צילי בומבך 21700148

חלק 1

סעיף א

על מנת לעבד קובץ לוג גדול בצורה יעילה, בחרתי לחלק אותו לטווחים (scopes) של בייטים במקום לפצל אותו לקבצים פיזיים נפרדים. כל תהליך מקבילי מקבל תחום קריאה מוגדר מראש — הוא מתחיל ממיקום מסוים בקובץ, קורא עד לסופו של התחום.
גישה זו חוסכת בזמן ובמשאבי דיסק, מאחר שהיא נמנעת מיצירת קבצים זמניים מיותרים, ומבצעת פחות פעולות קלט/פלט .(I/O) מספר התחומים נקבע לפי מספר הליבות הזמינות במחשב, כך שניתן לנצל בצורה מיטבית את יכולות העיבוד המקבילי של המערכת.

ה. סיבוכיות זמן ומקום

נסמן:
c- מספר הליבות הזמינות במחשב
N- מספר השורות בקובץ
k- מספר סוגי השגיאות
n- הפרמטר הנתון

סיבוכיות זמן
הקוד עובר על כל הקובץ פעם אחת בלבד , אמנם בזכות ה multiprocessing יש מעבר במקביל על חלקים שונים של הקובץ אך הדבר לא משפיע משמעותית על סיבוכיות זמן הריצה שהיא פונקציה של גודל הקובץ.
חילוק לscopes- $\theta(c)$ (זניח)
מעבר על הקובץ- $\theta(N)$ לכל התהליכים יחד, כל שורה עדכון ה counter ב0(1)
מיזוג ה counters -מיזוג של c מילונים בגודל 0(k) - $0(ck)$ (לרב זניח)
מציאת n השכיחים על ידי ערימה -O(klogn)
ברור שN הוא הדומיננטי ביותר
סה"כ

$$\theta(N) + 0\left(\frac{N}{c}\right) + 0(ck) + 0(klogn) = \theta(N)$$

(במקרה שבו c,k <N ובקבצים גדולים זה המקרה)

סיבוכיות מקום
כל תהליך קורא את השורות שלו **שורה-שורה** ,לא טוען את כל הקובץ לזיכרון
שימוש ב ()f.readline שומר רק שורה אחת בזיכרון בכל רגע.
כל תהליך שומר מונה שגודלו O(k)-רשימה של כל קודי השגיאה הייחודיים שהוא נתקל בהם.
עבור c תהליכים נקבל: O (ck), ולאחר המיזוג $\theta(k)$
מציאת n השגיאות עם השכיחות הגבוהה על ידי שימוש בערימה בגודל n: $\theta(n)$
החישוב לא כולל את מקום קובץ שלא נחשב לחלק מסיבוכיות המקום של הקוד
סה"כ:

$$0(1) + 0(ck) + \theta(n) = 0(ck)$$

בהנחה ש n<=k

סעיף ב

4. על מנת לתמוך בdata stream נבצע את הפתרון בצורה הבאה:
ניצור תור כך שכל שורה שנקלטת נכנסת לתור ומחכה לטיפול.
נעבור שורה שורה מהתור לפי הסדר ,נעדכן את הממוצע של השעה המתאימה בבסיס נתונים מקומי ונשמור את התוצאות מקומית.
נקבע חלון זמן כלשהו, בהתאם לקצב כניסת הנתונים ודרישת הדיוק במידע המוצג, שעבורו נעדכן את הממוצעים החדשים שנעשה בהם שינוי בקובץ הפלט האמיתי.
זאת אומרת, עבור כל שורה שנוציא מהתור נבדוק אם במבנה הנתונים המקומי השעה המתאימה כבר קיימת, אם כן נעדכן אות הממוצע שלה, אם לא, נביא את הנתון מקובץ היעד, נעדכן ממוצע ונוסיף למבנה הנתונים הזמני. כשעובר חלון הזמן נעדכן את הקובץ הסופי עם כל הנתונים שבבסיס הנתונים המקומי, נאפס את בסיס הנתונים ואת חלון הזמן ושוב מהתחלה.