| LPHashTable | 1.230 |
| AQPHashTable | 1.274 |
| DoubleHashTable | 1.721 |

נבחין כי בדומה לסעיף הקודם זמני הריצה של DoubleHashTable יותר ארוכים באופן יחסי לעומת שאר המימושים. ושאר המימושים רצים בזמנים מאוד דומים.

צריכות להשלים הסבר גם בנפרד כל אחד

בנוסף ניתן לראות כי זמני הריצה של LPHashTable ו- AQPHashTableגדלו פי 2, אנו מעריכות כי הדבר נובע מכך שכמות האיברים שאנחנו מכניסים לטבלה הוא בערך פי 2 בהתאמה.

לעומת זאת ב- DoubleHashTable...........................

ההסבר לכך שאין לבצע סעיף זה עבור הנה - כפי שראינו בשאלה הקודמת, פונקציית ה-hash במודל של quadratic probing לא נותנת לנו את כל האינדקסים בטבלה, ובפרט נותנת לנו רק חצי מהם.

לכן הכנסה של כמות איברים שגדולה מחצי מהטבלה עלולה לגרום לשגיאות "table is full".

בניסוי זה אנו מכניסים יותר מחצי איברים של הטבלה ולכן ייתכן ויהיו שגיאות (בשאלה זו לא אמורים להיזרק חריגים) ולכן לא נבצע בסעיף את המימוש עבור .

**שאלה 5 – השפעת מחיקת איברים ב-open addressing**

|  |  |
| --- | --- |
| Iterations | Running Time |
| First 3 iterations | 4.513 |
| Last 3 iterations | 19.45 |

נשווה את זמן ביצוע 3 האיטרציות הראשונות לזמן ביצוע 3 האיטרציות האחרונות.

נבחין כי קיים הבדל משמעותי בין זמני הריצה, ב-3 האיטרציות הראשונות זמני הרימה קצרים משמעותית מ-3 האיטרציות האחרונות.

אני מעריכות כי הסיבה להבדל הנה שאנו לא מאפסים בכל איטרציה מחדש את המבנה. לכן שעושים יותר איטרציות יש יותר תאים בטבלה שמסומנים כ-deleted ולא כ-null . בפונקציית ה-Insert שלנו אנחנו מכניסים איבר בתא שמסומן deleted רק אחרי שעברנו על כל הטבלה לוודא שאין מפתח זהה לזה שאנחנו רוצים להכניס (כיוון שיתכן שהמפתח נמצא בהמש הטבלה), לעומת זאת עבור תא שמכיל- null אנחנו מכניסים את האיבר החדש ומסיימים. לכן ככל שיהיו יותר תאים שמסומנים deleted נקבל זמני ריצה ארוכים יותר עבור ההכנסות ולכן זמני ריצה ארוכים יותר עבור הניסוי.