שאלות עם פתרונות

תזכורת של סימונים:

נסמן ב-T טקסט – מחרוזת תווים $T_1T_2...T_n$ באורך n.

נסמן ב-P תבנית – מחרוזת תווים $P_1P_2...P_m$ באורך m. בד"כ $m << n$.

בהנתן תבנית P וטקסט T, האלגוריתם KMP(T,P) מוצא את מופעי התבנית בתוך הטקסט.

בהנתן תבנית P, פונקצית כשלון $\pi : \{ 1,2,...,m \} \rightarrow \{ 0,1,...,m\text{-}1 \}$ מוגדרת כדלהלן:

$$\forall\ 1 \leq i \leq m \qquad \pi(i) = \mathrm{Max}\ \{\ j\ |\ j < i\ \wedge\ P_1P_2...P_j \sqsupset P_1P_2...P_i\ \}$$

פונקצית הכשלון משמשת ככלי עזר באלגוריתם KMP.

בהנתן תבנית P וטקסט T, פונקצית התאמה $\tau : \{ 1,2,...,n \} \rightarrow \{ 0,1,...,m \}$ מוגדרת כדלהלן:

$$\forall\ 1 \leq i \leq m \qquad \tau(i) = \mathrm{Max}\ \{\ j\ |\ P_1P_2...P_j \sqsupset T_1T_2...T_i\ \}$$

פונקצית ההתאמה ניתנת לחישוב תוך כדי ריצת האלגוריתם KMP ואוגרת מידע שימושי לגבי מספר ומיקום של מופעי התבנית בתוך הטקסט.

1. נתונה מחרוזת T באורך n. תאר אלגוריתם בעל סיבוכיות טובה ביותר המוצא מחרוזת x באורך מקסימלי כך ש- $T=xyx$ ו y מחרוזת באורך 10 לפחות.

פתרון:

רעיון:

למעשה, יש למצוא מחרוזת x אשר הינה רישא של T וגם סיפא של T. אורכה של x הוא המקסימום האפשרי שלא עולה על $\frac{n-10}{2}$ (נובע מהדרישה על האורך של y).

האלגוריתם:

1. **__חשב__** פונקציה π עבור T
2. $q \leftarrow \pi(n)$
3. **__כל עוד__** $q > \frac{n-10}{2}$ **__בצע__**
   a. $q \leftarrow \pi(q)$
4. **__החזר__** $T_1T_2...T_q$

סיבוכיות: $O(n)$ (חישוב π ומעבר על כמה q-ים).

2. נאמר שטקסט T הוא בעל מחזור שלם x אם ורק אם $T=x^i$ עבור $i>0$ שלם, ואז נאמר <u>שריבוי</u> המחזור הוא i.
נתון טקסט T באורך n. הצע אלגוריתם לבדיקת קיום מחזור שלם ב-T: על האלגוריתם להחזיר אורך המחזור המינימלי שריבויו גדול מ-1, ואם לא קיים להחזיר 0.

פתרון:

רעיון:

נניח ש- $T=x^i$. נבחן את ההתנהגות של פונקציה π של T. אם $|x| = k$ ולא קיים מחזור קצר מ-k, אז $\pi(k+1) \geq 1$, $\pi(k+2) \geq 2$, וכך הלאה. אחר כך $\pi(2k) = k$, $\pi(2k+1) = k+1$.
כלומר, לכל $2k \leq j \leq n$ מתקיים $\pi(j) = j-k$. ז"א במערך המכיל את π החל מאינדקס 2k ועד הסוף (אינדקס n) מופיעה סדרה עולה ממש.
מצד שני, אם לפונקציה π של T קיימת התכונה הנ"ל, אזי בוודאות יש ב-T מחזור בגודל k.

האלגוריתם:

1. <u>**חשב**</u> פונקציה π עבור T
2. $k \leftarrow n - \pi(n)$
3. <u>**אם**</u> $k = n$ <u>**או**</u> $n \bmod k \neq 0$ <u>**החזר**</u> 0
4. <u>**לכל**</u> j <u>**החל**</u> מ- 2k <u>**וכלה**</u> ב- n <u>**בצע**</u>
   a. <u>**אם**</u> $\pi(j) \neq j - k$ <u>**החזר**</u> 0
5. <u>**החזר**</u> k

סיבוכיות: $O(n)$.

3. $P=P_1\ldots P_m$ ו $T=T_1\ldots T_n$ מחרוזות מעל $\sum$. נאמר ש- P מופיעה ב- T בהיסט s עם שיבוש אחד אם השוויון $P[j]=T[s+j]$ מתקיים עבור כל j $1\le j\le m$ פרט לאחד בדיוק. תארו אלגוריתים יעיל ככל האפשר למציאת כל הערכים s $(0\le s\le n\text{-}m)$ עבורם P מופיע ב- T בהיסט s עם שיבוש אחד.

פתרון:

רעיון:

נשנה את האלגוריתם של KMP באופן הבא: נוסיף משתנה בו נשמור מידע לגבי אי התאמות שהיו. כל עוד קיימת התאמה מלאה בהשוואת רישא של P עם T, נרשה אי התאמה בתו אחד. אם כבר היתה אי התאמה בתו אחד, אז נדרוש התאמה מלאה בהמשך.
כאשר נצטרך לסגת, נאפס את המשתנה ברגע שנעבור את המקום בו היתה אי התאמה של תו אחד.

האלגוריתם:

1. **חשב** פונקציה π עבור P
2. $q \leftarrow 0$
3. $error \leftarrow 0$
4. **לכל** i **החל** מ- 1 **וכלה** ב- n **בצע**
   a. **כל עוד** $q > 0$ **וגם** $T_i \neq P_{q+1}$ **וגם** $error > 0$ **בצע**
      i. $q \leftarrow \pi(q)$
      ii. **אם** $q < error$ **בצע** $error \leftarrow 0$
   b. **אם** $T_i = P_{q+1}$ **בצע** $q \leftarrow q + 1$
   c. **אחרת אם** $error = 0$ **בצע**
      i. $q \leftarrow q + 1$
      ii. $error \leftarrow q$
   d. $\tau(i) \leftarrow q$
   e. **אם** $q = m$ **בצע** $q \leftarrow \pi(q)$
5. **החזר** τ

סיבוכיות: $O(n + m)$.

4. תארו אלגוריתם יעיל ככל האפשר המחשב לכל k ($1 \le k \le n$) את מספר הרישות הלא ריקות של P שהן סיפות של $T_k=T[1..k]$.

פתרון:

רעיון:

נשתמש בפונקציות π עבור P ו-τ עבור T: מאחר ו-$\pi(i)$ הוא אורך הרישא המקסימלי (ולא טריוויאלי) של P שהוא גם סיפא של $P[1..i]$, אזי ניתן ע"ס הפונקציה הזו לחשב לכל $1 \le i \le m$ את מספר הרישות (הלא ריקות) של P שהם סיפות של $P[1..i]$. נקרא לזה פונקציה N_P.
כעת ניתן לנצל את הפונקציה τ: הרי $\tau(i)$ הוא אורך הרישא המקסימלי של P שהוא גם סיפא של $T[1..i]$, לכן נותר רק לברר (בעזרת הפונקציה N) את מספר הרישות של P שהם סיפות של $P[1..\tau(i)]$. נקרא לזה פונקציה N_T.

האלגוריתם:

1. **חשב** פונקציה π עבור P
2. $N_P(1) \leftarrow 1$
3. **לכל** i **החל** מ- 2 **וכלה** ב- m **בצע**
   a. $N_P(i) \leftarrow 1 + N_P(\pi(i))$
4. **הרץ** KMP(T,P) תוך חישוב הפונקציה τ עבור T
5. **לכל** i **החל** מ- 1 **וכלה** ב- n **בצע**
   a. $N_T(i) \leftarrow N_P(\tau(i))$
6. **החזר** N_T

סיבוכיות: $O(n + m)$.

5. נתונות שתי מילים A,B מעל $\sum$. תאר אלגוריתם שבודק האם A היא תמורה של B.
הנחה: $|\sum|=m$ ו- $|A|=|B|=n$.

פתרון:

רעיון:

נמיין את האותיות של A ושל B (בנפרד). כעת נשווה בין שתי הסדרות הממויינות עד ההבדל הראשון.

סיבוכיות: את המיון ניתן לעשות בשיטת Bucket Sort, ואז הסיבוכיות היא $O(n + m)$.

6. תאר אלגוריתם <u>לינארי</u>, הבודק אם טקסט T הוא <u>סיבוב מעגלי</u> של טקסט T'. T הוא סיבוב מעגלי של $T'=t_1t_2\ldots t_n$ אם קיים $1\le i\le n$ כך ש: $T=t_i\ldots t_n t_1..t_{i-1}$.
לדוגמא: 'arc' סיבוב מעגלי של car.

פתרון:

רעיון:

נשרשר שני עתקים של T. במחרוזת המתקבלת נחפש את T'.

האלגוריתם:

1. $X \leftarrow T\cdot T$
2. <u>**הרץ**</u> KMP(X,T')
3. <u>**אם**</u> T' נמצא ב-X <u>**החזר**</u> TRUE
4. <u>**אחרת**</u> <u>**החזר**</u> FALSE

סיבוכיות: O(n).

7. נניח שאנו מתירים לתבנית P להכיל תו מיוחד '*'. תו זה יכול להתאים לאפס או יותר תווים באלפבית ולטקסט T אסור להכיל תו זה.
   א. תארו אלגוריתם יעיל שיכריע האם P מופיע ב-T.
   ב. מהי סיבוכיות זמן הריצה של האלגוריתם שתיארת בסעיף א? הוכח!

פתרון:

רעיון:

אם P מכילה k כוכביות ('*'), אז נייצג את P באופן הבא: $P = P_0*P_1*...*P_k$. בייצוג זה כל P_i הוא תת-מחרוזת של P בין שתי כוכביות עוקבות. מאחר וכל * ניתנת להחלפה במספר כלשהו של תווים, עלינו רק לבדוק, שכל התת-מחרוזות הנ"ל (P_i לכל $0 \le i \le k$) נמצאות בטקסט T בסדר האינדקסים שלהן וללא חפיפות.

האלגוריתם:

1. $X \leftarrow T$
2. **<u>לכל</u>** i **<u>החל</u>** מ- 0 **<u>וכלה</u>** ב- k **<u>בצע</u>**
   a. **<u>הרץ</u>** $KMP(X,P_i)$ עד למציאת *הופעה ראשונה* תוך איתור מיקום ההופעה – אינדקס j
   b. **<u>אם</u>** לא נמצאה הופעה **<u>החזר</u>** FALSE
   c. **<u>אחרת</u>** **<u>בצע</u>** $X \leftarrow T[j+|P_i| .. n]$
3. **<u>החזר</u>** TRUE

סיבוכיות:

למרות שמריצים את KMP מספר פעמים, נעשה סה"כ מעבר אחד על הטקסט T, כי כל הרצת KMP עוקבת מתחילה מהמקום בו הסתיימה ההרצה הקודמת. גם על התבנית P עוברים פעם אחת בלבד, כי כל התת-מחרוזות שמופיעות בין הכוכביות אינן חופפות ב-P. מכאן, סיבוכיות האלגוריתם הנ"ל הינה כסיבוכיות של הרצה בודדת KMP(T,P). כלומר, $O(m + n)$.