

מבני נתונים ואלגוריתמים - הרצאה 12

15 בינואר 2012

דוגמה - KMP

T מחרוזת, P - תבנית.
 $P_n = P[0] \dots P[n-1]$ - רישא באורך n של P .
יש לנו טבלה המוגדרת ש $C[i]$ הוא ה j המקסימלי (הקטן מ- i) כך ש P_j סיפא של P_i , ו $C[0] = -1$.
אם $P = aabaa$ אז:

i	0	1	2	3	4	5
$C[i]$	-1	0	1	0	1	2

הערה

תהינה T, P כנ"ל אז KMP מחשב לכל $0 \leq i < \text{len}(T)$ את $\sigma[i] = j$ המקסימלי כך ש P_j סיפא של T_i .

דוגמה

אם $T = aaabaacaa$ ו $P = aabaa$ אז $\sigma(4) = 3$ ו $\sigma(5) = 4$.

תרגיל

עבור מחרוזת T נגדיר:

$$T^* = T[n-1]T[n-2] \dots T[0]$$

כאשר $n = \text{len}(T)$.
 $T = T^*$ נקראת פלינדרום אם $T = T^*$.
נתונה מחרוזת T , מצאו i מקסימלי כך ש T_i פלינדרום.

דוגמה

אם $T = abaababa$ אז T_1, T_3, T_6 פלינדרומים, צריך להחזיר 6.

פתרון

אם X רישא של T אז קיים Y כך ש $T = XY$.
לכן, $T^* = Y^*X^*$ ואם X פלינדרום אז $T^* = Y^*X$.
כלומר, X רישא של T שהיא סיפא של T^* .
לבית - הראו ש X רישא של T וסיפא של T^* $\iff X$ פלינדרום.
לכן מספיק למצוא את הרישא הארוכה ביותר של T שהיא סיפא של T^* .
נריך KMP עם מחרוזת T^* ותבנית T , ונחזיר את $\sigma[n] - j$ המקסימלי כך ש T_j סיפא של T_n^* .

אלגוריתם RK

מחפשים הרבה מחרוזות P_1, \dots, P_k (באורך כולל m) בתוך מחרוזת T .
זמן - $O(n+m)$.
זיכרון - $O(k)$ + אורך הקלט.
משתמשים בRolling Hash.

תרגיל

שתי מחרוזות X, Y באורך n נקראות אנגרמה זו לזו (X אנגרמה של Y) אם קיימת תמורה $\tau \in S_{\{0, 1, \dots, n-1\}}$ כך שמתקיים:

$$Y = X[\tau(0)] X[\tau(1)] \dots X[\tau(n-1)]$$

כלומר Y מתקבלת מ- X ע"י החלפת סדר האותיות. כתבו אלגוריתם המקבל מחרוזות P ו- T ומוצא את כל תתי המחרוזות של T שהן אנגרמות של P .

פתרון - דרך א'

נסמן $n = \text{len}(T)$ ו- $k = \text{len}(P)$.

יש $O(k!)$ אנגרמות ל- P .

נמצא את כולן (לבית - איך עושים זאת ב- $k!$).

נריץ RK הסיבוכיות - $O(n + k!)$.

פתרון - דרך ב'

נבחר Rolling Hash שהערך שלו על מחרוזת X לא מושפע מסדר האותיות של X . כעת אם X אנגרמה של Y אז הערך שלהם בפונק' הגיבוב שווה, ואז נריץ RK . לדוגמה, פונק' גיבוב כזו היא: לכל אות באלפבית $a \in \Sigma$ נבחר מספר גדול N_a ונגדיר:

$$RH(S) = \sum_{i=0}^{\text{len}(S)-1} N_{S[i]}$$

הסיבוכיות כעת תהיה $n + k$.

אלגוריתם BM

הרעיון הכללי הוא:

- מתחילים לחפש את P מהסוף והולכים אחורה.
- כאשר מגיעים לאי התאמה קופצים קדימה לפי המקסימום של שתי טבלאות - BST ו- GST .
- טבלת BST - מקבלת את התו במחרוזת T בו הייתה אי התאמה. נסמן $n = \text{len}(T)$ ו- $m = \text{len}(P)$.

$$BST[x] = \begin{cases} m & x \notin P \\ m - i - 1 & x = P_{m-1}[i] \end{cases}$$

לדוגמה - אם $P = abcba$:

$P_{m-1} = abcb$

$$BST[x] = \begin{cases} 4 & x = a \\ 1 & x = b \\ 2 & x = c \\ 6 & \text{else} \end{cases}$$

- GST - מקבלת את המיקום שלנו ב- P ($j = \text{האינדקס ב-} P \text{ בו ההשוואה נכשלה}$). α יסמן תו ששונה מ- $P[j]$ (תו כלשהו). התבנית αP אומרת כל מחרוזת מהצורה xP כאשר $x \neq P[j]$. נגדיר ${}_t P$ סיפא של P ממקום t : ${}_t P = P[t] P[t+1] \dots P[k]$. אם α (${}_{j+1} P$) מופיעה בתוך P באינדקס i (i מקסימלי) אז $GST[j] = m - 1 - i$. אם α (${}_{j+1} P$) לא מופיעה ב- P אבל P_i הוא סיפא של ${}_{j+1} P$ אז $GST[j] = 2m - j - t - 1$.

למשל, אם $P = abcba$ אז:

j	$\alpha_{j+1}P$	$GST[j]$
5	\hat{a}	1
4	$\hat{b}a$	5
3	$\hat{c}ba$	7
2	$\hat{b}cba$	8
1	$\hat{a}bcba$	9
0	$\hat{a}abcba$	10