תוצאות המחקר : יאיר בן דוד

**נושא המחקר :**

בדיקת יעילות זמן הריצה בקלטים שונים על 2 אלגוריתמים  
1. מיון מהיר  
2. מיון ערימה

הקלטים בנויים מ2 חלקים   
K מסמל את גודל תת המערך  
n מסמל את גודל המערך המקורי

הדרישה היא לראות מה יעילות האלגוריתם כתלות בn וk המשימה היא להוציא תת מערך ממויין בגודל k ממערך לא ממוין בגודל n

האלגוריתם בו השתמשתי מצורף בתקייה תחת השם Research.py נכתב בשפה המדהימה פייתון.

מבנה נתונים : ערימה

ב: בגרף מולנו אנחנו רואים :  
ציר Y : את כמות ההשוואות  
ציר X : גודל התת מערך הממויין שאותו ביקשנו למצוא  
וכל קוו מחולק על פי גודל הקלט המקורי שלו n .

ג:

כדי למצוא בעזרת מבנה הנתונים ערימה את תת המערך הממויין שחיפשנו קודם כל בנינו את הערימת מינימום לאחר מכן הוצאנו כל פעם את השורש , האיבר הכי קטן, כגודל התת מערך שרצינו וכך בנינו את התת מערך הממויין   
הבדל בין מקרה הטוב למקרה הגרוע :

בניית ערימה ממערך לא ממויין :

* במקרה הטוב היא תיקח O(n)
  + המקרה הטוב : כל אב באמת יותר קטן מבניו או שהוא יהיה יותר קטן מבני בניו וכך שגרת heapify תיקח רק o(1) פעולות
* מקרה ממוצע וגרוע תיקח (O(nlog(n)
  + המקרה הגרוע : כל אבר לא יהיה קטן יותר מבניו בני בניו ובני בני בניו וכך אין לדעת כמה שגרת heapify תיקח נשערך אותה בסיבוכיות של O(logn)

הוצאת האיבר הכי קטן :

* השגרה היא בעצם להחליף בין האיבר האחרון לראשון, למשוך את האיבר האחרון מהערימה, ולעשות heapify על מנת להביא את האיבר הראשון למקומו.
* זמן ריצה : O(klog(n))
  + ניתוח : אנחנו עושים את השגרה heapify k פעמים .

כמו שאנחנו רואים גם מהגרף וגם מהניתוח התאורתי של זמן הריצה מזהים את היציבות של האלגוריתם, כמעט ולא שמים לב ליכולת של המקרה הרע או הטוב להופיע בכך שהגרף כמעט מתנהג כמו משוואת קוו ישר.

אם נחשב את זה בעצמנו נראה שגם אם נגיע למקרים הגרועים נראה הבדל אבל לא עצום, אם נסתכל על מערך D שגודלו 1000

* במקרה הטוב – 1000
* המקרה הגרוע – בערך 10000
  + בסה"כ פי עשירית מגודל הקלט.

אם נשווה דגימה מתוצאות הניסוי לתוצאות התאורטית לדוגמא על מערך 1000 = D k=100 נראה שהחישובים התאורטיים במקרה הגרוע מביאים 10962 השוואת ובמקרה הטוב 2000 השוואות ומה שיצא לנו בגרף ממש ממצע בין התוצאות התיאורטית שזה מאוד הגיוני לנוכח חוסר העקביות של הקלטים.

ניתוח זה מביא נקודות זכות גדולות בעניין היציבות מפני המקרים הגרועים שאין התדרדרות משמעותית בזמני הריצה.

אלגוריתם מיון מהיר :

ב: בגרף מולנו אנחנו רואים :  
ציר Y : את כמות ההשוואות  
ציר X : גודל התת מערך הממויין שאותו ביקשנו למצוא  
וכל קוו מחולק על פי גודל הקלט המקורי שלו n .

ג:

למציאת תת מערך ממויין בגודל k ביצעתי חיפוש האיבר במיקום ה k הכי קטן ואז שלחתי את המערך שנוצר לאור החיפוש למיון מהיר  
ניתוח זמן הריצה :

* מציאת האיבר הk הכי קטן O(nlog(n))
  + ככול שממשיכים עם חיפוש האיבר k מקטינים את המערך ב2 וכל ריצה משתמשים בשגרה partition O(n) שהיא עוברת על כל המערך ומחלקת אותו לפי אותו איבר, שבתקווה יהיה, k .
  + שימוש בבחירת איבר ציר בשיטה רנדומלית גרמה לחוסר ידעה אמיתית על הסיבוכיות- נראה את זה בניתוח התוצאות עם הגרף.
* מיון על קלט בגודל k במקרה הממוצעO(klog(k))

כמו שאנחנו רואים הגרף מאוד מאוד לא יציב, מוכיחה לנו את השבריריות של האלגוריתם מיון מהיר, בקלטים הגדולים אנחנו ממש רואים שני קיצונים, התחלה במקרה ממש טוב , אמצע במקרה גרוע, והסוף שוב במקרה טוב, ובקלט D בדיוק הפוך התחלה במקרה קטסטרופלי ממשיכים עם מקרה מצויין ואז שוב מחריף המצב אם נשווה דגימה מתוצאות הניסוי לתוצאות התאורטית על מערך 1000 = D k=100 נראה שהחישובים התאורטיים במקרה הגרוע מביאים 10630 השוואת והגרף מראה ממש פחות, עוד נקודה לחוסר יציבות של האלגוריתם

אני אישית מאוד אוהב את האלגוריתם למרות שהוא קצת לא יציב, התחכום שלו נפלא ופשוט ומאוד רלוונטי לצורת חשיבה אנושית.