Data Streaming Algorithms and Online Learning
תרגיל בית 1

חלק יבש (50 נק')

Sampling

בהרצאה ראינו את אלגוריתם reservoir sampling לדגימה אקראית של איברים כך שלכל איבר הסתברות זהה להידגם. עתה תעסקו בדגימה "בהסתברות לא זהה":

- נתון stream של n איברים (כולם מספרים חיוביים) a1,…an.
- תארו אלגוריתם streaming שידגום מהstream איבר אחד כך שההסתברות לדגימה של כל איבר תהיה פרופורציונית לערך שלו.
- הוכיחו את נכונות הדגימה, מה גודל הsketch?

אלגוריתם Morris

בשאלה זו תעסקו בהרחבה של אלגוריתם מוריס למקרה שיש גם "אירועים-שליליים" ולא רק "חיוביים" – והמטרה הינה לשערך את כמות האירועים האקטיביים בכל רגע, כלומר את הערך של logins פחות logouts (אפשר להניח שיהיו יותר logins מאשר logouts). מקרה כזה נקרה turnstile model.

עומרי הציע לעשות זאת ע"י אחזקה של שני "משערכי מוריס", אחד לספירת אירועים-חיוביים והשני לספירת אירועים-שליליים ואז לחסר ביניהם. אלינוי טוענת שהגישה של עומרי היא בזבוז זיכרון ואומרת "למה צריך לבזבז loglog n ביטים אם היו n אירועים-חיוביים וגם n אירועים-שליליים, שביחד נותנים 0? במקום, בואו נמצא דרך להקטין את הערך של המשערך באיזושהי הסתברות בהינתן אירוע-שלילי":

- תארו אלגוריתם שמתאים להצעה של אלינוי, אשר מרחיב את האלגוריתם של Morris שראינו בכיתה. בפרט: תארו איך מעדכנים את הcounter שאלגוריתם Morris מחזיק במקרה של אירוע-שלילי? מה תעשו במקרה שהcounter מגיע ל0? הוכיחו שהמשערך החדש המתקבל הינו unbiased.
- אם n אירועים-חיוביים הגיעו ואחריהם n אירועים-שליליים, האם באמת הצלחנו להיפטר מהn loglog ביטים כפי שאלינוי רצתה? הסבירו.

חלק רטוב (50 נק')

בחלק זה תממשו את אלגוריתמי AMS לשערוך F2 שנלמדו בהרצאות ו"תרגישו בידיים" את ההתכנסות שלהם, חתימת הזיכרון הנמוכה וכו', אתם יכולים לממש בכל שפה שתרצו.

הוראות הגשה:

- קובץ קוד (אם פייתון – עדיף notebook) שכולל את כל מה שנדרש להרצת הקוד וקבלת התוצאות שלכם עם תיעוד בסיסי (השתמשו ב random seed קבוע כדי לאפשר את שחזור התוצאות שקיבלתם)
- תיאור קצר של התוצאות, מסקנות, מחשבות וכו'

סעיפים – **שימו לב להערות חשובות בסוף המסמך:**

1. ממשו את 3 הגרסאות של האלגוריתם כפי שנלמדו בהרצאה.

2. סמלצו stream סינטטי או השתמשו ב dataset מעניין שתמצאו:
   a. גודל הstream צריך להיות לפחות מיליון elements
   b. מספר ה unique elements צריך להיות לפחות 10 אלף, והתפלגות ההופעות של כל unique element חייבת להיות לא יוניפורמית – למשל סמלצו ערכי אקראיים שונים לfrequencyים של כל unique element

3. <u>AMS Alpha version:</u>
   a. הריצו את המשערכים
   b. עפ"י הערכים של ההרצות השונות- חשבו תוחלת ו normalized-variance של כל משערך

4. <u>AMS Beta and Final:</u>
   a. הריצו את הגרסאות השונות, השתמשו במספר "עותקים" כלהלן:
      1. אלגוריתם Beta version: 10, 25, 50 עותקים של אלגוריתם Alpha version
      2. אלגוריתם Final version: 10 ו50 עותקים של Beta version , כל אחד עם 10, 25 50 עותקים של Alpha version
      ii. עבור כל "מספר עותקים", לפי התיאוריה שנלמדה בהרצאה, מה ה relative error המובטח עבור הסתברות הצלחה של **99%?** (חשבו מה ה delta וה epsilon...)
      iii. שימו לב באלגוריתם AMS להשתמש בפונקציות hash שונות בין ההרצות, אחרת ההרצות יהיו זהות זו לזו – **ראו התייחסות בהערות למטה.**
   b. עפ"י הערכים של ההרצות השונות- חשבו תוחלת ו normalized-variance של כל משערך והראו גרפים של שינוי הדיוק, גודל הזיכרון, התוחלת וה-normalized-variance לפי מספר העותקים בהם כל משערך משתמש
      i. איך באה לידי ביטוי ה unbiasedness של האלגוריתמים כפי שלמדנו?

<u>*הערות חשובות:</u>

- **בשביל לקבל "מובהקות סטטיסטית", הריצו "כל ניסוי משערך" לפחות 50 הרצות שונות**
- השונות המנורמלת של משערך מוגדרת כך: $Normalized\ variance = \frac{Var(\hat{n})}{n}$, כלומר השונות של המשערך מנורמלת בערך האמיתי שניסה לשערך (כאשר n הינו הערך האמיתי).
- **אתם לא צריכים להתעסק בכלל עם פונקציות hash -** הכוונה שתייצרו מספרים אקראיים שכמו מסמלצים ערכים של פונקציות hash אידאליות - הממירות בצורה חד חד ערכית מושלמת כל element לuniform בין 0 ל1 (וכמובן ממירות את כל ההופעות של אותו element לאותו uniform)
- **לגבי זמני הריצה, אין צורך להריץ במשך ימים ולילות, חשבו טוב על האלגוריתמים ובצעו "התאמות" לזירוז הריצה, כל עוד האלגוריתם לא נפגע- הרצה במקביל, הרצה בתצורה ווקטורית וכו'.**
- לגבי הזיכרון- הכוונה להראות איך גודל הזיכרון שבו עשיתם שימוש גדל בין הגרסאות השונות של האלגוריתם. אין צורך בmemory profiling או ניתוח מדוקדק, תראו למשל איך מספר הרגיסטרים גדל, וכו'

בהצלחה!