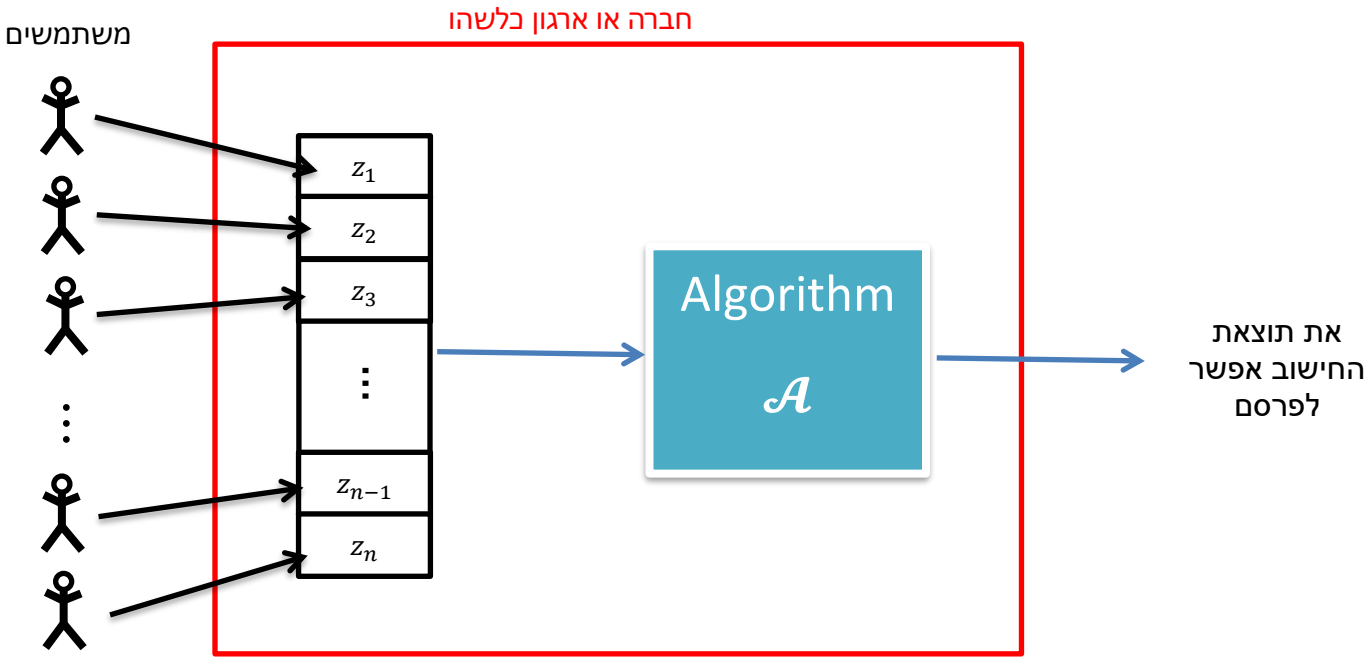
1.12.2022 ***פרטיות מידע***

הרצאה 6: Local Differential Privacy

Textbook: Cynthia Dwork and Aaron Roth. The Algorithmic Foundations of Differential Privacy

מרצה: אורי שטמר

<u>נזכר בסיפור המסגרת בקורס שלנו:</u>

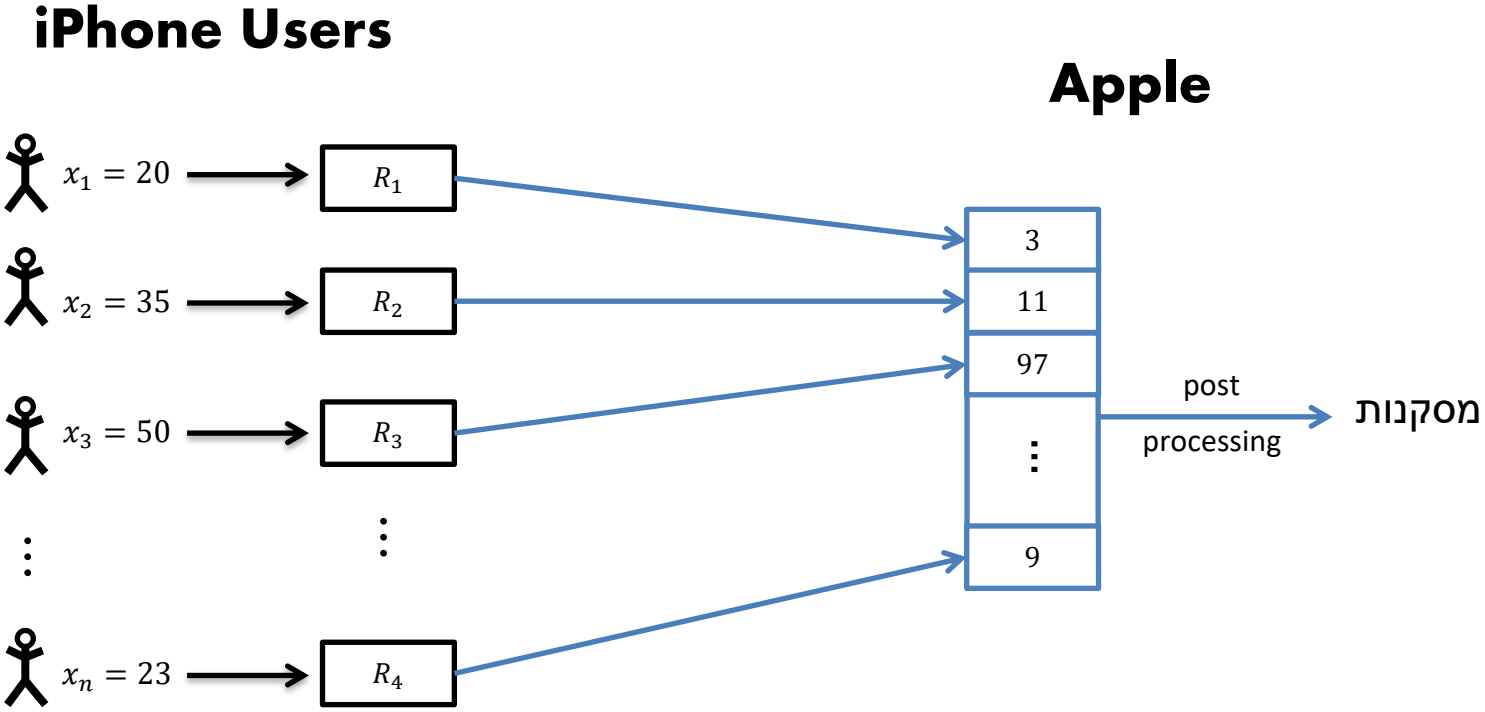


כלומר, עד עכשיו דיברנו על מקרה שבו חברה או ארגון כלשהו (למשל בית חולים) רוצה לנתח את המידע הפרטי שלנו ולפרסם את תוצאת החישוב, באופן כזה שיבטיח שמתוך תוצאת החישוב אי אפשר ללמוד הרבה על הנתונים האישיים של כל אחד מאתנו. אבל בסיפור הזה, אותה חברה שמריצה את החישוב, היא יודעת הכל. כלומר עליה אנחנו סומכים לחלוטין עם המידע שלנו. היא מנתחת את המידע וההגנה היא רק מפני גופים אחרים שיראו את תוצאת החישוב.

מה אפשר לעשות אם אנחנו מעוניינים להגן על הפרטיות שלנו גם בפני החברה/הארגון שמבצעים את החישוב?

נניח שחברה מסויימת (למשל Apple) רוצה לנתח את הנתונים של הלקוחות שלה ולהסיק מהם מסקנות. אם מדובר בנתונים רגישים (מבחינת פרטיות) אז יכול להיות שהלקוחות לא רוצים לגלות לחברה את הנתונים שלהם. איך החברה יכולה לנתח את הנתונים תוך כדי שהיא <u>מבטיחה</u> ללקוחות שהיא לא אוספת את הנתונים שלהם?

הציור עכשיו:



- ישנם n משתמשים, כאשר כל משתמש i מחזיק קלט x_i.
- נסמן $S = (x_1, x_2, \dots, x_n)$. נתייחס אל S בתור "דטהבייס מבוזר"
(כי הוא לא נמצא במקום אחד, אלא כל משתמש מחזיק את השורה שלו אצלו)
- יש שרת שמעוניין לנתח את אוסף הנתונים S.
- השחקנים <u>לא</u> שולחים את הקלטים שלהם לשרת. כל שחקן מפעיל (באופן מקומי) אלגוריתם פרטי על הקלט שלו ושולח לשרת את תוצאת החישוב.

<u>הגדרה:</u> אלגוריתם $R: X \to Y$ נקרא (ε, δ)-randomizer אם לכל זוג קלטים $x, x' \in X$ ולכל פלט $y \in Y$ מתקיים
$$\Pr[R(x) = y] \leq e^{\varepsilon} \cdot \Pr[R(x') = y] + \delta$$

כלומר זאת בדיוק אותה הגדרה כמו של פרטיות דפרנציאלית, חוץ מזה שקבענו את גודל הדטהבייס להיות 1.
לשם פשטות, לאורך כל ההרצאה נניח כי $\delta = 0$.

<u>הגדרה:</u> אלגוריתם $\mathcal{A}: X^n \to W$ מקיים ε-LDP אם הוא ניגש לכל אחד מהקלטים לכל היותר פעם אחת, דרך ε-randomizer בלבד.

<u>הערה:</u> באופן יותר כללי, נרשה לאלגוריתם לגשת k פעמים לקלט מסויים, דרך רנדומייזרים עם פרמטרים $\varepsilon_1, \dots, \varepsilon_k$ כך שמתקיים $\varepsilon_1 + \cdots \varepsilon_k \leq \varepsilon$

בדיקת שפיות: למה ההגדרה הזאת מבטיחה לי פרטיות? (נסו להסביר זאת בעזרת הציור האחרון)

<u>מוטיבציה למודל *LDP*:</u>

- מאפשר ללמוד על כלל האוכלוסיה תוך הבטחת פרטיות חזקה מאוד למשתמשים
- פרטיות נשמרת גם אם השרת חושף את כל הנתונים שלו לפי צו בית משפט, או אם יש פריצה לשרת

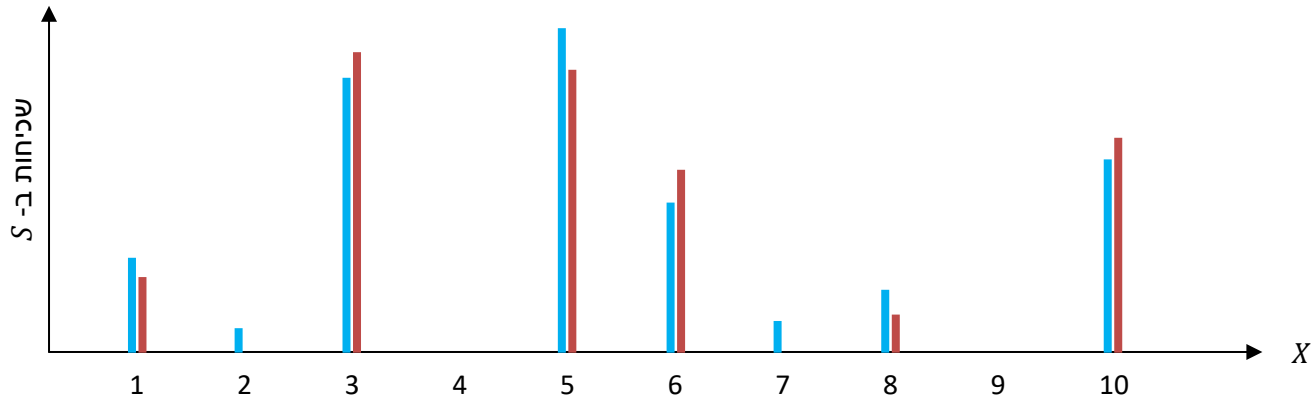
אבל הדברים הטובים האלה לא באים בחינם, כי יש לנו הרבה רעש במערכת (כל משתמש מוסיף רעש) ואנחנו צריכים לראות איך מתמודדים עם זה...

<u>הגדרת הבעיה שנרצה לפתור היום תחת *LDP*</u>

- ישנם n משתמשים, כאשר כל משתמש i מחזיק קלט $x_i \in X$. נסמן $S = (x_1, x_2, \dots, x_n)$.
- <u>המטרה:</u> לכל קלט אפשרי $x \in X$ נרצה להעריך את השכיחות של ב- S, המסומנת כ-

$$f_S(x) = |\{i \in [n] : x_i = x\}| = \text{מספר המופעים של } x \text{ ב } S$$

<u>דוגמה:</u>



נניח שהקווים האדומים מתארים את ההיסטוגרמה של S

נניח שהקווים הכחולים מתארים את ההערכות שאנחנו מקבלים, שנסמנם $\hat{f}_S(\cdot)$.

אנחנו מעוניינים שההערכות האלה יהיו כך ש-

$$\max_{x \in X} \left| \hat{f}_S(x) - f_S(x) \right|$$

יהיה קטן ככל האפשר.

<u>מוטיבציה:</u>

- בעיה טבעית והיו עליה הרבה עבודות
- פתרון לבעיה זו משמש בהרבה מקרים כ- *subroutine* לפתרון בעיות מורכבות יותר
- חברות גדולות בתעשייה משתמשות באלגוריתמים לבעיה זו ולוריאנטים שלה

<u>שאלת זמן ריצה:</u> אם הדומיין X הוא גדול מאוד, אז אלגוריתם יעיל לא יכול לפלוט הערכה $\hat{f}_S(x)$ לכל $x \in X$ באופן מפורש.

(ובאמת באפליקציה של חברות בתעשיה, הדומיין X הוא <u>גדול מאוד</u>)

איך נעקוף את הבעיה?

<u>דרך 1:</u> נתכנן אלגוריתם אשר בסיום הריצה פולט <u>מבנה נתונים</u> המאפשר לחשב את $\hat{f}_S(x)$ לכל $x \in X$. אלגוריתם כזה נקרא *Frequency Oracle*.

כלומר, אלגוריתם כזה לא כותב באופן מפורש את כל הקירובים שהוא חישב, אלא בסיום הריצה הוא נותן לי קופסה שאם אני מזין לה איבר x היא מחזירה לי הערכה $\hat{f}_S(x)$.

<u>דרך 2:</u> נתכנן אלגוריתם אשר בסיום הריצה פולט רשימה קצרה של איברים $L \subseteq X$ (נניח באורך לכל היותר n) ובנוסף פולט הערכה $\hat{f}_S(x)$ לכל $x \in L$. המוסכמה היא שמעריכים $\hat{f}_S(x) = 0$ לכל $x \notin L$. אלגוריתם כזה נקרא אלגוריתם למציאת *heavy-hitters*.

בדיקת שפיות: נסו להבין את ההבדלים בין הגישות

<u>הערות:</u>

(1) בשתי האפשרויות אנו מגדירים את <u>השגיאה</u> של האלגוריתם כ-
$$\max_{x \in X} \left| \hat{f}_S(x) - f_S(x) \right|$$
(2) כל אלגוריתם למציאת *heavy-hitters* הוא בפרט *Frequency Oracle* ולכן אפשרות 1 היא (אולי) קלה יותר להשגה.
(3) ללא שיקולי זמן ריצה האפשרויות זהות (כי אם יש לי *Frequency Oracle* אוכל לשאול אותו אחד אחד על כל איבר בדומיין עד שאמצא את הכבדים)

<u>תוכנית להמשך ההרצאה:</u>

(1) נראה רדוקציה מבעיית ה *heavy-hitters* לבעיית ה *Frequency Oracle* (רדוקציה משמרת יעילות). כלומר נראה כיצד בהינתן אלג' *Frequency Oracle* ניתן לבנות אלג' למציאת *heavy-hitters*.
(2) נראה כיצד לבנות *Frequency Oracle* המשמר *LDP*.

<u>חלק 1: *Frequency Oracle* $\Longrightarrow$ *Heavy-hitters*</u>

<u>משפט 1:</u> אם קיים *Frequency Oracle* עם שגיאה τ המשמר ε-LDP אזי קיים אלג' למציאת *heavy-hitters* עם שגיאה $O(\tau)$ המשמר ε-LDP עם (כמעט) אותו זמן ריצה, זיכרון, ותקשורת.

אנחנו נראה הוכחה של משפט קל יותר:

<u>משפט 2:</u> אם קיים *Frequency Oracle* יעיל עם שגיאה τ המשמר ε-LDP אזי קיים אלג' יעיל למציאת *heavy-hitters* עם שגיאה 2τ המשמר $\varepsilon \cdot \log|X|$-LDP.

<u>הוכחה (כמעט מלאה) למשפט 2:</u>

<u>סימונים:</u>

(1) נסמן ב- $\mathbb{O}$ את אלג' ה *Frequency Oracle* המובטח המשמר ε-LDP ומשיג שגיאה τ
(2) ישנם n משתמשים כאשר לכל $i \in [n]$, משתמש i מחזיק קלט $x_i \in X$. נסמן $S = (x_1, \dots, x_n)$.
(3) תהי $h: X \to [T]$ פונקציית *hash* (ידועה לכולם) הממפה איברים בדומיין X לטווח קטן יותר בגודל T

<u>הנחה משפטת:</u> אין התנגשויות בפונקציית ההאש מתוך S. כלומר לכל $x_i \neq x_j \in S$ מתקיים $h(x_i) \neq h(x_j)$.

<u>מטרה נוכחית:</u> אנחנו רוצים להשתמש ב- $\mathbb{O}$ על מנת לזהות את כל האיברים שמופיעים לפחות 2τ פעמים ב- S, מבלי לבצע חיפוש ממצה על X.

<u>אבחנה:</u> ברגע שנזהה את כל האיברים האלו, נוכל להעריך את השכיחויות שלהם בעזרת $\mathbb{O}$, כלומר נקבל אלג' למציאת *heavy-hitters* כפי שרצינו.

<u>הרעיון:</u> במקום לבצע חיפוש ממצה על X היינו רוצים לבצע חיפוש ממצה על $[T]$, שהוא קטן משמעותית. אבל איך?

<u>ראייה לעתיד:</u> אנחנו נפעיל את $\mathbb{O}$ כמה פעמים על דטהבייסים שונים (כלומר לא נפעיל את $\mathbb{O}$ ישירות על S).

- נקבע איבר $x^* \in X$ המופיע לפחות 2τ פעמים ב- S. כלומר $f_S(x^*) \geq 2\tau$.
  המטרה שלנו היא לזהות את x^*.
  נסמן $t^* = h(x^*)$.

- לכל $\ell \in [\log|X|]$ נגדיר דטהבייס מבוזר חדש: $S_\ell = \big((h(x_1), x_1[\ell]), \dots, (h(x_n), x_n[\ell]) \big)$
  כאשר $x_i[\ell]$ הוא הביט ה-ℓ-י בייצוג הבינארי של x_i.
  שימו לב: בגלל ש- h ידועה לכולם, כל משתמש i יכול לחשב את השורה שלו ב- S_ℓ בעצמו. כלומר התחלתי עם דטהבייס מבוזר S וייצרתי ממנו $\log|X|$ דטהבייסים מבוזרים S_ℓ.

- מתקיים ש- x^* מופיע ב- S לפחות 2τ פעמים, ולכן $(t^*, x^*[\ell])$ מופיע לפחות 2τ פעמים ב- S_ℓ (לכל ℓ)

- מצד שני, מכיוון שהנחנו שאין התנגשויות ב- h, אזי $(t^*, 1 - x^*[\ell])$ מופיע בדיוק 0 פעמים ב- S_ℓ.
  כלומר יש לנו כאן פער של לפחות 2τ

לכן נוכל לזהות את הביט ה-ℓ-י של x^* ע"י השוואת השכיחויות של $(t^*, 1), (t^*, 0)$ ב- S_ℓ בעזרת ה *Frequency Oracle* שלנו.

<u>האלגוריתם:</u>
לכל $t \in [T]$ נבנה איבר $\hat{x}^{(t)}$ באופן הבא:
- לכל $\ell \in [\log|X|]$
  - נקבל מ- $\mathbb{O}(S_\ell)$ הערכות $\hat{f}_{S_\ell}(t,1)$, $\hat{f}_{S_\ell}(t,0)$ ונקבע את הקואו' ה-ℓ-ית של $\hat{x}^{(t)}$ להיות
    $$\hat{x}^{(t)}[\ell] \leftarrow \underset{b \in \{0,1\}}{\operatorname{argmax}} \{\hat{f}_{S_\ell}(t,b)\}$$

לפי הדיון הקודם, כאשר נעבור על $t = t^*$ נקבל שהאיבר שנשחזר יהיה $\hat{x}^{(t)} = x^*$.

כלומר, האלגוריתם הנ"ל מחזיר רשימה של T איברים, המכילה את <u>כל</u> האיברים x כך ש- $f_S(x) \geq 2\tau$.

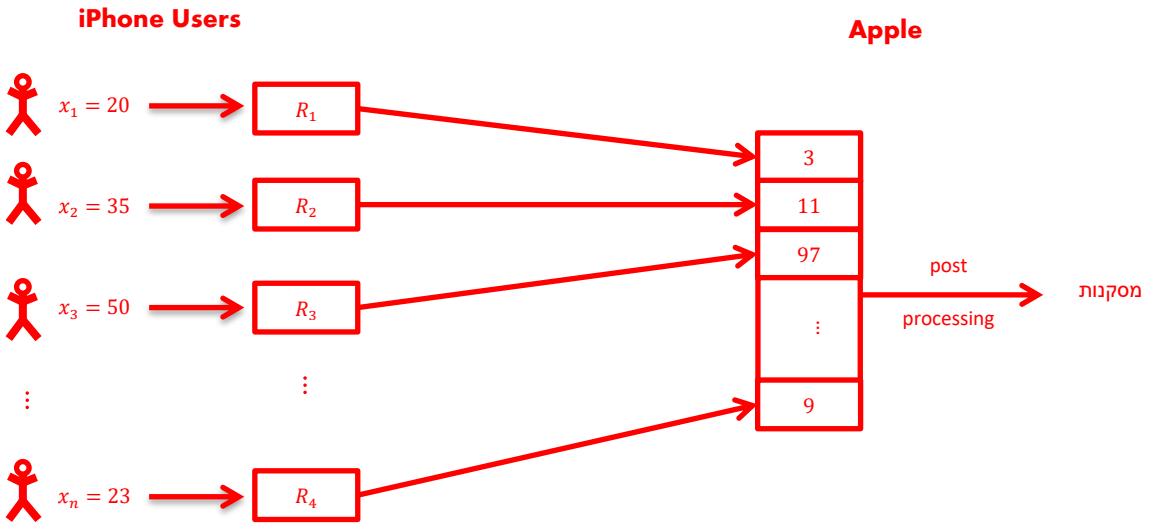
<u>עובדה:</u> כדי שההנחה המפשטת שלנו תתקיים (אין התנגשויות) מספיק לקחת $T \gtrsim n^2$ ולכן נקבל אלגוריתם בזמן ריצה (בערך) n^2 כפול זמן התגובה של $\mathbb{O}$ פר שאילתא.

<u>מה לגבי פרטיות?</u> הרצנו במקביל $\log|X|$ פעמים את ה *Frequency Oracle* שלנו. כל הפעלה כזאת משמרת ε-LDP. סה"כ קיבלנו אלג' המשמר $\varepsilon \cdot \log|X|$-LDP לפי הגדרת המודל (או לפי קומפיזיציה).

חלק 2: בניית *Frequency Oracle* המשמר ε-LDP

כשהראנו את הרדוקציה מ *heavy-hitters* ל *frequency oracle* לא באמת היינו צריכים להתמודד עם הרעש שיש במודל *LDP*, כי פשוט ירשנו גם את הבטחת הפרטיות וגם את הבטחת הדיוק מהקופסה השחורה שהשתמשנו בה. אבל עכשיו כשאנחנו רוצים לבנות *Frequency Oracle* אז כן נצטרך להתמודד עם הרעש שיש במערכת.

צעד אחורה: איך בכלל אפשר לעשות משהו במודל *LDP* עם כל הרעש??



חימום: נבנה *Frequency Oracle* עבור דומיין בגודל 2: $X = \{\pm 1\}$

- דטהבייס מבוזר $S = (x_1, \dots, x_n)$ כאשר כל משתמש i מחזיר את $x_i \in \{\pm 1\}$.
- מטרה: להעריך את מספר ה-1-ים ב- S, המסומן כ- $f_S(1)$.

נשתמש באלגוריתם הבא שנקרא Randomized Response:

Algorithm $R(\cdot)$
קלט: $x \in \{\pm 1\}$ פלט: • נחזיר את x בהסתברות $\frac{e^\varepsilon}{e^\varepsilon+1} \approx \frac{1}{2} + \varepsilon$ • נחזיר את $-x$ בהסתברות $\frac{1}{e^\varepsilon+1} \approx \frac{1}{2} + \varepsilon$

מיידי: אלגוריתם $R(\cdot)$ עונה על הגדרת ε-randomizer

הפרוטוקול:

- מכל משתמש i נקבל הודעה $y_i \leftarrow R(x_i)$.
- לאחר מכן נחזיר

$$\frac{1}{4\varepsilon}\left(2\varepsilon n + \sum_{i\in[n]} y_i\right)$$

ניתוח השגיאה:

$$\mathbb{E}\left[\sum_{i\in[n]} y_i\right] = \sum_{i:x_i=1} \mathbb{E}[y_i] + \sum_{i:x_i=-1} \underbrace{\mathbb{E}[y_i]}_{=(-1)\cdot\left(\frac{1}{2}+\varepsilon\right)+1\cdot\left(\frac{1}{2}-\varepsilon\right)=-2\varepsilon}$$

$$= \sum_{i:x_i=1} 2\varepsilon + \sum_{i:x_i=-1} -2\varepsilon$$

$$= 2\varepsilon \cdot f_S(1) + (-2\varepsilon) \cdot f_S(-1)$$

$$= 2\varepsilon \cdot f_S(1) + (-2\varepsilon) \cdot \left(n - f_S(1)\right) = 4\varepsilon \cdot f_S(1) - 2\varepsilon n$$

כלומר, תוחלת השגיאה של הפרוטוקול שלנו היא אפס.

כדי לקבל חסם הסתברותי על השגיאה, נוכל להשתמש בחסם הופדינג.

תזכורת – חסם הופדינג: יהיו $X_1, \dots, X_n$ מ"מ ב"ת המקבלים ערכים בקטע $[a,b]$. אזי לכל $t \geq 0$ מתקיים:

$$\Pr\left[\left|\sum_i X_i - \mathbb{E}\left[\sum_i X_i\right]\right| \geq t\right] \leq 2 \cdot \exp\left(\frac{-2 \cdot t^2}{n \cdot (b-a)^2}\right)$$

במקרה שלנו עבור $\sum y_i$ מתקיים $a = -1$, $b = 1$ ולכן נקבל ש- $\sum y_i$ סוטה מהתוחלת שלו ביותר מ $\sqrt{2n \ln\left(\frac{2}{\beta}\right)}$ בהסתברות לכל היותר β. ולכן בהסתברות לפחות $(1-\beta)$ נקבל שהשגיאה היא לכל היותר

$$\frac{1}{4\varepsilon} \cdot \sqrt{2n \ln\left(\frac{2}{\beta}\right)}$$

נגמר החימום. עכשיו אנחנו רוצים לבנות *Frequency Oracle* עבור דומיין X גדול.

בנייה כללית של *Frequency Oracle*

- דטהבייס מבוזר $S = (x_1, \dots, x_n)$ כאשר משתמש i מחזיק $x_i \in X$
- מטרה:

לכל $x \in X$ אנחנו רוצים להיות מסוגלים לחשב הערכה $\hat{f}_S(x)$ ל- $f_S(x)$ = מספר המופעים של x ב- S.

סימון: תהי $Z \in \{\pm 1\}^{|X| \times n}$ מטריצה שנבחרה באקראי באופן אחיד, ונניח ש- Z ידועה לכולם.

הפרוטוקול:

- האלגוריתם בצד של המשתמשים (כלומר תיאור ה ε-randomizer):
  - משתמש i מחזיק קלט x_i
    1) שולף ביט $Z[x_i, i]$ מהמטריצה
    2) שולח לשרת *randomized response* של הביט $Z[x_i, i]$, כלומר:

$$\text{שולח } y_i = Z[x_i, i] \text{ בהסתברות } \frac{1}{2} + \varepsilon$$

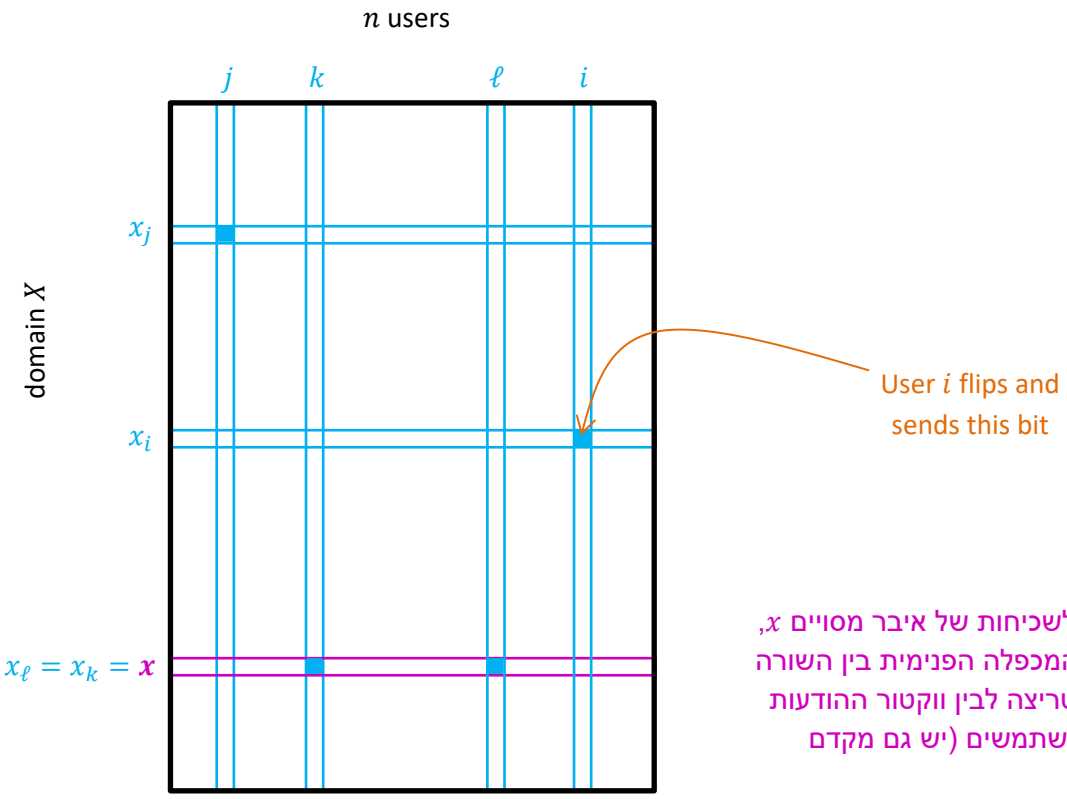
$$\text{שולח } y_i = -Z[x_i, i] \text{ בהסתברות } \frac{1}{2} - \varepsilon$$

(בדיוק כמו בחימום שעשינו, האלגוריתם הזה עונה על הגדרת ε-randomizer)

- האלגוריתם בצד של השרת:
  בהינתן שאילתא $\boldsymbol{x} \in X$, חשב והחזר

$$\hat{f}_S(\boldsymbol{x}) = \frac{1}{2\varepsilon} \sum_{i \in [n]} y_i \cdot Z[\boldsymbol{x}, i]$$

בציור:



כדי לחשב הערכה לשכיחות של איבר מסויים x, השרת מחשב את המכפלה הפנימית בין השורה שמתאימה ל x במטריצה לבין ווקטור ההודעות שהשרת קיבל מהמשתמשים (יש גם מקדם נירמול)

ניתוח הפרוטוקול:

$$\mathbb{E}\left[\sum_{i\in[n]} y_i \cdot Z[x,i]\right] = \sum_{i:x_i=x} \underbrace{\mathbb{E}\left[y_i \cdot Z[x,i]\right]}_{=2\varepsilon} + \sum_{i:x_i\neq x} \underbrace{\mathbb{E}\left[y_i \cdot Z[x,i]\right]}_{=0} = 2\varepsilon \cdot f_S(x)$$

כלומר, שוב תוחלת השגיאה של הפרוטוקול שלנו היא אפס.

כמו קודם, $\sum y_i \cdot Z[x,i]$ הוא סכום של n מ"מ ב"ת ± 1. הופדינג מבטיח שבהסתברות לפחות $(1-\beta)$ הסכום קרוב לתוחלת שלו עד כדי $\sqrt{2n\ln\left(\frac{2}{\beta}\right)}$. במקרה זה נקבל שהטעות שלנו היא לכל היותר

$$\frac{1}{2\varepsilon} \cdot \sqrt{2n\ln\left(\frac{2}{\beta}\right)}$$

שאלה לאינטואיציה:
איך מתנהגת ההערכה הנ"ל עבור x אשר איננו מופיע בדטה בכלל, כלומר $f_S(x) = 0$?
איך היא מתנהגת עבור x המקיים $f_S(x) = n$ בהנחה שקיים כזה ?

שאלה: איך מתחזקים את המטריצה הענקית Z ?

אבחנה: הבטחת הפרטיות בפרוטוקול שלנו לא התבססה על אופן ייצור המטריצה Z. כלומר, לא משנה מהי Z או איך היא נקבעה, הפרוטוקול מבטיח ε-LDP (כי כל משתמש עושה *flip* כמעט אקראי לביט בודד...)

לכן, בפועל אפשר להחליף את המטריצה הזאת באיזושהי פונקציה "שנראית אקראית" וזה לא פוגם בהבטחת הפרטיות. זה אולי יכול לפגום בהבטחת הנכונות, אבל בפועל זה יעבוד. למעשה, ישנן דרכים יעילות לתחזק את המטריצה הזאת בזול מבלי לפגום בהבטחת הנכונות (אבל לא נדבר על זה).