התאמת מחרוזות

- נתונים טקסט T ותבנית P שניהם מעל אלפבית Σ
- מציאת כל המופעים של P בT (find)

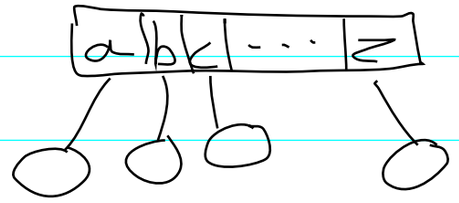
- בסיבוכיות זמן: $O(|T|+|P|)$ [Knuth Morris Pratt]
- שמירה נתונים מוכנים מראש (Indexing):
  - שמירה בסדר אותיות ואז שמירת עוד הרבה מבנים
  - מקרים $O(|T|)$ הזמן $O(|P|\cdot|T|)$
  - ניתן בנייה $O(|T|)$ בזמן $O(|T|+sort(\Sigma))$ כלשהו בΣ קצת אנלוגי
  - ניתן שאילתא $O(|P|)$

DNA /

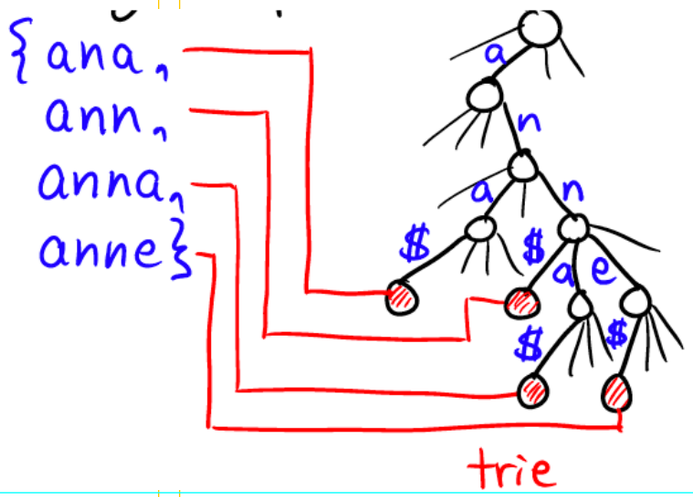
Trie

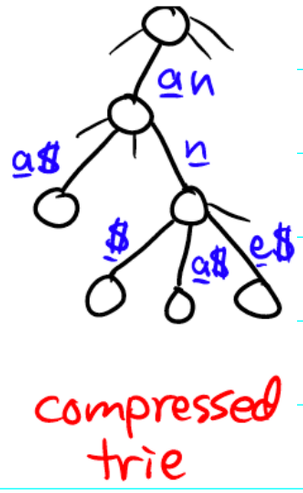
מבנה לשמירת מילון

- לכל צומת v נלכדים (על ידי הורה)



- מעבר שווים - עלה מוגדר מילה
- נוסיף לכל מילה כזו תו מיוחד $\$$ להבדיל בין מילים לרישות

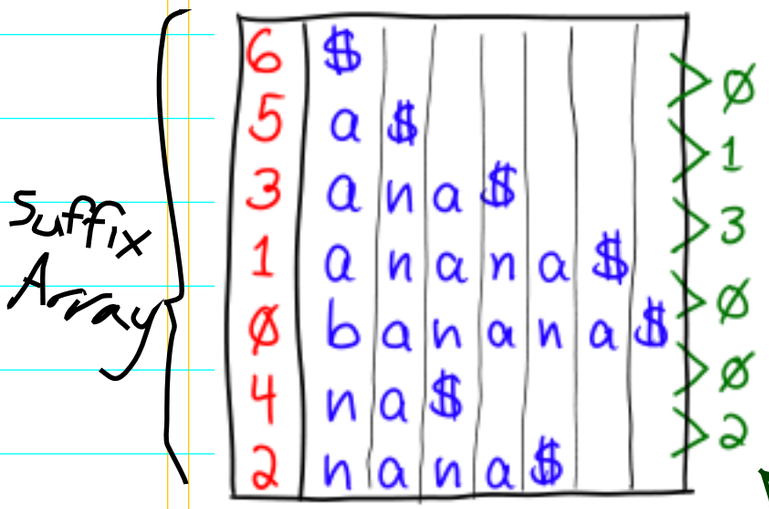


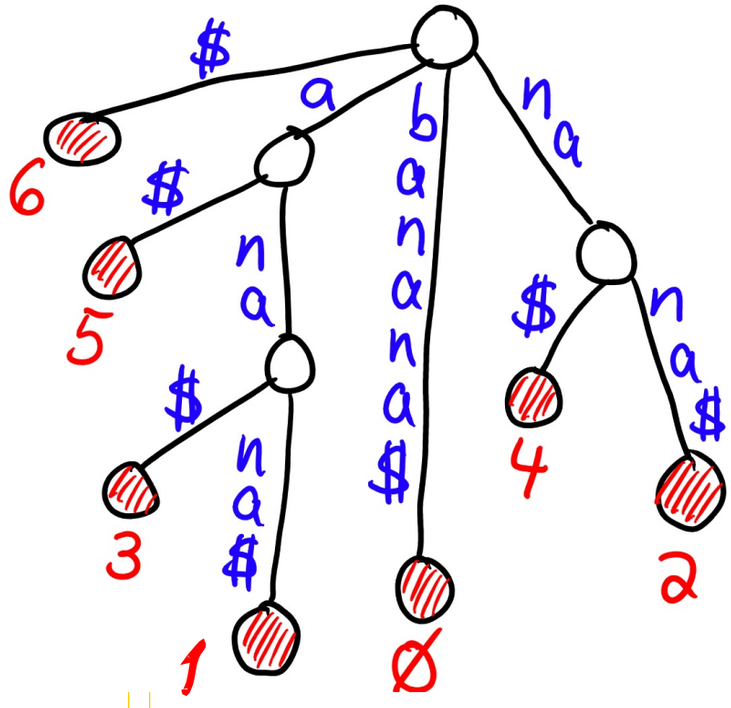


מבוא לעיבוד טקסט - "מבוגרים" מעלה על: Compressed Trie
פותרים בקשה, תרגיל בקורס או סיים אותה השנה

Suffix Tree

הסיבוכיות של כל $|T|+1$ ה-Compressed Trie של $T\#$





דוגמא: banana$
0 1 2 3 4 5 6

- $|T|+1$ עלים
- מתקבל על ידי מעבר על סדר מיון של ס"י שני מציתים ל-T

LCP Array

- $O(|T|)$ מקום. למה?

מגלת מחשב

- תיפוס ם לוקח $O(m)$

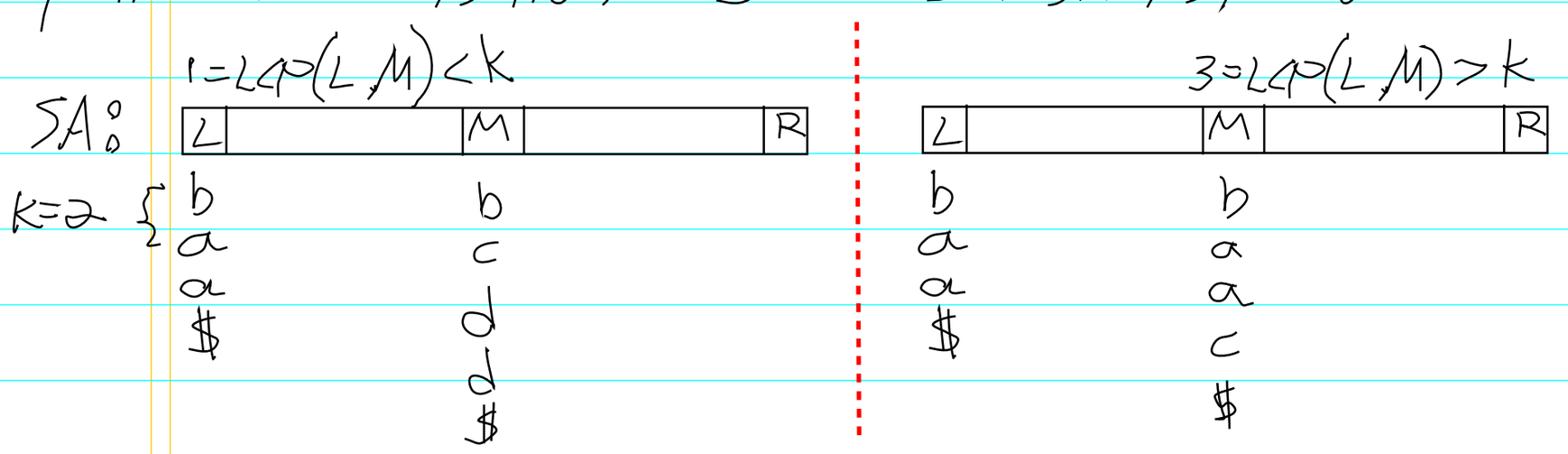
Suffix Array: האלגוריתם של הסיפורת שלהי מיון

- = הפלט של ה-suffix-tree משמש לי מיון
- ניתן למצוא זו בזמן $O(|P|\log|T|)$ כאשר
  ס"י מיפוש בינארי
- בא יחד עם מערך ה- LCP (בירור)
  כמה תווים צריך שיהיה בהתחלה $LCP[i]$ =
  הסיפול ה-i וה-i+1 במבר המיון

$$LCP(i,j) = RMQ(LCP[i],\dots,LCP[j])$$

↑ range minimum query

- בעזרת SA, LCP, RMQ ניתן לבדוק
  כל P בזמן $O(|P|+\log|T|)$:
  - נניח כי מחפשים P=bacde בתוך
    על SA, ונניח כי עד כה השווינו k=2 תווים



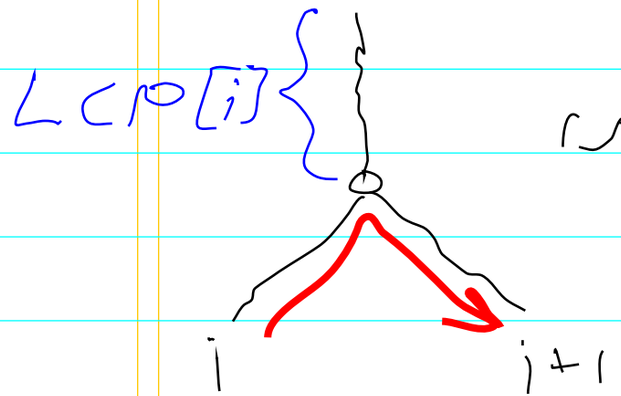
- אלגוריתם $k = LCP(L, M)$ לצורך מציאת לכל שורות
מערכים הגדל מהדרוש ה $i+1$
- כל שורה מושווה פעמים לכל היותר
- סה"כ $O(|T| \cdot (\text{מקס} + 1))$

שקילות SA ו- ST

$ST \Leftarrow SA + LCP$: פשוט לבדוק את הפלטים כמו ב-ST משמאל לימין.
ה-LCP הוא עומק ה-LCA של שני עלים עוקבים
באיטרציה ← → לא קשה לראות מיידית

$SA + LCP \Leftarrow ST$: מכניסים את הסיפורים ל-ST
את כל סיפ' עדיין ב-SA
- מנסים להכניס סיפ' עוקב. מטפסים מ-ST עד שהגענו
הנוכחי שכבר ל-LCP

לכל מטפסים על אותו
כמות פעמים
$O(|T|)$



בניית SA (+LCP) בזמן $O(|T|+Sort(\Sigma))$

[Kärkäinen & Sanders 2003]

T = a b c b b a d e a b c a

① ממיינים את Σ בזמן $Sort(\Sigma)$ ← בהינתן Radix Sort ב-$O(|T|)$

T_1 = 232 214 511 31

T = 1 2 3 2 2 1 4 5 1 2 3 1

T_0 = 123 221 451 231

② מרכיבים רקורסיבית את הסיפיקס של "$T_0 \cdot T_1$" ומוודאים "אי-זוגיות" (מיון האותיות ע"י Radix)

⇐ מקבלים את הסיפיות ממוינות + קל לשלוף

$$Suffixes(T) = Suffixes(T_0) + Suffixes(T_1) + Suffixes(T_2)$$

③ נמיין את $Suffixes(T_2)$ ע"י Radix באופן שנוכל בכל סיפא ב-$Suffixes(T_2)$: $\langle$ סיפא ב-T_0 , $a\rangle$

a ← את האות השנייה

סיפא ב-T_0 = מספר בין 1 ל $\frac{n}{3}$ ← מיקומה בסיפות של סיפא זו

- לאחר המיון, כל קשר בין שתי סיפות מקבות מתקבל ע"י קשר בין שתי סיפות ב-T_0 + תו ראשון

(4) כדי לקבל את SA נעשה Merge ל- $Suffixes(T_2)$ עם $Suffixes(T_0) + Suffixes(T_1)$:

- אם צריך להשוות סיפא של T_0 עם סיפא של T_2

$\langle a, T_0 \text{ סיפא של } \sim \rangle$ $\langle a, T_1 \text{ שאר של } \sim \rangle$

- אם צריך להשוות סיפא של T_1 עם סיפא של T_2

$\langle ab, T_0 \text{ סיפא של } \sim \rangle$ $\langle ab, T_1 \text{ שאר של } \sim \rangle$

- בשלופן פועה, אין קפלים את מערך סדר סי' אות ולו שיים בלוח סדר על $T_0 T_1$ ועל T_2

סה"כ: $T(n) = T\left(\frac{2}{3}n\right) + O(n) = O(n)$