בעיית Equality

לאליס מחרוזת $x \in \{0,1\}^n$ ולבוב מחרוזת $y \in \{0,1\}^n$. אליס ובוב רוצים לבדוק אם המחרוזות שוות, שהמטרה היא, לשלוח כמה שפחות ביטים.

אלגוריתם להתאמת מחרוזות מבוסס פולינומים:

אליס ובוב מחזיקים $A = (a_0, \dots, a_n)$ ו-$B = (b_0, \dots, b_n)$ בהתאמה.

- אליס בונה את הפולינום $R_A(X) = \sum_{i=1}^{\{n-1\}} a_i X^i$.
- אליס מוציאת ראשוני q כך ש-$q > \alpha n$.
- אליס מגרילה מספר t כך ש-$0 \le t < q$ בהתפלגות אחידה ומחשבת את $R_A(t) \ mod \ q$.
- אליס שולחת לבוב את q, את t ואת התוצאה של $R_A(t) \ mod \ q$.
- בוב בונה את הפולינום $R_B(X) = \sum_{i=0}^{n-1} b_i X^i$.
- בוב בודק האם $R_B(t) \ mod \ q$ שווה לתוצאה של $R_A(t) \ mod \ q$ שקיבל מאליס.
- אם המספרים שווים הוא מחזיר 'כן', אחרת 'לא'.

<u>הערות:</u>
מיצאת ראשוני q כך ש-$q > \alpha n$:
הגריל מספר בין αn ל-$(\alpha + 1)n$). בדוק אם הוא ראשוני (עולה לכל היותר n פעולות חילוק).
אם ראשוני – סיימנו, אחרת הגרל שוב מספר.
צפיפות הראשוני היא $\frac{1}{\log n}$ ולכן סך הכל להגריל מספר ראשוני הוא לכל היותר $n \log n$. פורמלית: יהי $p(n)$ מספר הראשוניים בקבוצה $\{1, \dots , n\}$, אז $\lim_{n\to\infty} \frac{n}{p(n)} = \ln n$.

המספרים $R_A(t)$, $\mathrm{R_B}(t)$ הם מספרים גדולים ויקרים לחישוב. לא מחשבים את המספר בשום שלב באופן ישיר. אחרי כל פעולת כפל מבצעים $mod \ q$, לכן המספר תמיד מיוצג ע"י $\log q$ ביטים. ז"א מחשבים $R_A(X) = \sum_{i=0}^{n-1}(b_i X^i \ mod \ q)$.

<u>סה"כ זמן ריצה</u> - $n \log n$ בגודל הקלט.
גודל הקלט $\log q$.
פעולות חיבור, כפל וכו' - $O(1)$. חישוב הפולינום $O(n)$.
הגרלת מספר ראשוני $n \log n$.

<u>מה המספר הביטים שנשלחים:</u>
$q, t < q, R_A(t) \ mod \ q$
ובסך הכל $3 \log q$.

<u>תזכורת:</u>
לפולינום מדרגה לכל היותר n יתכנו n שורשים לכל היותר מודולו q.
לדוגמא, נבחן את הפולינום $p(X) = x + x^2$ מודולו 5.
$p(0) = 0, \quad p(1) = 2, \quad p(2) = 1, \quad p(3) = 2, \quad p(4) = 0 \qquad mod \ 5$
דרגת הפולינום =2, מספר השורשים 2.

טענה: לאלגוריתם הנ"ל יש <u>טעות חד כיוון</u> של $1/\alpha$ לכל היותר.

מה הכוונה טעות חד צדדית כאן?
אם A ו-B מחוזות שוות, אז האלגוריתם תמיד צודק.
אם A ו-B שונות אז האלגוריתם טועה בהסתברות $1/\alpha$ לכל היות.

<u>הוכחה:</u>
אם A ו-B מחרוזות שוות, אז $R_A(X) = R_B(X)$ ולכן לכל ראשוני q ומספר $0 \leq t < q$ יתקיים $R_A(t)\ mod\ q = R_B(t)\ mod\ q$ ולכן יוחזר 'כן' בהסתברות 1.
אם A ו-B שונות, אז נסתכל על הפולינום ההפרש $D(X) = R_A(X) - R_B(X) = \sum_{i=0}^{n-1}(a_i - b_i)X^i$.
דרגתו של $D(X)$ הוא קטנה מ-n.
בהינתן ראשוני $q > \alpha n$ פולינום $D(X)$ שונה מפולינום האפס (אחרת המחרוזות שוות) ההסתברות לטעות היא ההסתברות שאליס מגרילה t כל ש-$D(t) \equiv 0\ mod\ q$.
כלומר אליס מגרילה מספר שהיא שורש של פולינום ההפרש מודולו q.
כמה מספרים כאלה יש? אפשר לספור לפי התזכורת:

$$pr[\text{אליס מגרילה שורש}] = pr\left[\frac{Alice\ chooses\ a\ root}{numbers\ Alice\ can\ choose}\right] = \frac{n}{q}$$

ובסך הכל קיבלנו:

$$\Pr[\text{האלגוריתם טעה}] = \Pr[\text{אליס מגרילה שורש}] = \frac{n}{q} < \frac{n}{\alpha n} = \frac{1}{\alpha}$$

כנדרש.

<u>שאלות למחשבה:</u>

איך מורידים את מסר הביטים הנשלחים אך שומרים על אותה רמת טעות?

האם אשר לשפר את הדיוק של האלגוריתם בלי לשנות את זמן הריצה?

בעיית התאמת מחרוזות

בסעיפים הבאים נעזר באלגוריתם הנ"ל בכדי להציע אלגוריתם להתאמת המחרוזות.
תזכורת: בבעיית התאמת המחרוזות נתונות שתי מחרוזות:
$\langle P[1], P[2], \dots, P[n] \rangle$, $\langle T[1], T[2], \dots, T[m] \rangle$
נניח כי $m > n$. יש להחזיר קבוצת האינדקסים S כך ש- $j \in S$, אם"ם
$\langle P[1], P[2], \dots, P[n] \rangle = \langle T[j+1], T[j+2], \dots, T[j+n] \rangle$.

נציע את האלגוריתם הבא:

1. $S \leftarrow \emptyset$
2. Find prime q suck that $q > 4mn$
3. Randomly generate $0 \leq t < q$
4. Build Polynomial $R(X) = \sum_i^n P[i+1]X^i$ From $\langle P[1], \dots, P[n] \rangle$
5. Calculate $\sigma = R(t) \ mod \ q$
6. For all $0 \leq j \leq m - n$
    a. Build polynomial $Q_j(X) = \sum_{i=0}^{n-1} P[j+i+1]X^i$ from T
    b. Calculate $\tau_j = Q_j(t) \ mod \ q$
    c. If $\tau_j = \sigma$ then $S \leftarrow S \cup \{j\}$
7. Return S

<u>סעיף א</u>

בשלב j באלגוריתם אנחנו בונים את הפולינום $Q_j(X)$.
הוכיחו כי $Q_j(X) = XQ_{j+1}(X) - T[j+n+1]X^n + T[j+1]$.

<u>סעיף ב</u>

הסבירו איך ניתן לחשב ביעילות את τ_j מתוך τ_{j+1} ו-$t^n \ mod \ q$.

<u>סעיף ג</u>

הציעו מימוש לאלגוריתם להתאמת מחרוזות שמתואר בתרגיל זה שהסיבוכיות שלו היא $O(n+m)$.

<u>סעיף ד</u>

הוכיחו כי האלגוריתם טועה בהסתברות לכל היותר 1/4.

תשובות

<u>סעיף א</u>

$$Q_j(X) = \sum_{i=0}^{n-1} T[j+i+1] = T[j+1]X^0 + \sum_{i=1}^{n-1} T[j+i+1]X\text{^}i$$

$$Q_{j+1}(X) = \sum_{i=0}^{n-1} T[j+i+2]X^i$$

$$XQ_{j+1}(X) = \sum_{i=0}^{n-1} t[j+i+2]X^{i+1} = T[j+n+1]X^n + \sum_{i=0}^{n-2} T[j+i+2]X^{i+1}$$

$$= T[j+n+1]X^n + \sum_{i=0}^{n-1} T[j+i+1]X^i$$

ולכן $Q_j(X) = XQ_{j+1}(X) - T[j+n+1]X^n + T[j+1]$.
בניתוח זמן הריצה בסעיפים הבאים נניח שהפעולות על מספר המיוצג על ידי כמת לוגריתמים של ביטים לוקחות $O(1)$. זוהי הנחה סבירה כי במחשבים יש חומרה יעודית שמבצעת פעולות אריתמטיות על מילים ולא ברמת הביטים. ניתן לנתח בצורה יותר מדוייקת ולקבל תוספת סיבוכיות של $O(\log^k n)$ עבור k קבוע.

<u>סעיף ב</u>

בהינתן $\tau_{j+1} = Q_{j+1}(t) \bmod q$ נכפיל ב-t, נחסיר את $T[j+n+1](t^n \bmod q)$ ונוסיף את $T[j+1]$. נפעיל $\bmod q$ על התוצאה ונקבל את τ_j. מכיוון שאורך הייצוג של כל המספרים בחישוב הוא $O(\bmod q)$, זמן החישוב הוא $O(1)$.

<u>סעיף ג</u>

בשלב הראשון נחשב את $t^i \bmod q$ עבור $i = 1, \dots, n$.
את שלב 6 באלגוריתם נעשה בסדר יורד, כלומר מ-$j = m-n$ עד $j = 1$ ולכן יהיה ניתן להשתמש בסעיף ג'.
שלב 1-3 באלגוריתם $O(1)$.
שלב 4 באלגוריתם $O(n)$.
שלב 5 באלגוריתם $O(n)$.
שלב 6 באלגוריתם: חישוב $O(n) - \tau_{m-n}$, חישוב $O(m) - \tau_0, \dots, \tau_{m-n-1}$.
ולכן קיבלנו $O(n+m)$.

<u>סעיף ד</u>

נחשב את המספר ה-t-ים שיגרמו לאלגוריתם לטעות. האלגוריתם טועה אם הוא בחר t כך שבאחד מ-$m-n+1$ הבדיקות מתקיים $R(t) \bmod q = Q_j(t) \bmod q$. לכל j ישנם לכל היותר n מספרים t כנ"ל. לכן, בסך הכל, יש לכל היותר $(m-n-1)n$ מספרים t שיגרמו לאלגוריתם לטעות $mn >$ וההסתברות לשגיאה היא לכל היותר:

$$\frac{mn}{q} < \frac{mn}{4mn} = \frac{1}{4}$$

<u>הצעה נוספת:</u> אפשר להשתמש בחסם האיחוד עבור הסתברויות ולרשום:

$$(m-n)\left(\frac{1}{\alpha}\right) \le (m-n)\left(\frac{1}{4m}\right) \le \frac{m}{4m} = \frac{1}{4}$$